DevTest Solutions

Utilisation de CA Application Test

Version 8.0

Support technique

Pour une assistance technique en ligne et une liste complète des sites, horaires d'ouverture et numéros de téléphone, contactez le support technique à l'adresse http://www.ca.com/worldwide.

Table des matières

Chapitre 1: Familiarisation avec le test d'application

Ce chapitre traite des sujets suivants :

Registre

Le registre fournit un emplacement central pour l'enregistrement de tous les composants de DevTest Server et de DevTest Workstation.

Il permet de suivre les emplacements de tous les composants d'exécution de DevTest et d'effectuer des recherches dans l'emplacement de chaque composant enregistré. Le registre fournit également les consoles Web pour l'administration des serveurs, la génération de rapports, le service de validation en continu et CA Continuous Application Insight. Le fournisseur JMS commun utilisé pour la communication entre les composants est démarré et exécuté lors du processus du registre. L'intermédiaire pour les agents Java déployés sur DevTest est également présent dans le registre.

Le nom complet du registre se présente comme suit :
tcp://nom-hôte-ou-adresse-IP:2010/nom-registre. Par exemple :

- tcp://localhost:2010/Registry
- tcp://myserver:2010/Registry
- tcp://myserver.example.com:2010/Registry
- tcp://172.24.255.255:2010/Registry

La station de travail DevTest Workstation comprend le moniteur de registre, qui permet de surveiller les scénarios de test, les simulateurs, les coordinateurs et les environnements virtuels pour une suite de tests.

Le registre est associé à au moins un laboratoire (page 366), nommé Default lab (Laboratoire par défaut). Si vous créez un coordinateur, un simulateur ou un serveur VSE sans spécifier un laboratoire, le serveur appartient au laboratoire par défaut.

Démarrage du registre

Si le registre n'est pas installé localement, connectez-vous au serveur sur lequel il est installé. Pour démarrer le registre, procédez comme suit.

Valide sous Windows

- Ouvrez une invite de commande, accédez au répertoire **LISA_HOME\bin** et entrez la commande suivante :

  ```
  Registre
  ```

- Cliquez sur le menu Démarrer, Tous les programmes, DevTest, Registry (Registre).
- Double-cliquez sur le fichier **Registry.exe** dans le répertoire **LISA_HOME\bin.**

Valide sous UNIX

- Ouvrez une fenêtre de terminal, accédez au répertoire **LISA_HOME\bin** et entrez la commande suivante :

  ```
  ./Registry
  ```

Patientez jusqu'à ce que le message suivant s'affiche :

```
Registry Ready (Le registre est prêt.).
```

Création d'un registre nommé

Pour créer un registre nommé, exécutez le fichier exécutable du registre avec l'option **-n** :

```
LISA_HOME\bin\Registry -n RegistryName
```

Les exemples suivants permettent de créer un registre nommé **registry1**.

Valide sous Windows

```
cd C:\Lisa\bin
Registry -n registry1
```

Valide sous UNIX

```
cd Lisa/bin
./Registry -n registry1
```

Modification du registre

Lorsque vous utilisez la station de travail DevTest Workstation, vous pouvez basculer vers un autre registre.

Procédez comme suit:

1. Dans le menu principal, sélectionnez System, Registry, Change DevTest Registry (Modifier le registre DevTest).

   La boîte de dialogue Set DevTest Registry (Définir le registre DevTest) s'affiche.

2. Saisissez le nom du registre, ou ouvrez la liste déroulante et sélectionnez un registre déjà utilisé.

3. Cochez ou décochez la case.

   Si l'option est **sélectionnée**, la boîte de dialogue Set DevTest Registry (Définir le registre DevTest) s'ouvrira lors du démarrage de la station de travail DevTest Workstation.

   Si elle est **désélectionnée**, la station de travail DevTest Workstation se connectera au dernier registre connecté lors du démarrage.

4. Cliquez sur OK.

Serveur de coordination

Les serveurs de coordination reçoivent les informations d'exécution de tests sous forme de documents et permettent de coordonner les tests exécutés sur un ou plusieurs serveurs de simulation (page 15). Lors de la simulation d'un test, sélectionnez le serveur de coordination à partir duquel le test sera coordonné. Spécifiez également un document de simulation qui spécifie les serveurs de simulation à utiliser lors de l'exécution des instances du test. Un serveur de coordination peut lancer des instances sur un serveur de simulation (en fonction du document de simulation).

Le serveur de coordination gère la collecte de mesures et la génération de rapports et communique les données de test à la station de travail DevTest Workstation à des fins de surveillance. En règle générale, un environnement DevTest Server peut contenir plusieurs serveurs de coordination.

Le serveur de coordination exécute des tests lorsqu'un document de simulation, un scénario de test et une configuration lui sont soumis.

La propriété **lisa.coordName** définit le nom par défaut d'un serveur de coordination.

`lisa.coordName=`*`Coordinator`*

Le nom complet d'un serveur de coordination est **tcp://nom-hôte_ou_adresse-IP:2011/nom-coordination**.

Création d'un serveur de coordination

Pour créer un serveur de coordination, exécutez la commande suivante :

```
LISA_HOME\bin\CoordinatorServer -n CoordinatorServerName -m RegistryName
```

Les exemples suivants permettent de créer un serveur de coordination nommé **coordinator1**. Le registre associé porte le nom **registry1**.

Valide sous Windows

```
cd C:\DevTest\bin
CoordinatorServer -n coordinator1 -m tcp://localhost:2010/registry1
```

Valide sous UNIX

```
cd DevTest/bin
./CoordinatorServer -n coordinator1 -m tcp://localhost:2010/registry1
```

Vous pouvez ajouter le coordinateur à un laboratoire (page 366) nommé à l'aide de l'option **-l**. Si vous ne spécifiez pas cette option **-l**, le coordinateur est ajouté au laboratoire par défaut.

Vous pouvez afficher le numéro de version à l'aide de l'option **--version.**

Pour plus d'informations, reportez-vous à la section Exécution des composants de serveur en tant que services de la rubrique *Administration.*

Surveillance des serveurs de coordination

Si la station de travail DevTest Workstation est connectée au registre associé, un serveur de coordination en cours d'exécution apparaîtra dans le moniteur de registre.

Pour surveiller des serveurs de coordination à partir du moniteur de registre :

1. Dans la station de travail DevTest Workstation, cliquez sur l'icône Toggle Registry (Basculer vers le registre).
2. Cliquez sur l'onglet Coord Servers (Serveurs de coordination).

Les serveurs de coordination s'affichent également dans le graphique de réseau de la console de serveur.

Pour surveiller des serveurs de coordination à partir de la console de serveur :

1. Dans le menu principal de la station de travail DevTest Workstation, sélectionnez View (Afficher), Server Console (Console de serveur).
2. Dans le graphique de réseau, cliquez sur le serveur de coordination.

   Une fenêtre contenant des informations s'affiche.

Serveur de simulation

Les serveurs de simulation exécutent les tests sous la supervision du serveur de coordination (page 13).

Les utilisateurs virtuels ou les instances de test sont créés et exécutés sur les serveurs de simulation. Le nombre d'utilisateurs virtuels, et donc le nombre de serveurs de simulation déployés, dépend de la nature des tests effectués. Chaque utilisateur virtuel communique avec le système client.

Si vous voulez effectuer des tests extensifs comprenant un grand nombre d'utilisateurs virtuels, vous pouvez répartir ces utilisateurs virtuels sur plusieurs serveurs de simulation.

La propriété **lisa.simulatorName** définit le nom par défaut d'un serveur de simulation.

```
lisa.simulatorName=Simulator
```

Le nom complet d'un serveur de simulation est **tcp://nom_hôte_ou_adresse_IP:2014/nom_simulateur**.

Création d'un serveur de simulation

Pour créer un simulateur, exécutez la commande suivante :

```
LISA_HOME\bin\Simulator -n SimulatorName -m RegistryName
```

Les exemples suivants permettent de créer un simulateur nommé **simulator1**. Le registre associé porte le nom **registry1**.

Valide sous Windows

```
cd C:\DevTest\bin
Simulator -n simulator1 -m tcp://localhost:2010/registry1
```

Valide sous UNIX

```
cd DevTest/bin
./Simulator -n simulator1 -m tcp://localhost:2010/registry1
```

Vous pouvez ajouter le simulateur à un laboratoire (page 366) nommé à l'aide de l'option -l. Si vous ne spécifiez pas cette option -l, le simulateur sera ajouté au laboratoire par défaut.

Vous pouvez spécifier le nombre d'utilisateurs virtuels pour ce simulateur à l'aide de l'option -i. Par exemple :

```
Simulator -n simulator1 -m tcp://localhost:2010/registry1 -i 100
```

Vous pouvez afficher le numéro de version à l'aide de l'option --version.

Lorsque vous créez un simulateur, vous pouvez remplacer le numéro de port par défaut en ajoutant deux points et le numéro de port non défini par défaut à l'option -n. Par exemple :

```
Simulator -n testSim1:35001
```

Pour exécuter plusieurs simulateurs, utilisez l'invite de commande pour créer les simulateurs comme indiqué plus haut.

Si le numéro de port n'est pas spécifié dans le nom, le simulateur tentera d'abord d'utiliser le port 2014. Si le port 2014 est occupé, le simulateur tentera d'utiliser le port 2015, puis 2016, ainsi de suite jusqu'au port 2024 avant de s'arrêter.

Pour plus d'informations sur l'utilisation d'un port, reportez-vous à la section Numéros de port par défaut de la rubrique *Administration.*

Pour plus d'informations sur l'exécution du simulateur en tant que service, consultez la section Exécution des composants de serveur comme services de la rubrique *Administration*.

Surveillance des serveurs de simulation

Si la station de travail DevTest Workstation est connectée au registre associé, un serveur de coordination en cours d'exécution apparaîtra dans le moniteur de registre.

Pour surveiller des serveurs de simulation à partir du moniteur de registre :

1. Dans DevTest Workstation, cliquez sur l'icône Toggle Registry (Basculer vers le registre).
2. Cliquez sur l'onglet Simulators (Simulateurs).

Les serveurs de simulation s'affichent également dans le graphique de réseau de la console Server Console (Console de serveur).

Pour surveiller des serveurs de simulation à partir de la console de serveur :

1. Dans le menu principal de DevTest Workstation, sélectionnez View (Afficher), Server Console (Console de serveur).
2. Dans le graphique de réseau, cliquez sur le simulateur.

   Une fenêtre contenant des informations s'affiche.

Utilitaires de ligne de commande

Le répertoire **LISA_HOME\bin** contient les utilitaires de ligne de commande suivants :

Outil de ligne de commande CAI

La commande **PFCmdLineTool** permet d'effectuer diverses tâches CA Continuous Application Insight à partir de la ligne de commande. Pour plus d'informations, consultez la section Outil de ligne de commande de CAI dans la rubrique *Utilisation de CA Continuous Application Insight*.

Gestionnaire CVS Manager

Le CVS Manager (Gestionnaire de services de validation en continu) permet d'ajouter des moniteurs au CVS Dashboard (Tableau de bord du service CVS) ou de les supprimer via une option de ligne de commande. Pour plus d'informations, consultez la section CVS Manager (Gestionnaire de services de validation en continu) (page 401) de la rubrique *Utilisation de CA Application Test.*

Utilitaire Make Mar

L'utilitaire Make Mar permet d'afficher le contenu de fichiers d'informations MAR (en mode autonome ou dans une archive), ou de créer des fichiers d'archive de modèle à partir de fichiers d'informations MAR. Pour plus d'informations, consultez la section Utilitaire Make Mar (page 289) de la rubrique *Utilisation de CA Application Test.*

Gestionnaire d'images de service

Le Service Image Manager (gestionnaire d'images de service) permet d'importer des transactions dans une image de service (nouvelle ou existante) et de combiner deux ou plusieurs images de service. Pour plus d'informations, consultez la section ServiceImageManager Manager (Gestionnaire d'images de service) de la rubrique *Utilisation de CA Service Virtualization.*

Service Manager (Gestionnaire de services)

Le Service Manager (Gestionnaire de services) permet de vérifier le statut d'un processus de serveur en cours d'exécution, de le réinitialiser, ou de l'arrêter. Pour plus d'informations, consultez la section Service Manager (Gestionnaire de services) de la rubrique *Administration*.

Utilitaire Test Runner (Exécuteur de tests)

L'exécuteur de tests Test Runner est une version de DevTest Workstation sans périphérique de contrôle, proposant la même fonctionnalité, mais sans aucune interface utilisateur. Pour plus d'informations, consultez la section Utilitaire Test Runner (Exécuteur de tests) (page 341) de la rubrique *Utilisation de CA Application Test.*

VSE Manager (Gestionnaire d'environnements de services virtuel)

Le gestionnaire VSE Manager (Gestionnaire d'environnements de services virtuel) est utilisé pour gérer des environnements de service virtuel. Pour plus d'informations, consultez la section Commandes du gestionnaire du VSE de la rubrique *Utilisation de CA Service Virtualization.*

Chapitre 2: Utilisation du portail DevTest avec CA Application Test

Ce chapitre traite des sujets suivants :

Ouverture du portail DevTest

Ouvrez le portail DevTest à partir d'un navigateur Web.

Remarque : Pour plus d'informations sur les composants de serveur qui doivent être exécuté, reportez-vous à la section Démarrage des processus ou services DevTest de la rubrique *Installation*.

Procédez comme suit:

1. Effectuez l'une des opérations suivantes :
   - Dans un navigateur Web, entrez **http://localhost:1507/devtest**. Si le registre se trouve sur un ordinateur distant, remplacez **localhost** par le nom ou l'adresse IP de l'ordinateur.
   - Sélectionnez View (Afficher), DevTest Portal (Portail) à partir de la station de travail DevTest Workstation.
2. Entrez votre nom d'utilisateur et votre mot de passe.
3. Cliquez sur Log in (Connexion).

Chapitre 3: Création d'un test API

La fenêtre Create API Test (Créer un test API) permet de créer un test manuellement ou avec des paires demande/réponse à partir d'un test d'API existant ou à partir du système de fichiers.

Procédez comme suit:

1. Sélectionnez un projet dans la liste déroulante Project, ou entrez un nom de projet. Si le projet existe, ce test d'API y sera ajouté. Sinon, il sera créé.
2. Saisissez le nom de votre test dans le champ Test Name (Nom du test). Après l'exécution du test, l'application ajoutera la date et l'heure au nom saisi de sorte qu'il soit unique.
3. Entrez la partie de base de l'URL pour les paires demande/réponse dans le champ Base URL (URL de base). L'URL de base est l'URL à laquelle une demande se connecte lors de son exécution. Par exemple, saisissez :

   ```
   http://localhost:8080
   ```
4. Entrez une description courte du test dans le champ Description.

Ce chapitre traite des sujets suivants :

Création d'un test via l'importation de paires demande/réponse à partir d'un test

Vous pouvez créer un test en chargeant un test API existant dans l'éditeur de tests.

Procédez comme suit:

1. A partir de la barre d'outils R/R Pair Editor (Editeur de paires demande/réponse), cliquez sur Import R/R pairs from existing tests (Importer des paires demande/réponse à partir de tests existants) .

   La fenêtre Load Existing API Tests (Charger des tests API existants) s'ouvre.
2. La liste affiche tous les projets disponibles sur le serveur de ressource et les tests API disponibles sous chaque projet.

   Si vous n'avez pas créé de test d'API dans la console DevTest, aucun élément ne sera visible dans la liste déroulante.
3. Pour filtrer les noms de projet, saisissez une chaîne dans le champ Filter (Filtrer).
4. Pour importer un test API, sélectionnez-le et cliquez sur Load (Charger).

Création d'un test via l'importation de paires demande/réponse à partir d'un fichier

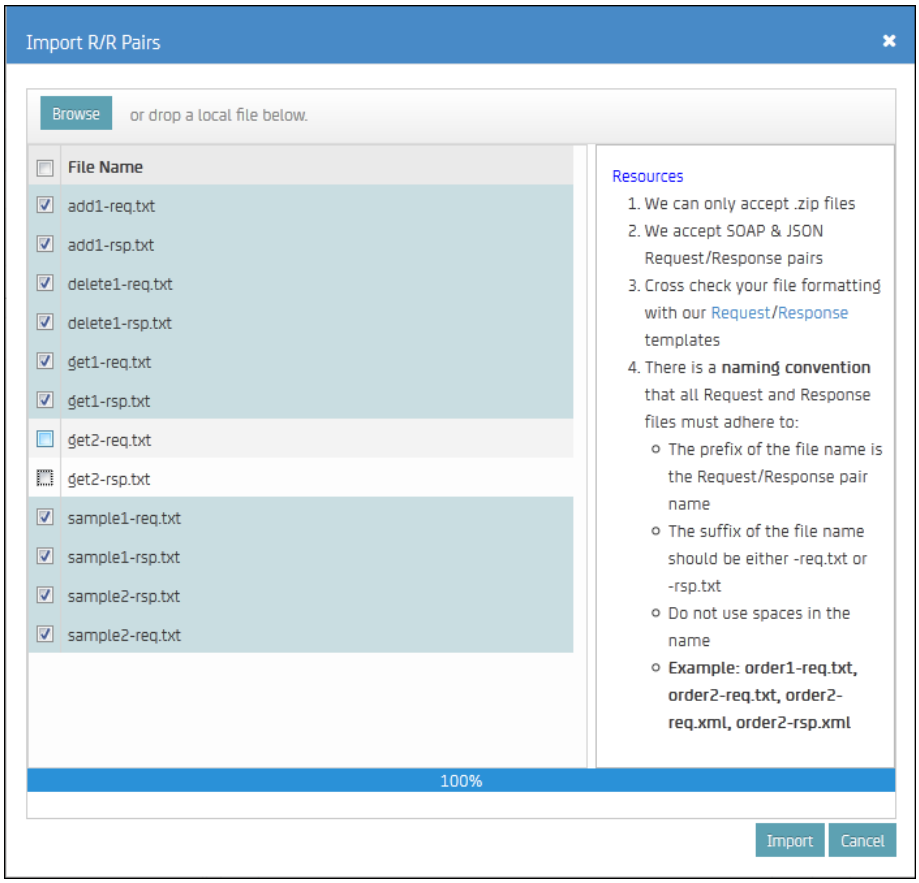
Vous pouvez créer un test en chargeant des paires demande/réponse à partir d'un fichier .zip dans l'éditeur de tests.

Procédez comme suit:

1. A partir de la barre d'outils R/R Pair Editor (Editeur de paires demande/réponse), cliquez sur Import R/R pairs from local zip files (Importer des paires demande/réponse à partir de fichiers .zip locaux existants)  .

   La fenêtre Import R/R Pairs (Importer des paires demande/réponse) s'ouvre.

2. Pour désigner un fichier .zip contenant des fichiers de paires demande/réponse, parcourez le système de fichiers, ou utilisez la fonction glisser-déposer pour déplacer un fichier .zip dans la fenêtre.



   Notez la configuration requise pour les fichiers de paires demande/réponse dans la section Resources de la fenêtre.

3. Sélectionnez les fichiers de paires demande/réponse à importer. Pour sélectionner tous les fichiers de paires demande/réponse, sélectionner la case à cocher File Name (Nom de fichier).
4. Cliquez sur Import (Importer).

Création manuelle d'un test

Pour créer un test manuellement, utilisez l'éditeur de tests.

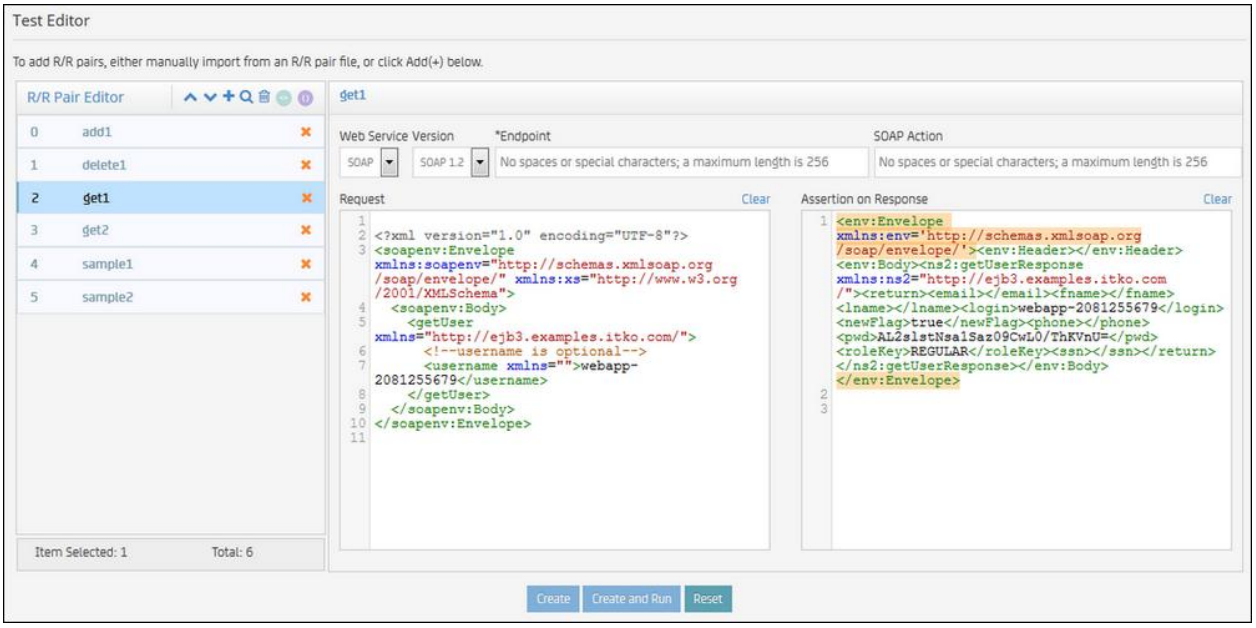
Procédez comme suit:

1. Cliquez sur Add a R/R Pair (Ajouter une paire de demande/réponse)  .

   La première paire de demande/réponse, portant le numéro 0, s'affiche. Son nom par défaut est **Baseline-0**. Pour la renommer, double-cliquez sur le nom. A l'issue de la modification, appuyez sur Entrée pour fermer le mode d'édition.
2. Dans le volet droit, sous le nom de la paire de demande/réponse, sélectionnez Web Service dans la liste déroulante. Les options valides sont REST et SOAP.
3. Effectuez l'une des opérations suivantes :
   - Dans le cas de REST, entrez un type (GET, POST, PUT, DELETE, ou HEAD) et un Resource Path (chemin de ressource, chemin au service Web).
   - Dans le cas de SOAP, entrez une version (SOAP 1.2 ou SOAP 1.1), un Endpoint (Terminal, terminal spécifique à cette demande) et une SOAP Action (facultative dans le cas de SOAP 1.2).
4. Entrez le texte de la demande dans le champ Request (Demande).
5. Entrez le texte de la réponse dans le champ Assertion on Response (Assertion portant sur la réponse).

Modification d'un test API

Pour modifier un test, utilisez l'éditeur de tests.



Pour changer le nom d'un test, double-cliquez sur le nom, entrez un nouveau nom et cliquez sur Enter (Entrée).

Pour créer un test API, cliquez sur le bouton Create (Créer). Si le test est créé, il sera automatiquement ouvert dans la page de l'éditeur de tests.

Pour créer et exécuter le test API, cliquez sur le bouton Create and Run (Créer et exécuter). Si le test est créé, le portlet de surveillance s'affichera.

Important : Il est très probable que DevTest ajoute l'extension .invalid à un nom de fichier de test, car une classe Java requise pour le test (par exemple, dans une étape d'exécution Java dynamique) n'est pas disponible dans DevTest. Lors de l'enregistrement ou de la fermeture de ce type de fichier de test, DevTest fait une copie du test et renomme la copie du test en ajoutant l'extension .invalid.

Chapitre 4: Surveillance de tests

La fenêtre Monitor Test (Surveiller des tests) permet d'afficher des tests API, des scénarios de test et des suites de tests exécutés dans CA Application Test.

Ce chapitre traite des sujets suivants :

Filtre de tests et de suites

Pour filtrer les scénarios de test et les suites de tests renvoyés à partir de la base de données, cliquez sur Search (Rechercher) .

Entrez les paramètres suivants.

Source

Définit les date et heure de début des scénarios de test et des suites de tests.

To (A)

Définit les date et heure de fin des scénarios de test et des suites de tests.

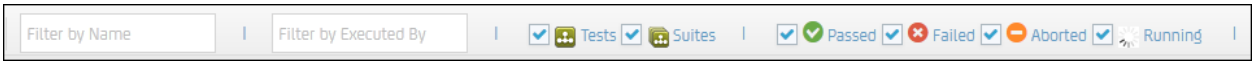
Executed By (Exécuté par)

Désigne l'utilisateur qui a soumis le scénario de test ou la suite de tests.

Pour afficher des scénarios de test et des suites de tests exécutés la semaine précédente, laissez la valeur par défaut Last 7 Days (7 derniers jours). La liste déroulante permet de sélectionner diverses plages de date et d'heure.

Pour appliquer votre filtre, cliquez sur Apply (Appliquer).

Lorsque les résultats sont affichés, vous pouvez filtrer davantage à l'aide des options de filtre dans la partie supérieure du volet.



Filter by Name (Filtrer par nom)

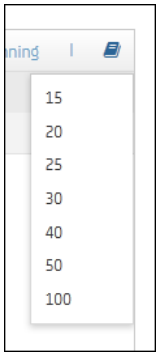
Pour afficher uniquement les résultats d'un scénario de test ou d'une suite spécifique, saisissez un nom complet ou partiel du test ou de la suite.

Filter by Executed By (Filtrer par auteur d'exécution)

Pour afficher uniquement les scénarios de test et les suites soumis par un utilisateur spécifique, entrez un ID d'utilisateur complet ou partiel.

Vous pouvez également filtrer par type, en sélectionnant Tests ou Suites. Pour filtrer par résultat, sélectionnez Passed (Réussite), Failed (Echec), Aborted (Interrompu), ou Running (En cours d'exécution).

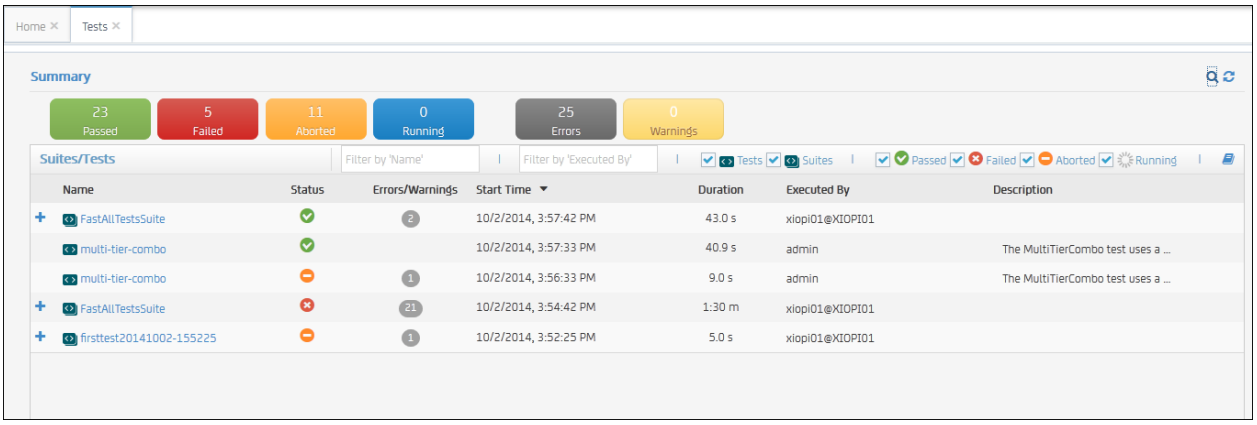
Pour sélectionner le nombre de lignes à afficher, cliquez sur Page Size (Taille de la page).


15
20
25
30
40
50
100

Surveillance de tests et de suites

Le volet Suites/Tests affiche les scénarios de test et les suites qui correspondent au critère de filtrage saisis.

Les icônes Summary (Récapitulatif) affichent le nombre de cycles réussi, en échoué, interrompus, ou en cours d'exécution.



Les champs suivants sont affichés :

nom

Affiche le nom du scénario de test ou de la suite. Le nom d'une suite est précédé du signe plus ou moins, sur lequel vous cliquez pour développer ou réduire les scénarios de test dans la suite. Le nom des suites de tests créées avec le portail de DevTest Solutions contient la date et l'heure.

Passez la souris sur le nom pour afficher une icône Info à droite du nom. Cliquez sur cette icône pour plus d'informations sur le test ou la suite.

Statut

Indique le statut du scénario de test ou de la suite de tests : Passed (Réussite), Failed (Echec), Aborted (Interrompu), ou Running (En cours d'exécution).

Errors/Warnings

Affiche le nombre d'erreurs (affiché dans un cercle gris) et le nombre d'avertissements (affiché dans un cercle jaune) pour chaque scénario de test et suite de tests.

Heure de début

Affiche les date et heure de démarrage de chaque scénario de test et suite de tests.

Durée

Affiche, en secondes, la durée de l'exécution du scénario de test ou de la suite de tests.

Executed By (Exécuté par)

Affiche l'ID de l'utilisateur ayant soumis le scénario de test ou la suite de tests.

Description

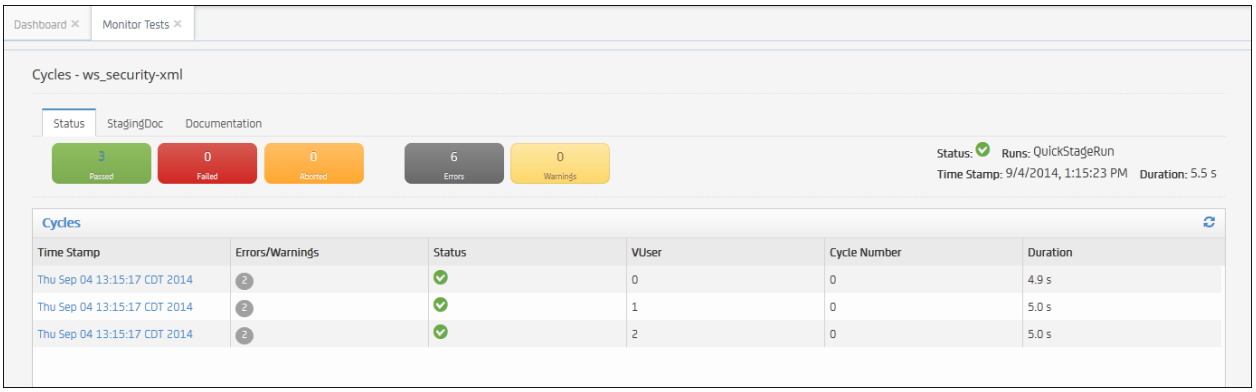
Affiche la description du scénario de test ou de la suite de tests.

Affichage des détails de tests et de suites

Dans le volet Suites/Tests, cliquez sur un nom de scénario de test pour afficher des détails sur son exécution de ce scénario. L'onglet Status (Statut) s'affiche.

Scénarios de test avec plusieurs cycles

Pour un scénario de test comprenant plusieurs cycles, les résultats des cycles s'affichent avec le statut de chacun d'eux.



Les champs suivants sont affichés pour chaque cycle :

Time Stamp (Horodatage)

Affiche les date et heure de démarrage de chaque cycle dans le scénario de test.

Errors/Warnings

Affiche le nombre d'erreurs (affiché dans un cercle gris) et le nombre d'avertissements (affiché dans un cercle jaune) pour chaque scénario de test et suite de tests.

Statut

Indique le statut du scénario de test ou de la suite de tests : Passed (Réussite), Failed (Echec), Aborted (Interrompu), ou Running (En cours d'exécution).

VUser (utilisateur virtuel)

Affiche le nombre d'utilisateurs virtuels pour chaque cycle.

Cycle Number (Numéro du cycle)

Affiche un index d'exécution des cycles. Si l'exécution du test comprend un seul cycle, le Cycle Number (Numéro du cycle) sera défini sur 0.

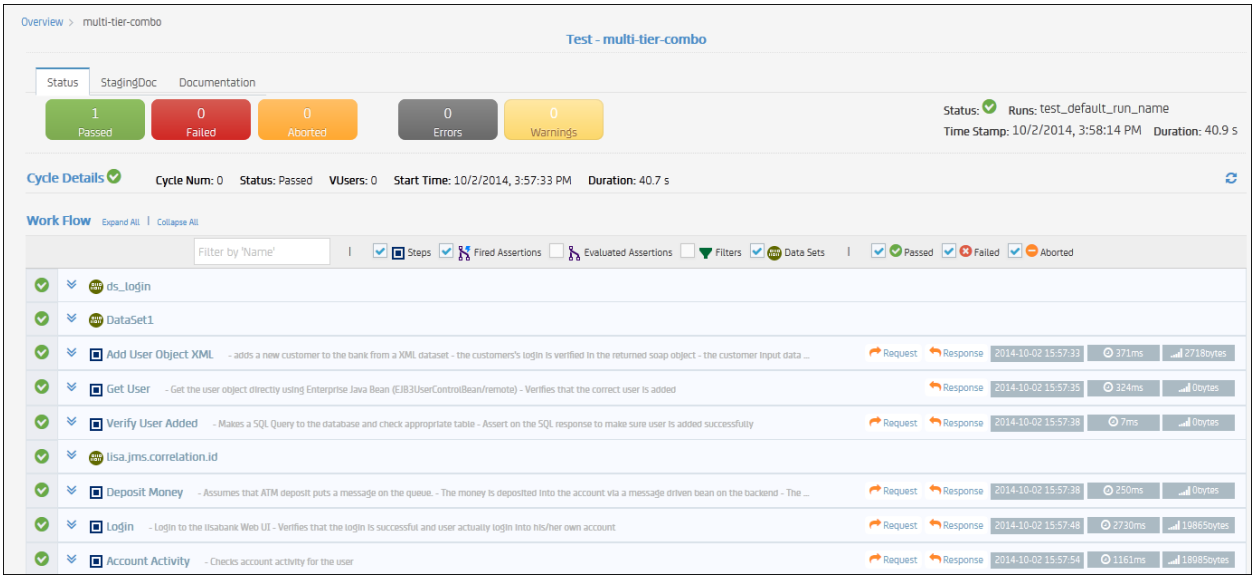
Durée

Affiche, en secondes, la durée de l'exécution du scénario de test ou de la suite de tests.

Cycle Details (Détails des cycles)

Pour afficher les détails des cycles, effectuez l'une des actions suivantes :

- Dans le cas de scénarios de test comprenant plusieurs cycles, double-cliquez sur l'horodatage du cycle.
- Dans le cas de scénarios de test comprenant un seul cycle, double-cliquez sur le nom du scénario de test.



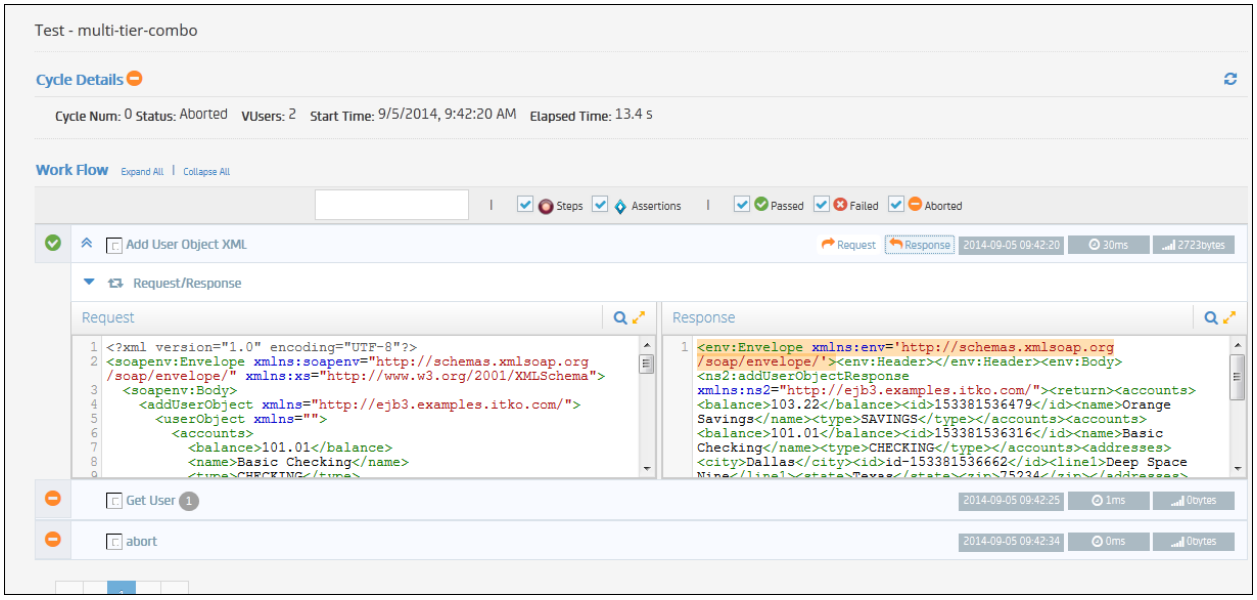
Chaque étape de test dans le scénario de test s'affiche. Tous les ensembles de données, les filtres (globaux et non globaux), et les assertions (globales et non globales) associés à chaque étape s'affichent également.

Pour développer toutes les étapes dans le scénario de test, cliquez sur Expand All (Développer tout). Pour réduire toutes les étapes, cliquez sur Collapse All (Tout réduire).

A gauche de chaque nom d'élément, une flèche  s'affiche. Pour afficher des détails sur l'élément, notamment la demande/réponse, les événements, les messages d'erreur ou d'avertissement et les propriétés, cliquez sur la flèche.

Si une étape a produit des erreurs ou des avertissements, le nombre d'erreurs s'affiche dans un cercle gris et le nombre d'avertissements le cercle orange, à droite du nom de l'étape. Passez la souris sur le nombre pour afficher le texte d'erreur ou d'avertissement.

Si une étape a une demande et une réponse, cliquez sur Request (Demande) ou Response (Réponse) pour afficher les détails de la demande ou de la réponse.



Pour rechercher le texte d'une demande ou d'une réponse, cliquez sur Search (Rechercher) . Entrez une nouvelle chaîne et cliquez sur Enter (Entrée).

Remarque : Si aucun événement ou aucune propriété n'est visible pour une étape qui en possédait, consultez le document de simulation de ce test pour veiller à ce qu'il soit défini sur des événements d'enregistrement, des demandes, des réponses, des captures d'écran et des propriétés, définis ou référencés. Vous pouvez utiliser le document de simulation **Run1User1CycleShowAll** pour veiller à ce que tous les éléments soient capturés.

Pour développer la vue d'une demande ou d'une réponse, cliquez sur Expand (Développer) .

Pour afficher le document de simulation utilisé par un test, cliquez sur l'onglet Staging Doc (Document de simulation).

Pour afficher la documentation associée à un test, cliquez sur l'onglet Documentation.

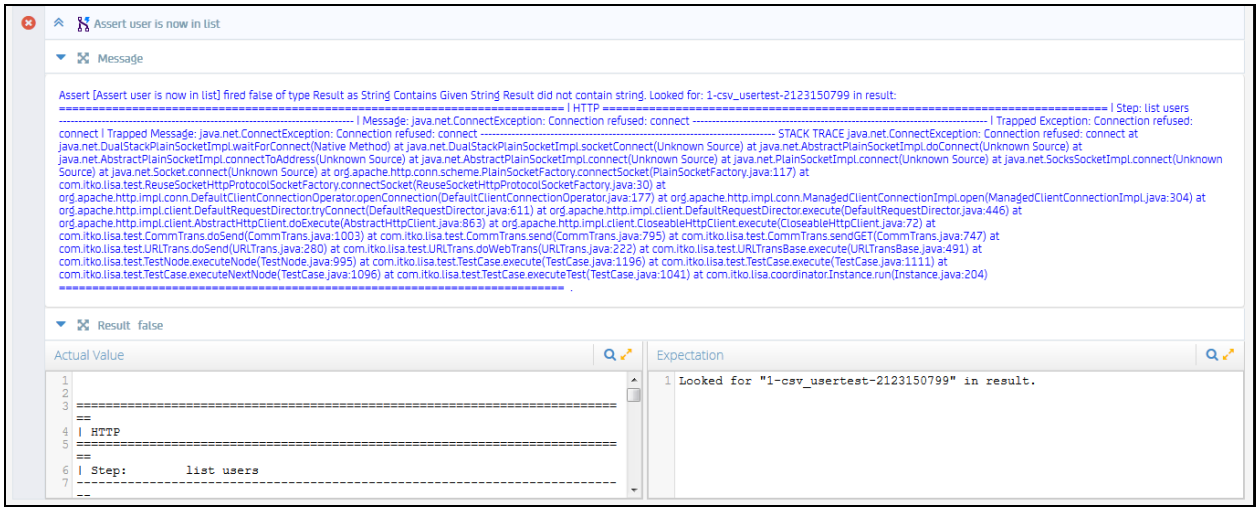
Affichage de détails d'assertions, de filtres, d'ensembles de données et de compagnons

Pour afficher les détails des résultats d'assertions, de filtres, d'ensembles de données et de compagnons, cliquez sur la flèche  à gauche du nom de l'élément.

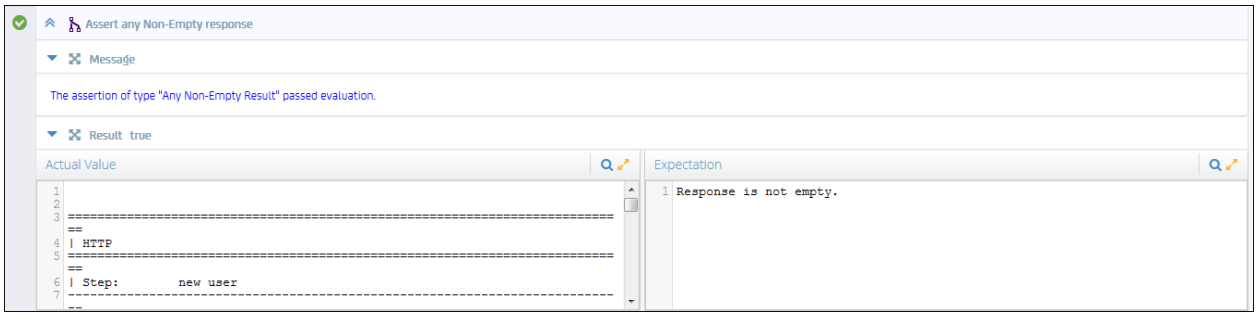
Détails d'assertions

Pour afficher les détails d'assertions déclenchées tirées ou d'assertions évaluées, vous pouvez sélectionner la case à cocher appropriée dans la barre de filtre.

Les détails des assertions déclenchées affichent le message produit par l'assertion et son résultat.

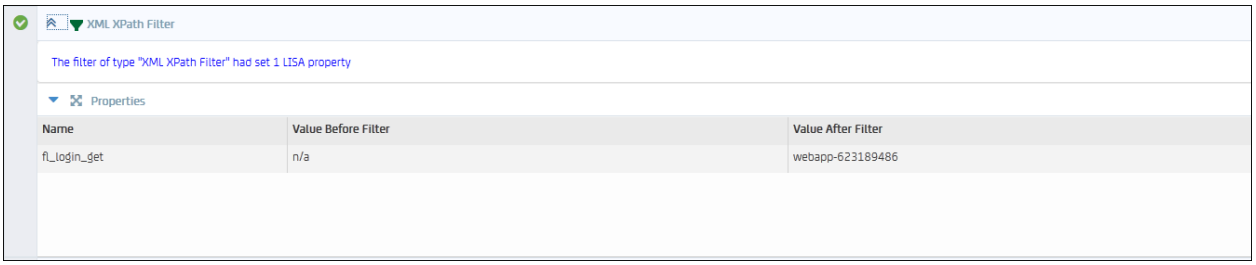


Les détails des assertions évaluées affichent le message produit par l'assertion et son résultat.



Détails de filtres

Les détails des filtres affichent les propriétés définies par le filtre.



Détails d'ensembles de données

Les détails des ensembles de données affichent le contenu d'ensembles de données et les propriétés qu'ils ont modifiées.

Détails de compagnons

Les détails des compagnons affichent l'action effectuée par le compagnon.

Chapitre 5: Gestion des artefacts de test

Pour gérer des artefacts de test, utilisez l'option Manage (Gérer) dans la barre de navigation gauche ou sélectionnez Manage Test Artifacts (Gérer des artefacts de test) à partir du portlet Quick Links (Liens rapides) sur le portail.

Pour gérer des tests créés à l'aide du portail DevTest, sélectionnez Manage API Tests (Gérer des tests d'API).

Pour gérer des scénarios de test créés via DevTest Workstation, sélectionnez Manage Test (Gérer des tests).

Pour gérer des suites de tests créées via DevTest Workstation, sélectionnez Manage Test (Gérer des tests).

Pour sélectionner un projet, utilisez la liste déroulante Project. Les projets disponibles dans la liste déroulante sont ceux du répertoire Projects du répertoire LISA_HOME.

Ce chapitre traite des sujets suivants :

Chapitre 6: Gestion des tests d'API

L'option Manage API Tests (Gérer des tests d'API) vous permet de modifier des tests créés dans le portail DevTest.

Procédez comme suit:

1. Sélectionnez un test dans la liste de tests, et cliquez-le dessus.

   L'éditeur de tests s'ouvre.

2. Utilisez l'éditeur de tests tel que décrit dans la rubrique Modification d'un test d'API (page 27).

Chapitre 7: Gestion des tests

Pour exécuter un test, cliquez avec le bouton droit de la souris sur son nom et sélectionnez Run (Exécuter).

Pour supprimer un test, cliquez avec le bouton droit de la souris sur son nom et sélectionnez Delete (Supprimer).

Pour télécharger le fichier .mar (page 278) pour le test, cliquez avec le bouton droit de la souris sur le nom du test et sélectionnez Download Mar (Télécharger le fichier .mar).

Pour afficher des informations et la documentation sur le test, cliquez sur son nom.

Chapitre 8: Gestion des suites de tests

Pour exécuter une suite de tests, cliquez avec le bouton droit de la souris sur son nom et sélectionnez Run (Exécuter).

Pour supprimer une suite de tests, cliquez avec le bouton droit de la souris sur son nom et sélectionnez Delete (Supprimer).

Pour télécharger le fichier .mar (page 278) pour la suite de tests, cliquez avec le bouton droit de la souris sur le nom de la suite et sélectionnez Download Mar (Télécharger le fichier .mar).

Pour afficher des informations et la documentation sur la suite de tests, cliquez sur son nom.

Chapitre 9: Utilisation de la Workstation (Station de travail) et de la console avec CA Application Test

Ce chapitre traite des sujets suivants :

Présentation de la station de travail et de la console

Cette section comprend les rubriques suivantes :

DevTest Workstation

DevTest Workstation est un environnement intégré pour le développement, la simulation et la surveillance de tests.

Vous pouvez utiliser une version de DevTest Workstation ou un environnement DevTest Server.

Dans la station de travail DevTest Workstation, les tests sont gérés et exécutés dans l'environnement de la station de travail. DevTest Workstation est un client de test utilisé par les équipes de QA/QE, de développement et d'analyse métier pour tester les composants suivants :

- Navigateur riche et interfaces utilisateur Web
- Eléments structuraux sous l'interface utilisateur

DevTest Workstation permet de créer et de simuler le test des unités, des fonctions, de l'intégration, de la régression et des processus métier, sans devoir recourir à du code. L'exécution de la station de travail DevTest Workstation requiert un registre. Les tests sont gérés et exécutés dans l'environnement DevTest Workstation (IDE de création de test), qui exécute des serveurs de coordination et de simulation intégrés.

Dans DevTest Server, les tests sont également gérés et exécutés dans l'environnement Workstation (Station de travail). La station de travail se connecte au serveur à déployer et surveille les tests développés dans DevTest Workstation.

Le moteur côté serveur pour les scénarios de test et les services virtuels DevTest (IDE de création modèle de test et de service virtuel) permet de gérer, planifier et diriger des scénarios de test DevTest pour des tests d'unités, de fonctions, de charge et de performances, de manière régulière.

Ouverture de DevTest Workstation

Lorsque vous ouvrez la station de travail DevTest Workstation, vous êtes invité à spécifier un registre (page 11).

Si DevTest Server est installé sur votre ordinateur, vous pouvez utiliser :

- Un registre exécuté sur un ordinateur local
- Un registre qui s'exécute sur un ordinateur distant.

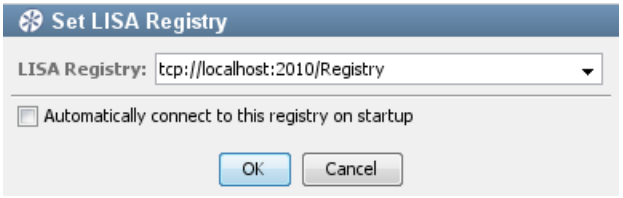
Utilisez un registre exécuté sur un ordinateur distant si la station de travail DevTest Workstation est déjà installée sur votre ordinateur.

Pour plus d'informations sur la spécification du registre avec SSL activé, Reportez-vous à la section Utilisation du protocole SSL pour sécuriser la communication de la rubrique *Administration.*

Pour ouvrir DevTest Workstation,

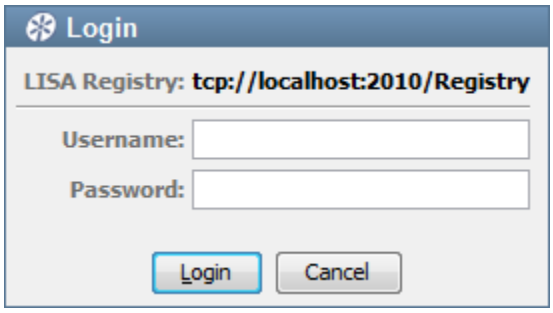
1. Effectuez l'une des opérations suivantes :
   - Ouvrez une invite de commande, accédez au répertoire **LISA_HOME\bin** et exécutez le fichier exécutable de DevTest Workstation.
   - Si vous disposez d'une icône d'application DevTest  sur votre bureau, double-cliquez dessus.
   - Cliquez sur le menu Démarrer, Tous les programmes, DevTest, DevTest Workstation.

   La boîte de dialogue Set DevTest Registry (Définir le registre DevTest) s'affiche.



2. Acceptez le registre par défaut ou spécifiez un registre différent. DevTest Workstation étant une interface utilisateur graphique, il n'est pas possible de définir un registre distant par défaut lors du démarrage. Spécifiez un registre dans cette invite, ou si vous souhaitez modifier de registre, utilisez la commande Toggle Registry (Basculer vers le registre).
3. Cliquez sur OK.

   La boîte de dialogue Login (connexion) apparaît.



4. Saisissez votre nom d'utilisateur et votre mot de passe, puis cliquez sur Login (Connexion).

Menu Main (Principal)

Le menu Main (Principal) de DevTest Workstation comprend des options pour toutes les fonctions principales disponibles. Les options disponibles sont dynamiques et correspondent aux sélections actuelles. Certains menus, listes déroulantes et barres d'outils sont toujours disponibles dans DevTest Workstation.

Remarque : Les éléments de ce menu sont dynamiques. L'administrateur DevTest contrôle les éléments disponibles dans votre profil de sécurité.

Panneau Project (Projet)

Un projet est une collection de fichiers DevTest liés. Les fichiers peuvent inclure des scénarios de test, des suites, des modèles de service virtuel, des images de service, des configurations, des documents d'audit, des documents de simulation, des ensembles de données, des moniteurs et des fichiers d'informations MAR.

Dans la station de travail DevTest Workstation, un seul projet peut être ouvert à la fois.

Le répertoire **LISA_HOME\Projects** est créé lors de la première création de projet (page 65).

Tous les dossiers du panneau Project (Projet) de DevTest Workstation sont identiques à ceux créés dans le système de fichiers. Vous pouvez consulter le répertoire **LISA_HOME** pour afficher la structure de répertoires et les fichiers.

Vous pouvez importer des fichiers dans un projet.

Disposition du panneau Project (Projet)

Le panneau Project contient une arborescence de projets.

Lorsque vous créez un projet depuis le début, la structure de l'arborescence de projets est la suivante :

- VirtualServices
  - Images
  - VRScenarios
- Tests
  - Subprocesses
  - AuditDocs
  - Suites
  - StagingDocs
- Configs
- Data (Données)

Pour ouvrir ou fermer le panneau Project (Projet), cliquez sur Project.

La barre d'outils dans le panneau Project contient des icônes pour les actions suivantes :

- Actualiser  le projet
- Ancrer ou annuler l'ancrage  du panneau Project
- Fermer  le panneau Project

Le projet inclut un répertoire **.settings,** qui ne s'affiche pas dans le panneau Project. Le répertoire **.settings** est utilisé pour enregistrer certains paramètres en interne et peut être consulté dans le système de fichiers, à partir du répertoire Projects (Projets).

Menu contextuel du panneau Project (Projet)

Vous pouvez créer différents documents dans un projet à partir du panneau Project (Projet). Lorsque vous cliquez avec le bouton droit de la souris sur un élément du panneau Project, le menu qui s'affiche est dynamique et dépend de l'élément sélectionné.

L'ordre de ces éléments varie selon l'élément sélectionné avec le bouton droit de la souris.

Les choix de menu contextuel décrits ici sont également disponibles à partir du menu principal en sélectionnant File (Fichier), New (Créer).

Create New Test Case (Créer un scénario de test)

Pour plus d'informations, consultez la section Création d'un scénario de test (page 219) dans la rubrique *Utilisation de CA Application Test.*

Create New VS Model (Créer un modèle de service virtuel)

Pour plus d'informations, reportez-vous à la section Utilisation des modèles de service virtuel dans la rubrique *Utilisation de CA Service Virtualization.* Cette fonctionnalité requiert une licence supplémentaire.

Create New VS Image (Créer une image de service virtuel)

Pour plus d'informations, consultez la section Création d'une image de service dans la rubrique *Utilisation de CA Service Virtualization.* Cette fonctionnalité requiert une licence supplémentaire.

Create New Staging Document (Créer un document de simulation)

Pour plus d'informations, consultez la section Génération de documents de simulation (page 235) dans la rubrique *Utilisation de CA Application Test.*

Create New Suite (Créer une suite)

Pour plus d'informations, consultez la section Génération de suites de tests (page 265) dans la rubrique *Utilisation de CA Application Test.*

Create New Test Audit (Créer un audit de test)

Pour plus d'informations, consultez la section Génération de documents d'audit (page 258) dans la rubrique *Utilisation de CA Application Test.*

Certains des choix suivants peuvent également être disponibles :

- Add New Folder (Ajouter un dossier)
- Import Files (Importer des fichiers)
- Rename Project (Renommer un projet)
- Supprimer
- Paste (Coller)
- View/Edit Project Metadata (Afficher/modifier les métadonnées du projet)

Si vous voulez que les documents que vous créez soient enregistrés par défaut dans le dossier dont le nom correspond, cliquez avec le bouton droit de la souris sur l'option appropriée lors de l'ajout. Si vous ajoutez un document sans cliquer avec le bouton droit de la souris sur le nom du dossier, DevTest l'ajoutera au bas de la liste de panneau Project (Projet). Pour modifier son emplacement, accédez au répertoire racine et déplacez manuellement le fichier dans le dossier approprié, puis rouvrez le projet.

Vous pouvez conserver différents types de fichier (comme .tst) sous le dossier recommandé (Tests, etc.). Le fichier peut être enregistré n'importe où dans le projet. Les seules exceptions sont les fichiers .config ; ils doivent se trouver dans les dossiers Configs (Configurations).

Projet Examples (Exemples)

Lors de la première ouverture de DevTest Workstation, le panneau Project (Projet) contient le projet nommé **Examples** (Exemples).

Ouvrez un des exemples de fichiers en double-cliquant sur celui-ci. L'éditeur approprié s'ouvre dans le panneau droit.

Les fichiers du projet actif sont disponibles dans le répertoire **LISA_HOME\examples**.

MARInfos (page 279)

AllTestsSuite

Fichier MAR de suite qui inclut la suite de tests AllTestsSuite, avec tous les fichiers .tst et les fichiers de données associés pour l'exécution de la suite. La suite inclut également le document de simulation 1User1Cycle0Think, le document d'audit DefaultAudit et le fichier de configuration project.config.

creditCheckValidate

Fichier MAR de moniteur basé sur un test qui inclut le scénario de test creditCheckValidate et le document de simulation monitorRunBase.

DatabaseModel

Fichier MAR de service virtuel qui inclut le modèle de service virtuel DatabaseModel et l'image de service virtuel.

OnlineBankingExternalCreditCheck-local

Fichier MAR de moniteur basé sur un test qui inclut le scénario de test webservices-xml-fail, le document de simulation Run1User1Cycle et le fichier project.config.

OnlineBankingJMStest-local

Fichier MAR de moniteur basé sur un test qui inclut le scénario de test async-consumer-jms, le document de simulation Run1User1Cycle et le fichier de configuration project.config.

OnlineBankingTransactionMonitor-local

Fichier MAR de moniteur basé sur un test qui inclut les éléments suivants :

- Le scénario de test à plusieurs niveaux
- Le document de simulation Run1User1Cycle
- Tous les fichiers de données pour la prise en charge du scénario de test
- Le fichier de configuration project.config

OnlineBankingWebServices-local

Fichier MAR de moniteur basé sur un test qui inclut le scénario de test webservices-xml, le document de simulation Run1User1Cycle et le fichier de configuration project.config.

rawSoap

Fichier MAR de test qui inclut le scénario de test rawSoap, le document de simulation 1User0Think_RunContinuously et le fichier de configuration project.config.

Documents d'audit (page 258)

DefaultAudit

main_all_should_fail

ws_security-xml

Configs (page 108)

Project (Projet)

Le fichier project.config contient des valeurs par défaut intelligentes pour la plupart des propriétés.

Data (Données)

Le répertoire Data (Données) contient des ensembles de données, des référentiels de clés et des fichiers WSDL requis pour l'exécution de certains exemples pour le serveur de démonstration.

Monitors (Moniteurs)

creditCheckValidate.tst

Scénario de test utilisé pour les démonstrations du moniteur de service de validation en continu. Un échec se produit de manière aléatoire pour un ID de contenu spécifique.

monitorRunBase.stg

Document de simulation spécifiant un utilisateur, un cycle, un délai de réflexion de 100 %, CAI non activé et aucun temps maximum d'exécution

monitorRunMultiple.stg

Document de simulation spécifiant un utilisateur, une exécution continue, un délai de réflexion de 136 %, CAI activé et un temps d'exécution maximum de 15 secondes

monitorSLARun.stg

Document de simulation spécifiant un utilisateur, un cycle, un délai de réflexion de 100 %, CAI non activé et aucun temps maximum d'exécution
L'enregistrement des mesures JMX et JBoss est sélectionné.

selenium.capabilities.conf

Exemple de fichier squelette contenant des paramètres de pilote Web Selenium que vous pouvez définir pour personnaliser des options de scénarios de test Selenium

serviceValidator.tst

Utilisé pour les démonstrations du moniteur de service de validation en continu. Un échec se produit de manière aléatoire pour un ID de contenu spécifique.

userAddDelete.tst

Ce test est utilisé dans l'installation du moniteur pour les démonstrations du service de validation en continu.

Setup (Configuration)

Le répertoire Setup (Configuration) situé sous le répertoire Examples (Exemples) contient des fichiers de commandes permettant de démarrer ou d'arrêter tous les composants DevTest et de charger les moniteurs du service de validation en continu.

Pour utiliser les scripts avec la liste de contrôle d'accès activée, ajoutez les options de nom d'utilisateur et de mot de passe aux commandes du gestionnaire de services dans le script. Le mot de passe n'est pas automatiquement chiffré ; veillez donc à protéger le fichier à l'aide de la méthode appropriée pour votre système d'exploitation.

Documents de simulation **(page 235)**

1User0Think_RunContinuously

Ce document de simulation exécute un utilisateur virtuel unique avec un délai de réflexion de zéro. Il exécute également le ou les tests en continu, ce qui ne signifie pas nécessairement "pour toujours".

Lorsqu'un test exécuté par ce document de simulation remplit toutes les conditions suivantes et ne dispose pas suffisamment de données, l'exécution se termine :

- Un ensemble de données est inclus dans la première étape du test.
- L'étape End of data (A la fin des données) de l'ensemble de données est End the test (Arrêter le test).
- L'ensemble de données n'est pas local, mais global.

S'il y a plusieurs ensembles de données correspondant à ces conditions, le premier ensemble de données qui expire termine l'exécution. Pour consulter un exemple significatif, reportez-vous au scénario de test à plusieurs niveaux multi-tier-combo.tst.

1user1cycle0think

Ce document de simulation exécute un test une fois avec un seul utilisateur et un pourcentage de délai de réflexion de 0 %.

ip-spoofing

Pour tester la prise en charge de l'usurpation d'adresse IP avec votre installation de DevTest, utilisez ce document de simulation. L'usurpation de l'adresse IP est activée dans ce document de simulation dans l'onglet IP Spoofing (Usurpation de l'adresse IP). Si vous exécutez le serveur d'exemples, une page Web de test d'usurpation d'adresse IP est disponible à l'adresse suivante : http://localhost:8080/ip-spoof-test.

jboss-jmx-metrics-localhost

Ce document de simulation exécute un test avec trois utilisateurs, de manière continue, avec un temps maximum d'exécution de 440 secondes et un délai de réflexion de 100 %. Il sélectionne les quatre cases à cocher correspondant aux paramètres de générateur de rapports et spécifie également toutes les mesures JMX JBoss à collecter.

Run1User1Cycle

Ce document de simulation exécute un test une fois avec un seul utilisateur et un pourcentage de délai de réflexion de 100 %.

Run1User1CyclePF

Ce document de simulation exécute un test une fois avec un seul utilisateur, un pourcentage de délai de réflexion de 100 % et CAI activé.

Run1User1CycleShowAll

Ce document de simulation exécute un test une fois avec un seul utilisateur et un pourcentage de délai de réflexion de 100 %. Il sélectionne également les quatre cases à cocher dans le générateur de rapports par défaut, de sorte que d'autres informations s'affichent dans la page d'exécution du modèle Web.

Subprocesses

ws-sec-sub

Ce scénario de test à étape unique est marqué comme sous-processus et peut être appelé à partir d'une étape Execute Sub Process (Exécuter un sous-processus).

Suites de tests (page 265)

AllTestsSuite

La suite AllTestsSuite exécute tous les tests du répertoire Tests de DevTest, à l'aide du document de simulation 1user1cycle et du document d'audit par défaut AuditDocs\DefaultAudit.aud. Elle enregistre uniquement les demandes et les réponses pour les mesures de rapport et renvoie les mesures par défaut. Son mode d'exécution est Serial (En série).

FastAllTestsSuite

La suite AllTestsSuite exécute tous les tests du répertoire Tests de DevTest, à l'aide du document de simulation 1user1cycle0think et du document d'audit par défaut AuditDocs\DefaultAudit.aud. Elle enregistre uniquement les demandes et les réponses pour les mesures de rapport et renvoie les mesures par défaut. Son mode d'exécution est Parallel (Parallèle).

Scénarios de test (page 79)

AccountControlMDB

Test JMS simple qui ajoute un utilisateur avec un compte à LISA Bank. Les modèles attendus dans les réponses sont utilisés dans des assertions à partir des deux étapes.

async-consumer-jms

Il s'agit d'un exemple de file d'attente de consommateur asynchrone dans laquelle le scénario de test accepte des messages en continu à partir d'une rubrique ou d'une file d'attente de réponses. La file d'attente met également les messages à la disposition du scénario de test dans l'ordre d'arrivée des messages.

La première étape crée la file d'attente (interne au test).

La deuxième étape envoie trois messages à une file d'attente JMS sur le serveur de démonstration. La file d'attente asynchrone reçoit les messages.

La troisième étape valide la réception des trois messages dans la file d'attente asynchrone.

ejb3EJBTest

Test EJB exclusif de la fonctionnalité LISA Bank. En général, vous testez les applications en enregistrant un navigateur Web ou une autre interface utilisateur. Ces tests sont des tests d'intégration de bout en bout. Ils ne sont pas les plus accessibles et requièrent l'implication de rédacteurs techniques supplémentaires, même si vous ne devez toutefois écrire encore aucun code.

Ces tests sont utiles à l'équipe de développement, car ils permettent de constamment tester et valider le code sans recourir à une interface utilisateur, qui peut ne pas exister ou être modifiée continuellement, si bien que les tests ne peuvent pas suivre.

ejb3WSTest

Ce modèle teste consciencieusement les services Web de LISA Bank. Il est presque identique au test ejb3EJB en termes de fonctionnalité et d'utilité. Pour plus d'informations, reportez-vous à la documentation de ce scénario de test.

ip-spoofing

Cet exemple de scénario de test illustre la prise en charge de l'usurpation d'adresse IP dans DevTest.

Ce test demande l'URL http://localhost:8080/ip-spoof-test à l'aide d'une étape REST, une page Web qui contient l'adresse IP du client demandeur. Le test envoie alors une demande SOAP à l'URL suivante, qui identifie un service Web contenant une opération qui renvoie l'adresse IP du client demandeur :

```
http://localhost:8080/itko-examples/ip-spoof-test/webservic
e
```

Le scénario de test exécute les deux demandes dans une boucle dix fois. Vous pouvez simuler ce test avec le document de simulation de test d'usurpation de l'adresse IP ip-spoofing.stg. La configuration d'interface réseau correcte vous permet de consulter les différentes adresses IP utilisées par les utilisateurs virtuels lorsqu'ils effectuent des demandes HTTP et SOAP.

JMS

Ce scénario de test est un exemple JMS simple indiquant la procédure d'envoi de messages de texte ou XML, et d'objets au format Java natif.

Lisa_config_info

Ce scénario de test récupère les informations de diagnostic de l'ordinateur exécutant DevTest. Les résultats peuvent permettre au support technique de résoudre les problèmes de configuration.

load-csv-dataset-web

Ce modèle de test utilise un fichier CSV de valeurs séparées par des virgules comme ensemble de données pour tester une application Web. L'application Web de démonstration fournie avec DevTest permet d'ajouter des utilisateurs à la base de données et de les supprimer.

log-watcher

Cet exemple indique la procédure à suivre pour mettre en échec un test en recherchant les messages ERROR ou WARNING dans un fichier journal.

L'exemple utilise un ensemble de données pour fournir deux nombres à diviser à l'exemple de bean AntiPattern. A la moitié de l'ensemble de données environ, l'opérande 0 est défini. Cet opérande entraîne une exception de division par zéro sur le serveur. Le bean AntiPattern enregistre l'exception dans le journal et renvoie la valeur -1 comme résultat.

Cet exemple est un anti-modèle commun : des erreurs internes se produisent, mais les intervenants extérieurs ignorent que le résultat est incorrect. Le résultat paraît plausible, mais il est erroné. Le résultat attendu est la propagation en retour de l'exception par le composant EJB vers l'appelant.

Ce scénario de test échoue avec CAI, car l'agent enregistre le fait qu'une exception a été renvoyée, ce qui permet à DevTest de déterminer qu'une erreur s'est produite.

Une alternative à l'utilisation de CAI consiste à configurer une assertion globale pour surveiller le fichier journal du serveur. Définissez les éléments compris dans une expression régulière. Dans le cas présent, il s'agit simplement de l'erreur de test ERROR. Vous pouvez choisir le niveau de complexité des expressions régulières qui vous convient le mieux.

En général, vous définissez l'assertion de sorte à mettre le test en échec immédiatement. Dans ce cas, passez à l'étape Error detected in log (Erreur détectée dans le fichier journal) et terminez le test normalement.

DevTest attend que les applications en cours de test qui journalisent des erreurs ou des avertissements ne soient jamais transmises. Pensez à utiliser un compagnon équivalent dans vos scénarios de test par défaut.

main_all_should_fail

Ce test est un exemple de test négatif. Un échec des étapes du test doit se produire, c'est-à-dire, les données de service fournies doivent entraîner une erreur.

Le test comprend le compagnon NegativeTestingCompanion, qui met le test en échec si une étape réussit.

Dans ce scénario, vous tentez de créer des utilisateurs dans le serveur de démonstration qui existent déjà. Les données sont récupérées à partir de la base de données, grâce à l'ensemble de données de nom d'utilisateur qui interroge la table directement. Si une étape réussit, le test complet échoue.

La seule étape qui réussit est l'étape de "réussite silencieuse". Cette étape vous permet de marquer les étapes dont vous attendez la transmission dans ce type de scénario comme Quiet (silencieuses) dans l'éditeur, de sorte qu'elles ne soient pas traitées par le NegativeTestingCompanion (Compagnon de test négatif).

main_all_should_pass

Ce test appelle un sous-processus permettant d'insérer un nom d'utilisateur unique dans la table USERS du serveur de démonstration.

L'ensemble de données est intéressant, car il compte sur une feuille de données pour extraire des valeurs à partir d'un générateur de code unique. La même opération peut être effectuée à l'aide d'un générateur de code unique comprenant un ensemble de données de compteur. Cet exemple illustre l'influence que peut exercer un ensemble de données sur un autre.

Les ensembles de données sont évalués dans l'ordre dans lequel ils sont spécifiés. A chaque exécution de l'étape, une nouvelle valeur est affectée à la propriété UniqueUser. La feuille de données mentionne {{UniqueUser}} quatre fois. Cinq valeurs uniques sont donc obtenues.

Si l'une des étapes échoue, le test échoue immédiatement.

Comparez ce test au test similaire main_all_should_fail, dans lequel un échec de chaque étape est attendu et qui est mis en échec si une étape est réussie. Ce processus utilise la méthode du test négatif.

multi-tier-combo

Le test multi-tier-combo (à plusieurs niveaux) utilise différents terminaux de service pour valider l'exemple LisaBank. Testez des transactions SOAP, EJB, JMS, Selenium et Web et validez-les de différentes manières, notamment par validation directe de la base de données du serveur de démonstration. Ce test requiert le navigateur Firefox pour l'exécution des étapes Selenium.

Le test à plusieurs niveaux utilise différents terminaux de service pour valider l'exemple LISA Bank. Le scénario de test teste les transactions Web, SOAP, EJB et JMS, puis les valide de plusieurs façons, y compris via la validation directe de la base de données du serveur de démonstration.

Le test indique également la procédure de création d'objets SOAP complexes à partir de feuilles de calcul. La feuille de calcul **multi-tier-users.xls** du dossier Data (Données) du projet sauvegarde l'ensemble de données de l'**utilisateur** lors de la première étape.

Si vous exécutez ce test dans la fenêtre Interactive Test Run (Exécuter un test interactif), un utilisateur unique est créé à partir de la première ligne de la feuille de calcul, puis le test se termine.

Si vous simulez le test avec l'exemple de document de simulation **1User0Think_RunContinuously**, il redémarrera jusqu'à atteindre la fin de l'ensemble de données. Ce processus est la méthode préférée pour une itération répétée sur un grand ensemble de données. Une boucle pourrait également être utilisée dans le scénario de test, mais elle ne fournirait pas autant de flexibilité.

Si vous permettez au document de simulation qui contrôle l'ensemble de données de terminer le test, vous pouvez diffuser le test auprès de plusieurs utilisateurs virtuels. Vous pouvez également contrôler le rythme du test à l'aide de délais de réflexion et d'autres paramètres.

Seuls les ensembles de données globaux définis lors de la première étape du test affectent le comportement de fin du test en continu du document de simulation. Si l'ensemble de données est local ou déclaré à un autre emplacement dans le test, l'exécution continue implique une exécution permanente réelle.

multi-tier-combo-se

Le test multi-tier-combo-se (à plusieurs niveaux) utilise différents terminaux de service pour valider l'exemple LISA Bank. Testez des transactions SOAP, EJB, JMS et Web et validez-les de différentes manières, notamment par validation directe de la base de données du serveur de démonstration.

Le test indique également la procédure de création d'objets SOAP complexes à partir de feuilles de calcul. L'ensemble de données de l'**utilisateur** de la première étape est confirmé par la feuille de calcul **multi-tier-users.xls** du dossier **Data** (Données) du projet.

Si vous exécutez ce test dans la fenêtre Interactive Test Run (Exécution d'un test interactif), un utilisateur unique est créé à partir de la première ligne de la feuille de calcul, puis le test se termine.

Si vous simulez le test avec l'exemple de document de simulation **1User0Think_RunContinuously**, il redémarrera jusqu'à atteindre la fin de l'ensemble de données. Ce processus est la méthode recommandée pour une itération répétée sur un ensemble de données volumineux. Vous pouvez introduire une boucle dans le scénario de test, mais cette méthode est loin d'être flexible.

Si vous laissez le document de simulation contrôler l'ensemble de données à la fin du test, vous pourrez répartir le test parmi plusieurs utilisateurs virtuels, ou contrôler le rythme du test à l'aide de délais de réflexion.

Notez que le comportement impliquant la fin de l'exécution continue du test du document de simulation est uniquement affecté par des ensembles de données globaux définis dans la première étape du test. Si l'ensemble de données est local ou déclaré à un autre emplacement dans le test, l'exécution continue implique une exécution permanente réelle.

rawSoap

L'étape rawSoap est un scénario de test en une étape unique présentant une demande SOAP brute simple dans l'étape listUsers.

rest-example

Le test rest-example présente la procédure d'exécution des services RESTful. Le serveur de démonstration contient un exemple de JAX-RS. Chaque étape du test indique la procédure à suivre pour interagir avec ce service via XML et JSON.

Génération de scripts

CA Application Test peut utiliser des moteurs de génération de scripts JSR-223, ce qui vous permet d'utiliser de nombreux moteurs de génération de scripts dans des étapes de script, des assertions, des gestionnaires de protocole de données, des scripts de correspondance et pratiquement à tout emplacement à l'aide de la syntaxe {{=%language%}}.

sendJMSrr

Permet de tester la capacité de VSEasy à créer un service JMS à partir de paires demande-réponse.

service-validation

Ce test est un exemple simple de validation de service. Le test appelle un service Web et un service EJB, puis examine la base de données sous-jacente à l'aide de SQL pour valider le fonctionnement approprié des services.

web-application

Ce test est un test Web simple généré à l'aide de l'enregistreur Web. Il contient certaines assertions de base comme l'assertion assert nonempty response (assertion d'une réponse non vide), qui est automatiquement générée. Le test contient également des assertions assert title (assertion de titre) créées lors de l'analyse des réponses HTML pour la balise <title>. Ces assertions permettent de vérifier que la page enregistrée est identique à celle utilisée lors de la réutilisation du test.

webservices

Ce scénario de test permet d'ajouter, de récupérer et de supprimer un utilisateur de services Web EJB3. Le test utilise un générateur de code unique pour créer un nombre comprenant un préfixe qui est une valeur {{user}} provenant du fichier de configuration. Le mot de passe est codé de manière irréversible dans le fichier de configuration.

ws_attachments

Ce scénario de test teste la capacité d'envoi et de réception de BLOB de données intégrés codés en base64 et de pièces jointes XOP/MTOM. Les filtres et les assertions dans les étapes vérifient que les demandes et les réponses soient correctes.

ws_security

Le scénario de test ws_security indique la procédure à suivre pour utiliser des messages SOAP signés et chiffrés. Les deux premières étapes réussissent et les deux dernières échouent. Les appels sont les mêmes, mais le service Web n'accepte pas les messages non chiffrés ou non signés.

ws_security-xml

Ce modèle de test indique la procédure à suivre pour utiliser des messages SOAP signés et chiffrés. Les deux premières étapes doivent se terminer correctement et les deux dernières doivent échouer. Les appels sont identiques, mais le service Web n'accepte pas les messages non chiffrés ou signés.

Ce plan de test requiert la prise en charge par votre environnement Java du niveau de chiffrement illimité. Si l'une des deux premières étapes échoue, il est probable que vos fichiers JAR de JCE doivent être mis à jour pour activer le niveau de chiffrement illimité. Vous pouvez télécharger les fichiers JAR de JCE à partir du site suivant : www.oracle.com. Recherchez les mots clés JCE unlimited strength (Niveau illimité JCE) et sélectionnez la bibliothèque de JCE correspondant à votre version Java, par exemple JCE 7 pour la version Java 7. Une fois les fichiers JAR de JCE installés, vous devez redémarrer les services DevTest.

Dans le document d'audit, deux occurrences Step Error (Erreur d'étape) sont attendues. Procédez à l'audit des deux erreurs d'étape, car les deux dernières étapes doivent déclencher des erreurs. De cette manière, le document d'audit peut également procéder à l'audit du nombre d'occurrences reçues pour un événement. Si vous ajoutez une troisième entrée Step Error (Erreur d'étape) au document d'audit, l'auditeur de ce test échouera, car deux événements d'erreur d'étape seulement sont déclenchés.

Echecs de tests

webservices-xml-fail

Ce scénario de test permet d'ajouter, de récupérer et de supprimer un utilisateur de services Web EJB3. Le test utilise un générateur de code unique pour créer un nombre comprenant un préfixe qui est une valeur {{user}} provenant du fichier de configuration. Le mot de passe est codé de manière irréversible dans le fichier de configuration.

VServices

statefullATM.tst

statelessATM.tst

web-app-proxy.tst

DatabaseModel.vsm

kioskV4model.vsm

kioskV5.vsm

kioskV6.vsm

WebServicesModel.vsm

webservices-vs.vsm

Images

DatabaseModel.vsi

kioskV4ServiceImage.vsi

kioskV6.vsi

si-kioskV5-dynamic.vsi

si-kioskV5.vsi

WebServicesModel.vsm

Création d'un projet

Vous pouvez créer un projet depuis le début ou à partir d'un fichier MAR (Model Archive, archive de modèle) existant.

Remarque : Vous pouvez uniquement créer un projet à partir d'un fichier MAR de VSE. Vous ne pouvez pas ajouter un fichier MAR à un projet existant à partir de cette boîte de dialogue.

Procédez comme suit:

1. Dans le menu principal, sélectionnez File (Fichier), New (Créer), Project (Projet).

   La boîte dialogue Create New DevTest Project (Créer un projet DevTest) s'affiche.
2. Dans le champ Project Name (Nom du projet), entrez le nom du projet.
3. Pour créer le projet à un emplacement autre que le répertoire **LISA_HOME\Projects**, cliquez sur l'icône près du champ Project Name (nom du projet) et spécifiez l'emplacement.
4. Pour créer le projet à partir d'un fichier MAR existant, procédez comme suit :
   a. Cochez la case Base the new project on one of the following (Baser le nouveau projet sur l'un des éléments suivants).
   b. Sélectionnez l'option MAR File (Fichier MAR).
   c. Spécifiez le nom de fichier MAR et l'emplacement de répertoire.
5. Cliquez sur Create (Créer).

Ouverture d'un projet

Vous pouvez ouvrir un projet à partir de DevTest Workstation ou de la ligne de commande.

Pour ouvrir un projet à partir de DevTest Workstation :

1. Dans le menu principal, sélectionnez File (Fichier), Open (Ouvrir), Project (Projet).

   Si vous avez déjà ouvert plusieurs projets, vous pouvez sélectionner un projet dans la liste des projets récemment ouverts qui s'affiche. Une fois que vous avez sélectionné un projet, il s'ouvrira.

2. Pour rechercher un fichier de projet, sélectionnez File System (Système de fichiers) dans la liste de choix et la fenêtre Open Project (Ouvrir un projet) s'ouvre.

3. Sélectionnez le projet et cliquez sur Open (Ouvrir).

Pour ouvrir un projet à partir de la ligne de commande :

1. Accédez au répertoire **LISA_HOME\bin**.

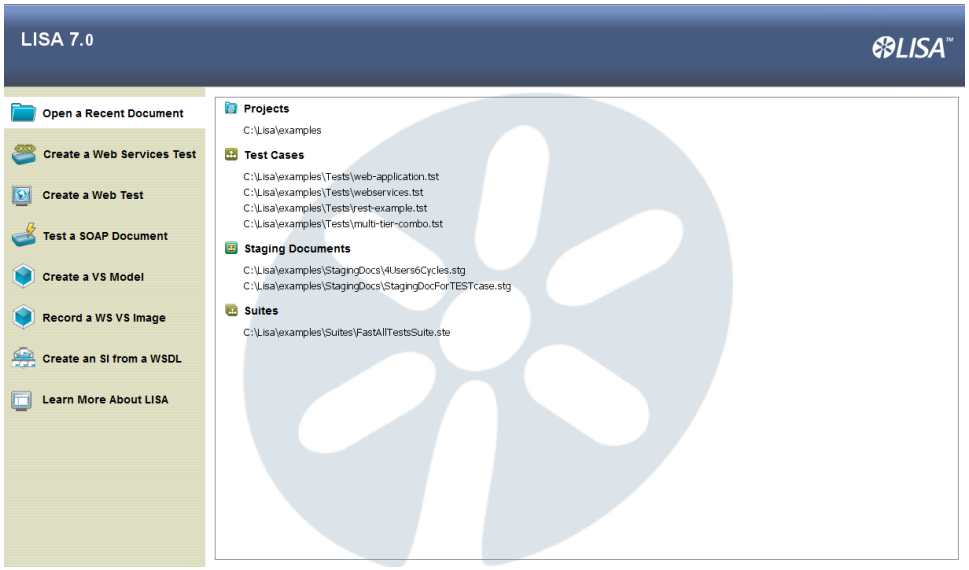
2. Exécutez le fichier exécutable de DevTest Workstation et spécifiez le répertoire de projet en tant qu'argument. Le répertoire de projet peut être un chemin absolu ou un chemin d'accès relatif. L'exemple suivant utilise un chemin absolu :

   ```
   LISAWorkstation "C:\Program Files\CA\Lisa\Projects\Project1"
   ```

Fenêtre Quick Start (Démarrage rapide)

La fenêtre Quick Start (Démarrage rapide) est la première fenêtre qui s'affiche lorsque vous ouvrez DevTest Workstation.

Remarque : La fenêtre Quick Start est toujours disponible sous la forme d'un onglet une fois que plusieurs onglets sont ouverts.



La fenêtre Quick Start (Démarrage rapide) contient certaines des options les plus utiles présentes dans DevTest Workstation. Lorsque vous cliquez sur une option, ses paramètres sont répertoriés sur le côté droit de la fenêtre.

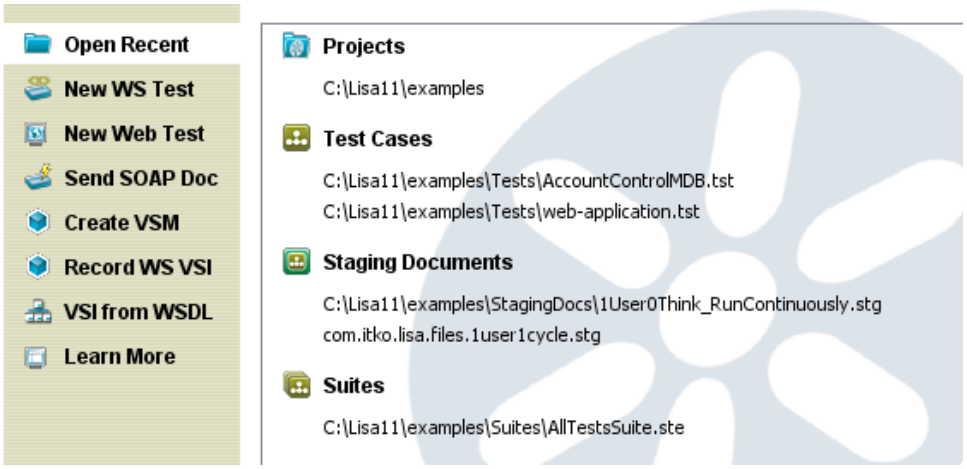
Les choix suivants sont disponibles dans cette fenêtre. Selon votre résolution d'écran, vous pouvez afficher la formulation compacte ou étendue pour chaque option du menu. Pour afficher toutes les options du menu, cliquez sur la flèche vers le bas dans la partie inférieure de la fenêtre.

- Open Recent ou Open a Recent Document (Ouvrir un document récent)
- New WS Test (Nouveau test de service Web) ou Create a Web Services Test (Créer un test de service Web)
- New Web Test (Nouveau test Web) ou Create a Web Test (Créer un test Web)
- Send SOAP Doc (Envoyer un document SOAP) ou Test a SOAP Document (Tester un document SOAP)
- Create VSM ou Create a VS Model (Créer un modèle de service virtuel)
- Record WS VSI ou Record a WS VS Image (Enregistrer une image de service virtuel de service Web)
- VSI from WSDL (Image de service virtuel à partir d'un fichier WSDL) ou Create an SI from a WSDL (Créer une image de service à partir d'un fichier WSDL)

- Learn More (En savoir plus) ou Learn More About DevTest (En savoir plus sur DevTest)

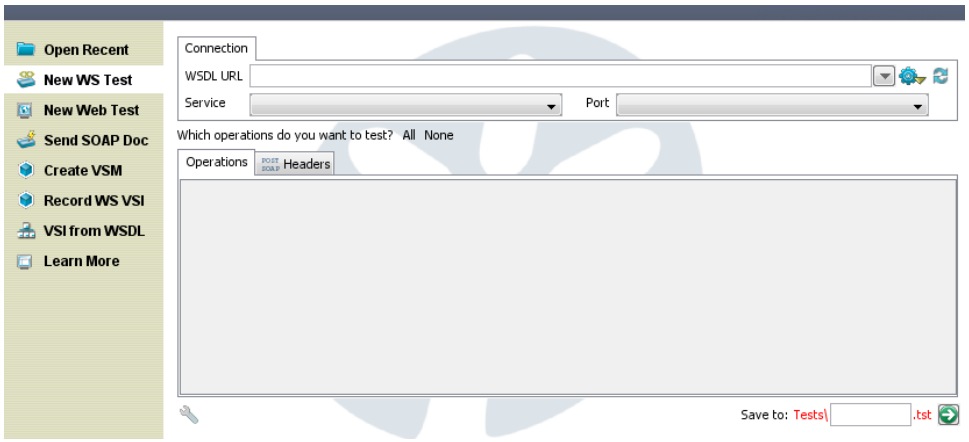
Open Recent (Ouvrir un document récent)

Lorsque DevTest Workstation s'ouvre, cette option est sélectionnée par défaut. Le volet droit affiche les projets, les scénarios de test, les suites, les documents de simulation, les modèles de service virtuel ou les images de service virtuel récemment ouverts, le cas échéant.



New WS Test (Nouveau test de service Web)

L'option New WS Test (Nouveau test de service Web) de la fenêtre Quick Start (Démarrage rapide) vous permet de créer un scénario de test de service Web. Les paramètres requis sont affichés dans le volet droit.



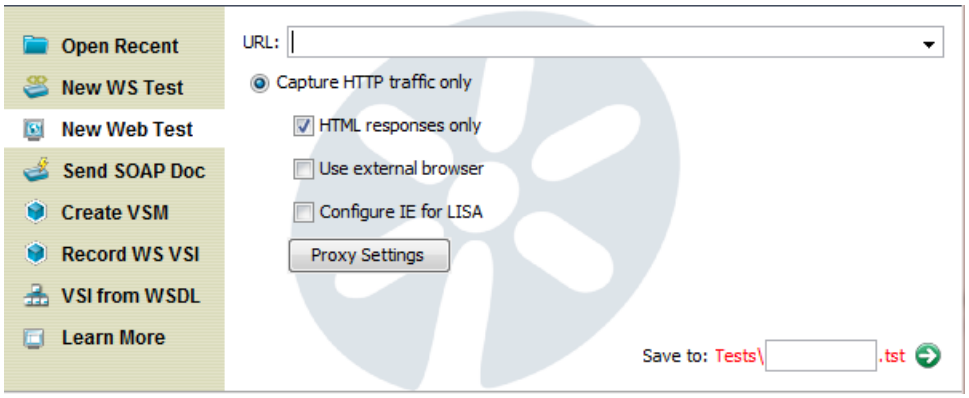
Procédez comme suit:

1. Entrez des valeurs dans les champs WSDL URL (URL du document WSDL), Service et Port.
2. Sélectionnez les opérations que vous voulez tester à partir de All (Tout) ou None (Aucun), ou sélectionnez chaque élément séparément.
3. Dans le volet droit, entrez le nom du test et sélectionnez l'emplacement sous lequel vous souhaitez l'enregistrer dans le dossier de projet. Lorsque vous sélectionnez l'emplacement, il s'affiche dans le champ Path (Chemin). Dans ce volet, vous pouvez cliquer avec le bouton droit de la souris sur un dossier, en créer un, renommer ou supprimer un dossier existant.
4. Pour créer le scénario de test, cliquez sur la flèche verte  .

Pour plus d'informations, consultez la section Génération d'un service Web (page 445) de la rubrique *Utilisation de CA Application Test.*

New Web Test (Nouveau test Web)

L'option New Web Test (Nouveau test Web) de la fenêtre Quick Start (Démarrage rapide) vous permet de créer un test Web.



Procédez comme suit:

1. Entrez l'URL pour le test Web.
2. Sélectionnez l'option Capture HTTP traffic only (Capturer le trafic HTTP uniquement).
3. Sélectionnez des options en cochant les cases correspondantes :

   HTML Responses Only (Réponses HTML uniquement)

   Permet de capturer uniquement les réponses HTML.

   Use External Browser (Utiliser un navigateur externe)

   Permet d'ouvrir une fenêtre de navigateur externe.

   Configure IE for DevTest (Configurer Internet Explorer pour DevTest)

   Permet de configurer Internet Explorer pour DevTest.
4. Dans le volet droit, entrez le nom du test et sélectionnez l'emplacement sous lequel vous souhaitez l'enregistrer dans le dossier de projet. Lorsque vous sélectionnez l'emplacement, il s'affiche dans le champ Path (Chemin). Dans ce volet, vous pouvez cliquer avec le bouton droit de la souris sur un dossier et créer, renommer ou supprimer un dossier.
5. Pour créer le scénario de test, cliquez sur la flèche verte  .

Pour plus d'informations, consultez la section Enregistrement d'un service Web (page 447) de la rubrique *Utilisation de CA Application Test.*

Send SOAP Doc (Envoyer un document SOAP)

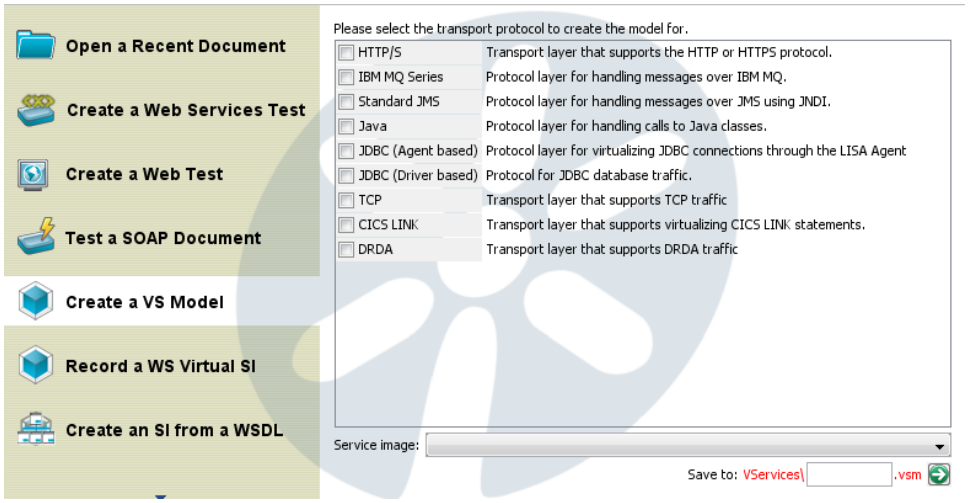
L'option Send SOAP doc (Envoyer un document SOAP) de la fenêtre Quick Start (Démarrage rapide) vous permet de créer un scénario de test de demande SOAP.

Procédez comme suit:

1. Entrez l'URL de serveur SOAP et l'action SOAP.
2. Entrez l'URL du terminal de service Web dans le champ SOAP Server URL (URL du serveur SOAP).
3. Dans le champ SOAP Action (Action SOAP), entrez l'action SOAP comme indiquée dans la balise <soap: operation> dans le document WSDL pour la méthode appelée. Cette valeur est requise pour SOAP 1.1 et doit être souvent laissée vide pour SOAP 1.2.
4. Entrez ou collez la demande SOAP dans l'éditeur.
5. Entrez le nom du test dans le champ Save To (Enregistrer dans).
6. Pour créer le scénario de test, cliquez sur la flèche verte  .

Create VSM (Créer un modèle de service virtuel)

L'option Create VSM (Créer un modèle de service virtuel) de la fenêtre Quick Start (Démarrage rapide) vous permet de créer un modèle de service virtuel et l'image de service virtuel correspondante.



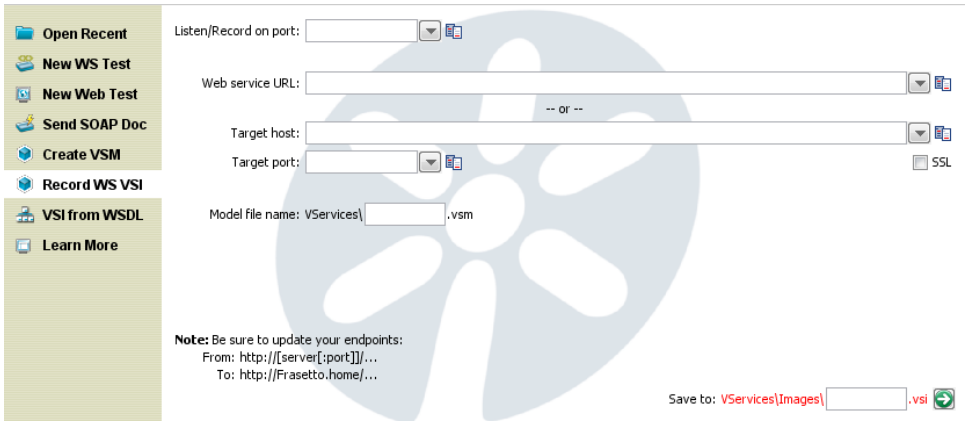
Procédez comme suit:

1. Sélectionnez le protocole de transport dans la liste des protocoles disponibles. L'option sélectionnée déterminera la séquence de fenêtres suivantes.
2. Entrez le nom de l'image de service.
3. Entrez le nom du modèle de service virtuel et sélectionnez l'emplacement sous lequel vous souhaitez l'enregistrer dans le dossier de projet. Lorsque vous sélectionnez l'emplacement, il s'affiche dans le champ Path (Chemin).
4. Pour créer le scénario de test, cliquez sur la flèche verte .

Pour des informations détaillées sur les fenêtres et les options de création de modèle de service virtuel et d'image de service virtuel, consultez la rubrique Création d'une image de service.

Record WS VSI (Enregistrer une image de service virtuel de service Web)

L'option Record WS VSI (Enregistrer une image de service virtuel de service Web) de la fenêtre Quick Start (Démarrage rapide) vous permet de créer une image de service virtuel des services Web. Les paramètres requis sont affichés dans le volet droit.

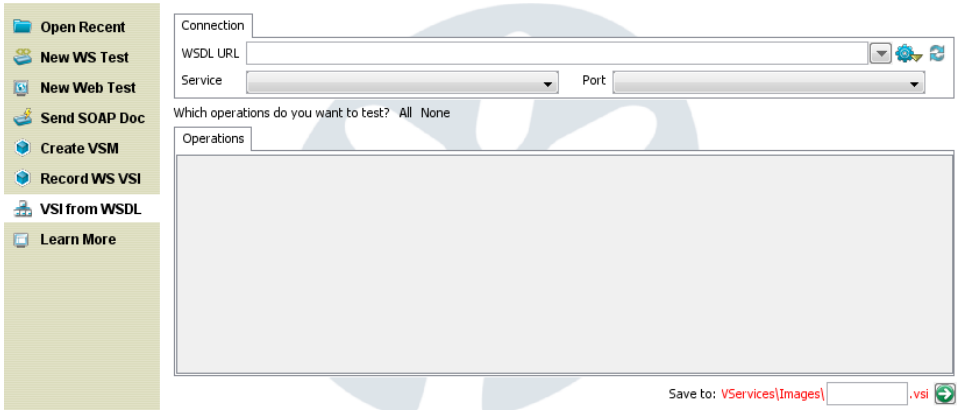


Procédez comme suit:

1. Entrez un port sur lequel effectuer les écoutes et l'enregistrement.
2. Entrez l'URL du service Web à enregistrer.
3. Entrez l'hôte cible et le port à enregistrer.
4. Pour indiquer qu'une connexion SSL doit être utilisée avec cette image de service, utilisez la case à cocher SSL.
5. Entrez le nom du modèle de service virtuel et de l'image de service virtuel.
6. Pour créer le scénario de test, cliquez sur la flèche verte .

VSI from WSDL (Image de service virtuel à partir d'un fichier WSDL)

L'option VSI from WSDL (Image de service virtuel à partir d'un fichier WSDL) de la fenêtre Quick Start (Démarrage rapide) vous permet de créer une image de service virtuel à partir d'un fichier WSDL. Les propriétés requises sont affichées dans le volet droit.



Procédez comme suit:

1. Entrez des valeurs dans les champs WSDL URL (URL du document WSDL), Service et Port. Si l'URL du document WSDL que vous saisissez ne comprend aucune propriété dans son chemin, les champs Service et Port seront remplis automatiquement.
2. Pour sélectionner les opérations à tester, cliquez sur All (Tout) ou None (Aucun), ou sélectionnez chaque élément séparément.
3. Saisissez le nom de l'image de service et sélectionnez le chemin dans le dossier de projet dans lequel elle sera enregistrée.
4. Pour créer le scénario de test, cliquez sur la flèche verte  .

En savoir plus

L'option Learn More (En savoir plus) de la fenêtre Quick Start (Démarrage rapide) affiche un lien vers la documentation, le support en ligne et la communauté globale DevTest.

L'ensemble de la documentation de l'utilisateur disponible pour DevTest est accessible à partir de ce menu.

Compteur de mémoire de DevTest Workstation

Dans le coin droit inférieur de la fenêtre principale de DevTest Workstation, le compteur de mémoire de DevTest Workstation affiche les informations sur l'utilisation et la disponibilité de la mémoire d'exécution.

- Pour exécuter le nettoyage de la mémoire, cliquez sur Memory Meter (Compteur de mémoire).
- Pour générer un fichier de vidage de segment de mémoire (HProf), cliquez avec le bouton droit de la souris sur Memory Meter (Compteur de mémoire). Sélectionnez l'option Dump Heap (Vider le segment de mémoire).

  Remarque : Cette fonctionnalité est un outil de diagnostic permettant au support de CA de déterminer les causes des conditions OutOfMemory. Le segment de mémoire est automatiquement vidé si une condition OutOfMemory se produit ; cette option permet de déclencher manuellement le vidage du segment de mémoire.

  Une fois que l'image mémoire est créée, un message indique que le segment de mémoire a été vidé.

Panneaux de barre d'état

DevTest Workstation comprend des panneaux de barre d'état pour certaines fonctionnalités, comme l'étape Output Log Message (Message de journal de sortie).

Vous pouvez contrôler l'animation de l'ouverture et de la fermeture des panneaux de barre d'état. Par défaut, l'animation est désactivée dans le but d'améliorer les performances pour les utilisateurs qui accèdent à DevTest Workstation à partir d'un ordinateur de bureau distant.

Pour activer l'animation, ajoutez la ligne suivante au fichier **site.properties** ou **local.properties** :

```
lisa.ui.tray.animation=true
```

Console DevTest

La console Web DevTest permet d'accéder aux consoles et aux tableaux de bord suivants :

- Reporting Dashboard (Tableau de bord de génération de rapports)
- Continuous Validation Service (Service de validation en continu)
- Server Console (Console de serveur)
- VSEasy

Reporting Dashboard (Tableau de bord de génération de rapports)

Une visionneuse de rapports affiche les informations d'événement et de mesures, ainsi que des informations dérivées des données capturées pendant l'exécution des tests. Vous pouvez utiliser un document de simulation pour définir les événements et les mesures à capturer à des fins de génération de rapports.

Pour plus d'informations, consultez la section Rapports (page 403) de la rubrique *Utilisation de CA Application Test*.

Continuous Validation Service (Service de validation en continu)

Le service Continuous Validation Service (Service de validation en continu) vous permet de planifier des tests et des suites de tests à exécuter régulièrement sur une période étendue.

Pour plus d'informations, consultez la section Service de validation en continu (page 385) dans la rubrique *Utilisation de CA Application Test.*

Server Console (Console de serveur)

La console Server Console (Console de serveur) vous permet de gérer des laboratoires et de configurer le contrôle d'accès basé sur les rôles. Cette console vous permet également d'accéder au tableau de bord VSE Dashboard.

VSEasy

VSEasy vous permet de créer des images de service HTTP et des images de service à partir de fichiers VRS. Vous pouvez utiliser VSEasy pour créer des services virtuels HTTP sans état avec des charges utiles SOAP, JSON, XML ou HTML.

Pour plus d'informations, consultez les sections Laboratoires DevTest Cloud (page 365), Liste de contrôle d'accès, VSE Dashboard (Tableau de bord) et Création et déploiement d'un service virtuel à l'aide de VSEasy dans les rubriques *Utilisation de CA Application Test, Administration et Utilisation de CA Service Virtualization*.

Ouverture de la console DevTest Console

Vous pouvez ouvrir la DevTest Console à partir du menu principal de DevTest Workstation ou d'un navigateur Web.

La page d'accueil de la console DevTest permet d'accéder au Reporting Dashboard (Tableau de bord de génération de rapports), au CVS Dashboard (Tableau de bord du service CVS), à la Server Console (Console de serveur) et au VSEasy. La page d'accueil inclut également un lien vers la partie CAI du portail DevTest. La partie inférieure droite de la page d'accueil affiche le numéro de version.

Pour ouvrir la console DevTest à partir du menu principal de DevTest Workstation :

1. Dans le menu principal, sélectionnez View (Afficher), DevTest Portal (Portail DevTest).

   Une boîte de dialogue de connexion s'affiche.

2. Saisissez votre nom d'utilisateur et votre mot de passe, puis cliquez sur Login (Connexion).

Pour ouvrir la console DevTest à partir d'un navigateur Web :

1. Vérifiez que le registre (page 11) est en cours d'exécution.
2. Dans un navigateur Web, entrez **http://localhost:1505/**. Si le registre se trouve sur un ordinateur distant, remplacez **localhost** par le nom ou l'adresse IP de l'ordinateur.

   Une boîte de dialogue de connexion s'affiche.

3. Saisissez votre nom d'utilisateur et votre mot de passe, puis cliquez sur Login (Connexion).

Délais d'expiration du serveur Web

Par défaut, le serveur Web que la console DevTest utilise attend l'exécution d'un processus sur le serveur pendant 90 secondes. Si la durée de l'exécution d'un processus excède 90 secondes, la connexion sera interrompue et l'application cliente ou le navigateur gèrera cette interruption de manière appropriée.

Vous pouvez modifier la valeur du délai d'expiration par défaut en ajoutant la propriété **lisa.webserver.socket.timeout** dans le fichier **local.properties.** La valeur est en millisecondes. Par exemple :

```
lisa.webserver.socket.timeout=120000
```

Chapitre 10: Génération de scénarios de test

Pour générer des scénarios de test, vous devez savoir définir des propriétés, utiliser des configurations et les appliquer à votre projet, appliquer des filtres, ajouter des assertions, des ensembles de données et des compagnons. Cette section traite également de l'éditeur Complex Object Editor (Editeur d'objets complexes).

Ce chapitre traite des sujets suivants :

Anatomie d'un scénario de test

Un scénario de test regroupe les spécifications qui permettent de mener à bien un test sur un composant professionnel dans le système testé. Un scénario de test est stocké sous la forme d'un document XML, qui contient toutes les informations nécessaires au test du composant ou du système spécifié.

Un scénario de test est un flux de travaux rassemblant des étapes de test connectées par des chemins qui représentent la réussite ou l'échec des étapes. Des assertions peuvent accompagner une étape et différents chemins sont fournis en fonction du déclenchement des assertions.

Remarque : Enregistrez vos scénarios de test régulièrement.

Les rubriques suivantes sont incluses.

Démarrage rapide du scénario de test

Pour commencer à travailler avec des scénarios de test :

1. Démarrez le registre.

   Reportez-vous à la section Registre DevTest (page 11) de la rubrique *Utilisation de CA Application Test.*

2. Créez ou ouvrez un projet dans DevTest Workstation.

   Reportez-vous à la section Panneau Project (Projet) (page 50) de la rubrique *Utilisation de CA Application Test.*

3. Créez un scénario de test dans le projet.

   Reportez-vous à la section Création d'un scénario de test (page 219) de la rubrique *Utilisation de CA Application Test.*

   Vous pouvez ouvrir un scénario de test à partir d'un projet ou en dehors d'un projet en sélectionnant File (Fichier), Open (Ouvrir), Test Case (Scénario de test) à partir du menu principal.

Le scénario de test **multi-tier-combo.tst** est illustré dans les graphiques suivants. Pour plus d'informations sur le scénario de test à plusieurs niveaux, consultez la section Scénario de test à plusieurs niveaux (page 84) dans la rubrique *Utilisation de CA Application Test.*

DevTest Workstation est divisé en trois zones principales (de gauche à droite) :

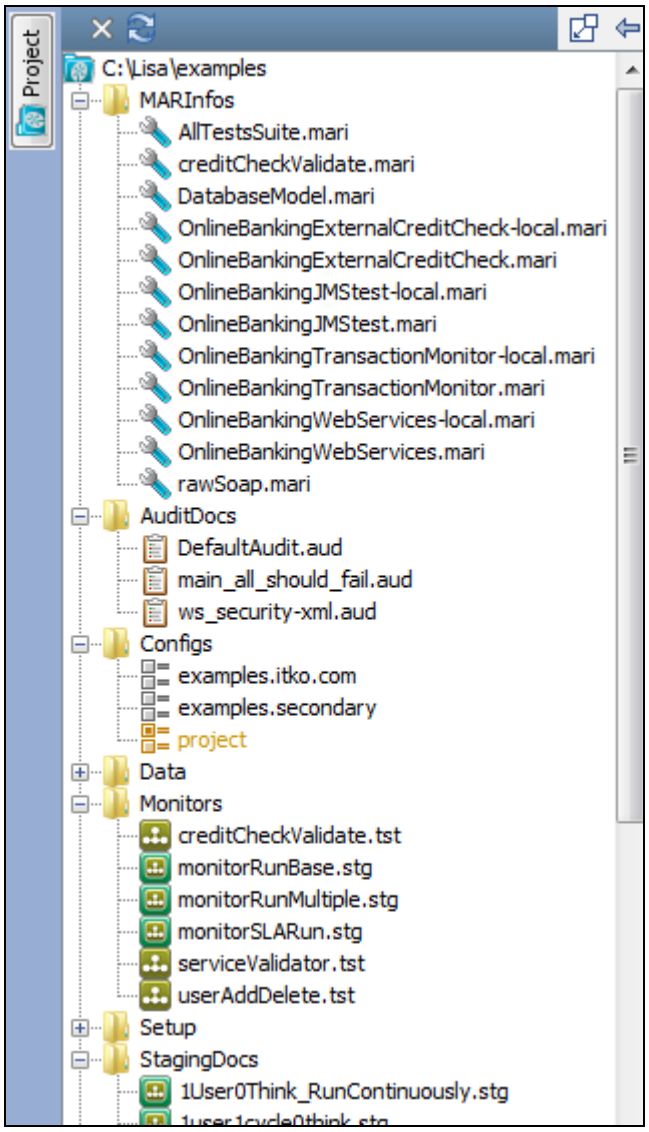
- Panneau Project (Projet)
- Model editor (Editeur de modèles)
- Panneau Element

Panneau Project (Projet)

Le panneau Project (Projet) situé dans la partie gauche de la fenêtre, est ancrable. Vous pouvez ouvrir ou fermer le panneau Project en cliquant sur 

Project sur la gauche.

Lors de l'ouverture de DevTest Workstation, le dernier projet que vous avez ouvert s'ouvre par défaut.

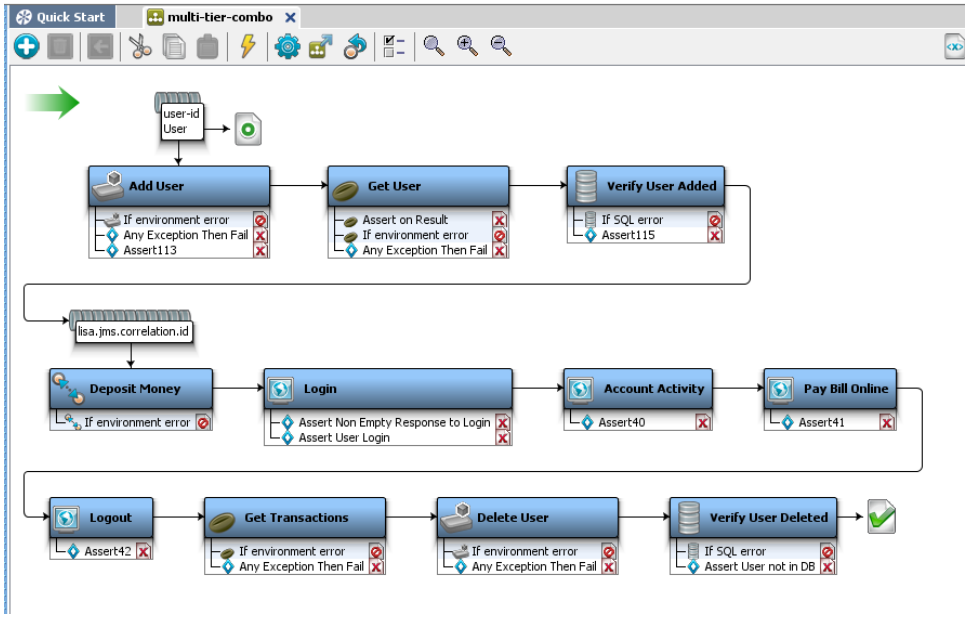


Pour plus d'informations, consultez la section Panneau Project (Projet) (page 50) de la rubrique *Utilisation de CA Application Test*.

Model Editor (Editeur de modèles)

L'éditeur Model Editor (Editeur de modèles) vous permet de créer et d'afficher le flux de travaux du scénario de test. Le flux de travaux du scénario de test rassemble toutes les étapes, les filtres et les assertions du test qui sont appliqués à un scénario de test particulier.

L'éditeur de modèles vous permet de créer et d'afficher des scénarios de test. La partie centrale de la fenêtre affiche un onglet qui correspond au nom du scénario de test.



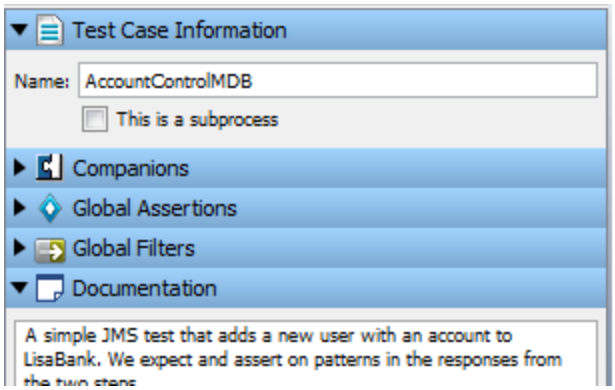
La flèche verte  marque le début du scénario de test.

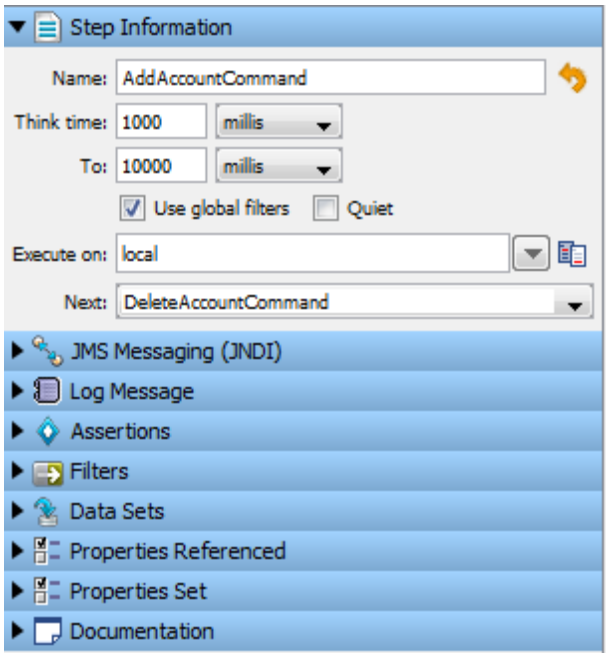
Le flux de travaux dans l'éditeur de modèles fournit une vue graphique d'un scénario de test. Cette vue est utile, car elle vous permet de valider rapidement le flux de travaux du scénario de test.

Dans l'éditeur de modèles, une icône dans le flux de travaux représente chaque étape du scénario de test. Les icônes diffèrent en fonction du type d'étape de test. Par exemple, si vous avez une étape de base de données, une icône de base de données, ainsi que les filtres et les assertions associés sont joints à l'étape.

Panneau Element

Le panneau Element contient les éléments requis pour un scénario de test ou une étape de test.





Pour remplacer le nom de l'étape par le nom d'étape par défaut, cliquez sur Revert to Default Name (Rétablir le nom par défaut) .

Vous pouvez ajouter ou supprimer un élément en cliquant sur l'élément de scénario de test ou d'étape de test requis.

Vous pouvez appliquer certains éléments au niveau global (à un scénario de test entier) ou au niveau d'une étape (uniquement à une étape de test particulière).

Entrez les informations requises au début du scénario de test.

Par exemple :

- Test Case Info (Informations sur le scénario de test) : entrez le nom du scénario de test et indiquez s'il s'agit d'un sous-processus.
- Documentation : entrez la documentation pour le scénario de test.

Une fois que vous avez ajouté des étapes à ce scénario de test, un flux de travaux d'étapes commence à se former. Un nouvel ensemble d'éléments s'affichent au niveau de l'étape de test dans le panneau Element.

Scénario de test à plusieurs niveaux

Le projet Examples (page 54) contient un scénario de test nommé **multi-tier-combo**.

Ce scénario de test utilise différents terminaux de service pour valider l'application de démonstration LISA Bank. Le scénario teste les transactions Web, SOAP, EJB et JMS, puis les valide de plusieurs façons, y compris via la validation directe de la base de données du serveur de démonstration.

Le scénario de test présente également la procédure de création d'objets SOAP complexes à partir de feuilles de calcul. La feuille de calcul **multi-tier-users.xls** du dossier Data (Données) du projet sauvegarde les données de l'utilisateur définies à la première étape.

Lorsque vous exécutez ce test dans la fenêtre Interactive Test Run (ITR) (page 293) (Exécution d'un test interactif), un utilisateur unique est créé à partir de la première ligne de la feuille de calcul, puis le test se termine.

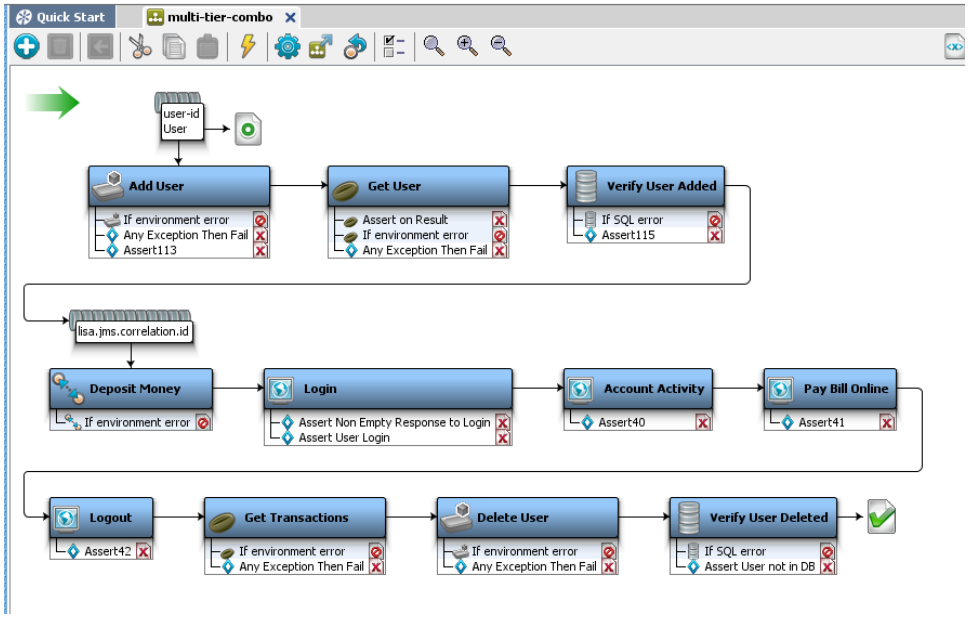
Si vous simulez le test (page 316) avec l'exemple de document de simulation **1User0Think_RunContinuously**, le test redémarre jusqu'à ce que la fin de l'ensemble de données soit atteinte. Cette méthode est recommandée pour une itération sur un grand ensemble de données. Une boucle pourrait également être utilisée dans le scénario de test, mais elle ne fournirait pas autant de flexibilité.

Si vous permettez au document de simulation qui contrôle l'ensemble de données de terminer le test, vous pouvez diffuser le test auprès de plusieurs utilisateurs virtuels. Vous pouvez également contrôler le rythme du test à l'aide de délais de réflexion par exemple.

Seuls les ensembles de données globaux définis lors de la première étape du test affectent le comportement de fin du test en continu du document de simulation. Si l'ensemble de données est local ou déclaré ailleurs dans le test, l'exécution continue signifie vraiment une exécution permanente.

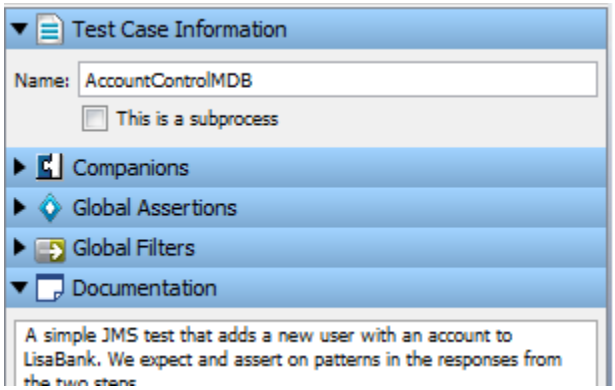
Remarquez les dossiers **d'exemples** de projet ouverts dans le panneau Project (Projet) et un ensemble d'éléments de scénario de test dans le panneau Element.

Les informations de scénario de test sont affichées dans la section Model Editor (Editeur de modèles).


Quick Start
multi-tier-combo
user-id
User
Add User
If environment error
Any Exception Then Fail
Assert113
Get User
Assert on Result
If environment error
Any Exception Then Fail
Verify User Added
If SQL error
Assert115
lisa.jms.correlation.id
Deposit Money
If environment error
Login
Assert Non Empty Response to Login
Assert User Login
Account Activity
Assert40
Pay Bill Online
Assert41
Logout
Assert42
Get Transactions
If environment error
Any Exception Then Fail
Delete User
If environment error
Any Exception Then Fail
Verify User Deleted
If SQL error
Assert User not in DB

Eléments d'un scénario de test

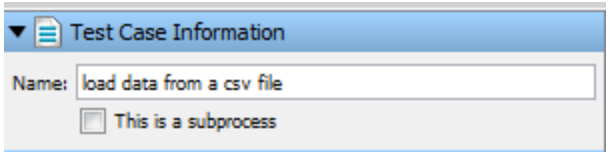
Les éléments d'un scénario de test facilitent la création de l'ensemble du scénario de test. Le graphique suivant illustre l'affichage des éléments du scénario de test dans le panneau Test Case (Scénario de test) sur le côté droit de la station de travail DevTest Workstation.



Test Case Info (Informations sur le scénario de test)

L'onglet Test Case Info (Informations sur le scénario de test) vous permet de changer le nom d'un scénario de test.

Cet onglet est également utilisé comme point de départ pour la création d'un sous-processus ou la conversion d'un scénario de test en sous-processus. Un sous-processus est un scénario de test appelé par un autre scénario de test au lieu d'être exécuté comme test autonome.

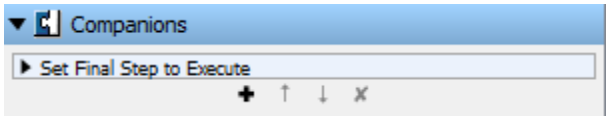


Pour plus d'informations sur les sous-processus, consultez la section Génération de sous-processus de la rubrique *Utilisation de CA Application Test.*

Compagnons

Un compagnon est un élément qui s'exécute avant et après toutes les exécutions de scénario de test. Les compagnons sont utilisés pour configurer le comportement global dans le scénario de test. Un redémarrage entraîne une nouvelle exécution des compagnons.

Pour ouvrir l'éditeur du compagnon, double-cliquez sur le compagnon dans le panneau Elements.

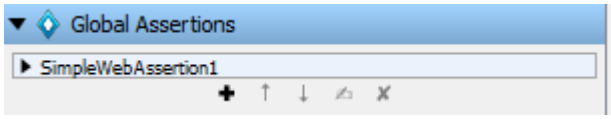


Pour plus d'informations, consultez la section Compagnons (page 173) de la rubrique *Utilisation de CA Application Test*.

Assertions globales

Une assertion est un élément qui s'exécute après l'exécution d'une étape et de tous ses filtres. Les assertions vérifient que les résultats de l'exécution de l'étape sont conformes à vos prévisions. Une *assertion globale* s'applique à l'ensemble du scénario de test.

Pour ouvrir l'éditeur d'assertions, double-cliquez sur l'assertion dans la liste Global Assertions (Assertions globales).

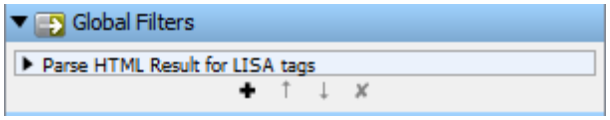


Pour plus d'informations, consultez la section Assertions (page 142) de la rubrique *Utilisation de CA Application Test*.

Filtres globaux

Un filtre est un élément qui s'exécute avant et après une étape de test. Les filtres vous donnent l'opportunité de modifier les données des résultats ou de stocker les valeurs dans des propriétés. Un *filtre global* s'applique à l'ensemble du scénario de test.

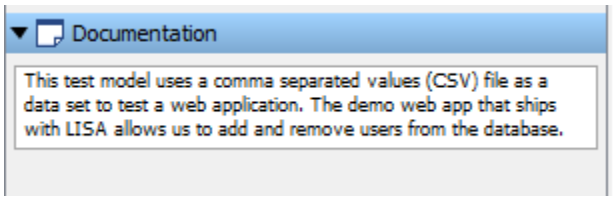
Pour ouvrir l'éditeur de filtres, double-cliquez sur le filtre dans la liste Global Filters (Filtres globaux).



Pour plus d'informations, consultez la section Filtres (page 124) de la rubrique *Utilisation de CA Application Test*.

Documentation

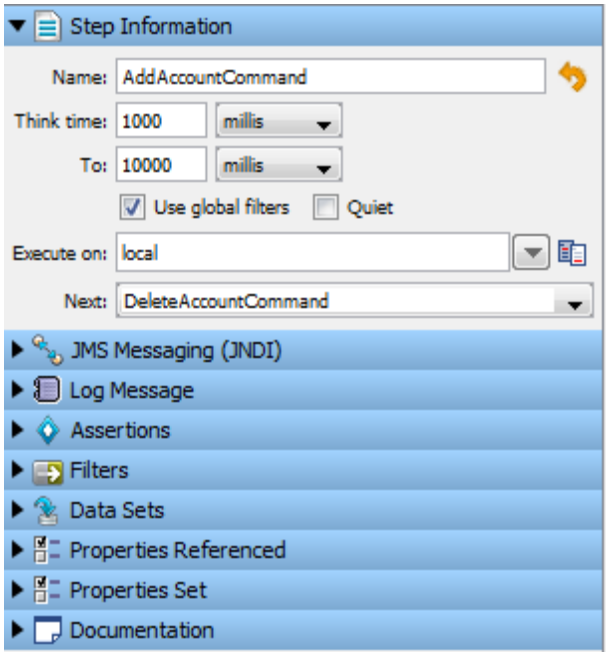
La zone Documentation vous permet d'ajouter une documentation à votre scénario de test. Ce texte n'est utilisé dans aucun processus, mais il peut être très pratique, car il vous permet de conserver une description du scénario de test et de vos remarques pour les autres utilisateurs du scénario de test.


Documentation
This test model uses a comma separated values (CSV) file as a data set to test a web application. The demo web app that ships with LISA allows us to add and remove users from the database.

Eléments d'une étape de test

Une étape de test est un élément de scénario de test d'un flux de travaux qui effectue une action de base pour valider un processus du système testé. Vous pouvez utiliser des étapes pour appeler des parties du système testé. Ces étapes sont généralement associées pour créer des flux de travaux, tels que des scénarios de test, dans l'éditeur de modèles. A partir de chaque étape, vous pouvez créer des filtres pour extraire des données ou créer des assertions pour valider des données de réponse.

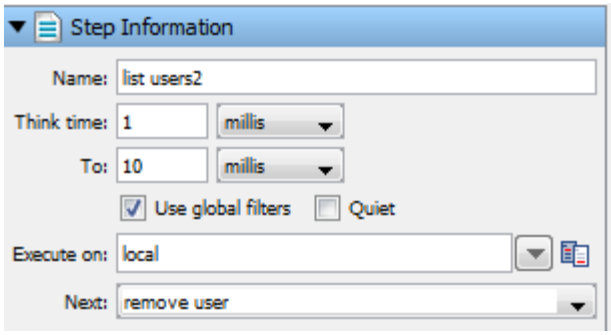
Ces éléments sont traités en détail dans la section Génération d'étapes de test (page 206) de la rubrique *Utilisation de CA Application Test.*



Step Information (Informations sur l'étape)

La section Step Info (Informations sur l'étape) vous permet de documenter des informations de base sur l'étape de test.

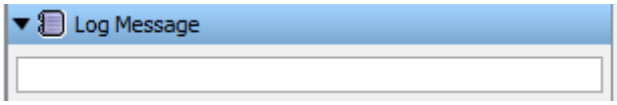
Vous pouvez entrer le nom d'étape, le délai de réflexion et les détails des champs Execute On (Exécuter au niveau de) et Next Step (Etape suivante). Vous pouvez également spécifier des paramètres pour l'exécution de l'étape à l'aide de filtres globaux et l'exécution silencieuse de l'étape.



Pour plus d'informations, consultez la section Génération d'étapes de test (page 206) de la rubrique *Utilisation de CA Application Test.*

Log Message (Message de journal)

Un message de journal est un champ de texte dans lequel vous pouvez saisir un message pour une étape. Ce message est affiché lors de l'exécution de l'étape de test ou du scénario.

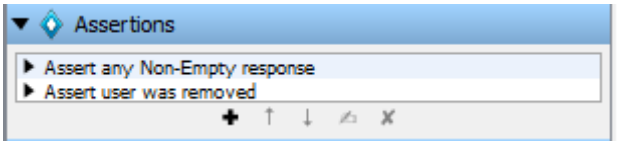


Pour plus d'informations, consultez la rubrique Enregistreur d'étape de test de la rubrique *Administration*.

Assertions

Une assertion est un élément qui s'exécute après l'exécution d'une étape et de tous ses filtres. Les assertions vérifient que les résultats de l'exécution de l'étape sont conformes à vos prévisions. Le résultat d'une assertion est toujours une valeur booléenne (true ou false).

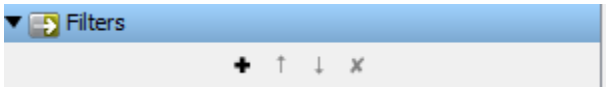
Le résultat détermine la réussite ou l'échec d'une étape de test, ainsi que l'étape suivante à exécuter dans le scénario de test. L'assertion peut donc altérer de façon dynamique le flux de travaux du scénario de test en introduisant une logique conditionnelle (création de branches) dans le flux de travaux.



Pour plus d'informations, consultez la section Assertions (page 142) de la rubrique *Utilisation de CA Application Test*.

Filters (Filtres)

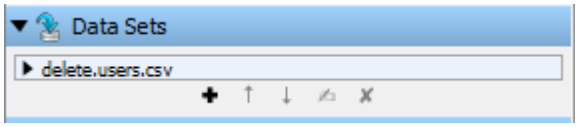
Un filtre est un élément qui s'exécute avant et après une étape de test. Les filtres vous donnent l'opportunité de modifier les données des résultats ou de stocker les valeurs dans des propriétés.



Pour plus d'informations, consultez la section Filtres (page 124) de la rubrique *Utilisation de CA Application Test*.

Ensembles de données

Un ensemble de données est une collection de valeurs que vous pouvez utiliser pour définir des propriétés dans un scénario de test pendant l'exécution d'un test. Cette capacité permet d'introduire des données de test externes à un scénario de test.

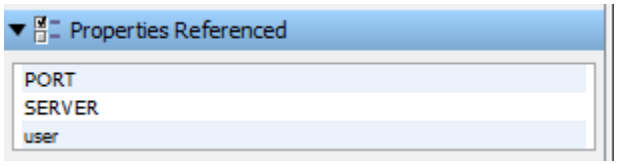


Pour plus d'informations, consultez la section Ensembles de données (page 162) de la rubrique *Utilisation de CA Application Test*.

Properties Referenced (Propriétés référencées)

Cette section contient une liste des propriétés que l'étape de test utilise ou référence.

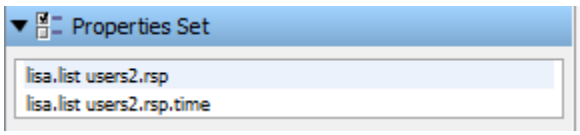
Pour ouvrir la vue étendue et obtenir sa valeur de variable, cliquez avec le bouton droit de la souris sur la propriété.



Pour plus d'informations, consultez la section Propriétés (page 93) de la rubrique *Utilisation de CA Application Test*.

Properties Set (Ensemble de propriétés)

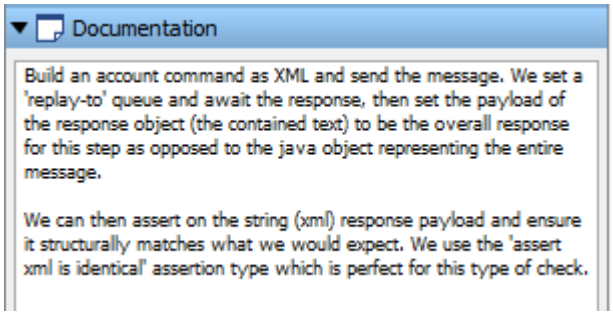
Cette section contient une liste des propriétés définies par l'étape de test. Les sections Properties Referenced (Propriétés référencées) et Properties Set (Ensembles de propriétés) appartiennent à une étape particulière et changent lorsqu'une autre étape est sélectionnée.



Pour plus d'informations, consultez la section Propriétés (page 93) de la rubrique *Utilisation de CA Application Test*.

Documentation

La zone Documentation vous permet d'ajouter une documentation à votre étape de test. Ce texte n'est utilisé dans aucun processus, mais il peut être très pratique, car il vous permet de conserver une description de votre étape de test et de vos remarques pour les autres utilisateurs de l'étape de test.



Propriétés

Les propriétés de test sont des paires nom-valeur, également appelées paires clé-valeur.

L'indépendance des données, la réutilisabilité et la portabilité des scénarios de test repose sur la capacité de remplacer des valeurs de données du scénario de test par des variables. Ces variables sont appelées des *propriétés*. Certaines propriétés sont prédéfinies et guident l'exécution de l'application. Vous créez d'autres propriétés lors de la création de tests.

Une compréhension exhaustive des propriétés est importante pour la création de scénarios de test. Dans le contexte d'un scénario de test, il est approprié d'utiliser une propriété pour chaque élément qui peut être modifié. Ce scénario inclut, par exemple, des valeurs dans les étapes de test et des valeurs dans les configurations.

Vous pouvez définir des propriétés de plusieurs façons. Une fois que les propriétés sont définies, elles peuvent être utilisées dans les étapes, les assertions et les filtres ultérieurs du scénario de test (car elles sont globales et s'appliquent au niveau du scénario de test). A quelques exceptions près, vous pouvez remplacer les propriétés d'un scénario de test.

Lorsqu'une valeur de propriété est définie, un événement Property Set (Propriété définie) est enregistré. L'événement contient le nom et la valeur de la propriété.

Les valeurs de la propriété ne sont pas limitées à des valeurs de chaîne. Une propriété peut conserver des chaînes, des nombres, des fragments XML, des objets Java adaptés en série ou la réponse complète d'une étape de test. Plusieurs propriétés sont créées pendant l'exécution d'un test et peuvent être utilisées dans des étapes de test ultérieures. Par exemple, la propriété **lisa.stepname.rsp** contient la réponse à l'étape stepname.

Les rubriques suivantes sont incluses.

Spécification d'une propriété

La syntaxe d'une propriété est {{property_name}}.

Lorsqu'une propriété est identifiée et va être utilisée, la valeur actuelle de la propriété remplace {{property_name}}. Parfois, une propriété est attendue et constitue le seul choix disponible. En d'autres occasions, vous êtes invité à spécifier le nom de la propriété explicitement. Dans ce cas, entrez le nom de la propriété sans les accolades. Utilisez la notation avec les accolades lorsque les propriétés sont intégrées dans une chaîne de texte. Une syntaxe supplémentaire permet d'utiliser des expressions de propriété : {{=expression}} ou {{property_name=expression}}.

Un nom de propriété peut contenir des espaces. Toutefois, l'utilisation d'espaces n'est pas recommandée. Vous ne pouvez pas utiliser les caractères qui définissent la syntaxe de propriété ({, } et =). Si vous référencez une propriété non existante ou non valide, DevTest conservera les accolades.

Lors de la modification de fichiers de propriétés qui contiennent des sauts de ligne de type UNIX, utilisez un éditeur approprié tel que WordPad.

Remarque : Tous les noms de propriété qui commencent par **lisa.** sont réservés à une utilisation interne. DevTest peut masquer ou supprimer des propriétés qui commencent par **lisa.**

Expressions de propriété

Les propriétés peuvent stocker plusieurs types de données différents. Elles peuvent également évaluer et stocker des expressions. Ces expressions peuvent contenir des expressions Java ou JavaScript valide que BeanShell peut évaluer. BeanShell. BeanShell est un environnement d'interprétation Java. Ces expressions peuvent également constituées des modèles de chaîne, qui donnent une apparence réelle aux chaînes factices, dans des buts divers.

Pour plus d'informations sur BeanShell, consultez la rubrique Utilisation de BeanShell dans DevTest (page 473) ou le site Web www.beanshell.org.

Pour utiliser une expression de propriété, utilisez l'un des formats suivants :

{{=expression}}

BeanShell est utilisé pour évaluer l'expression et remplacer {{expression}} par le résultat de l'évaluation. Par exemple, {{=Math.random()}} évalue la méthode Java statique et remplace la construction {{}} par le nombre aléatoire renvoyé.

{{key=expression}}

{{rand=Math.random()}} permet de définir une valeur de propriété **rand** égale au nombre aléatoire renvoyé et de remplacer la construction {{}} par ce nombre aléatoire.

Vous pouvez référencer des propriétés par nom dans une expression de propriété uniquement (sans accolades), car elles sont déjà définies comme propriétés. Si la propriété est introuvable, l'expression de propriété est renvoyée entre accolades, pour indiquer qu'un problème est survenu dans l'expression.

Modèles de chaîne

Les *modèles de chaîne* sont des types d'expressions de propriété spéciaux, dont la syntaxe se présente comme suit {{ =[:nom_modèle:] }}. Par exemple, pour formater un prénom, la propriété de modèle de chaîne peut être {{ =[:First Name:] }}. La propriété est évaluée sur un prénom factice qui ressemble à un nom réel.

Ce comportement offre un meilleur résultat que la possibilité de traiter des chaînes aléatoires qui ne ressemblent pas à un nom réel. La fonctionnalité de modèle de chaîne prend en charge plusieurs modèles en plus des prénoms : les noms, les dates, les numéros de sécurité sociale, les numéros de carte de crédit, les dates d'expiration de carte de crédit, etc. Ces données factices sont enregistrées dans la table TESTDATA de la base de données reportdb.

Si vous avez besoin d'un prénom dans votre scénario de test, il est recommandé d'utiliser {{ =[:First Name:] }} dans un ensemble de données. La partie "{{=[" est un signal indiquant d'utiliser le modèle de chaîne et qu'elle inclut une liste d'éléments enregistrés reconnus.

Par exemple, l'utilisation des informations suivantes dans une étape de journal :

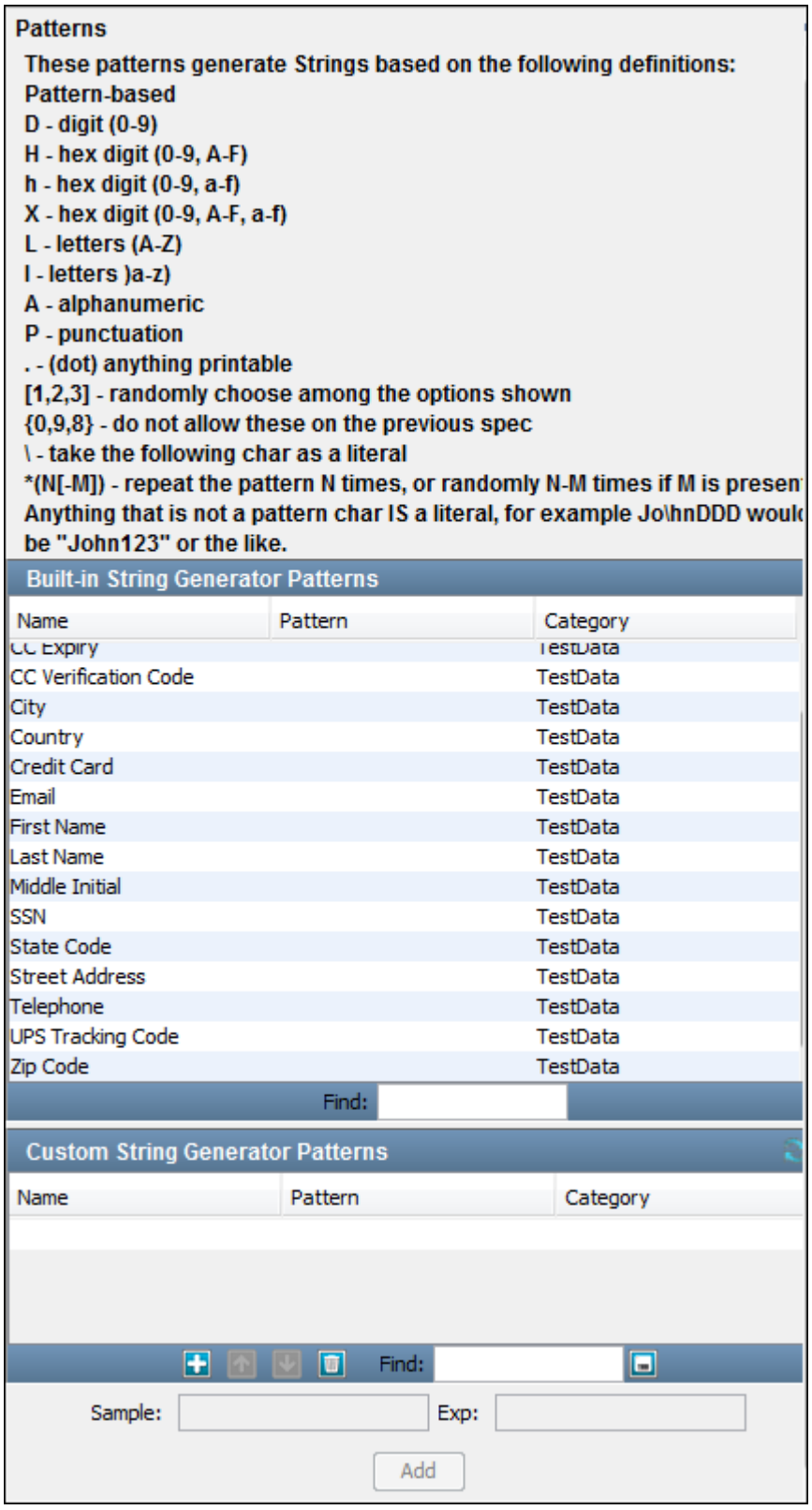
```
{{=[:First Name:]}} {{=[:Middle Initial:]}} {{=[:Last Name:]}}
{{=[:Street Address:]}}
{{=[:City:]}}, {{=[:State Code:]}}
{{=[:ZIP Code:]}} {{=[:Country:]}}
SSN: {{=[:SSN:]}}
Card: {{=[:Credit Card:]}} Expires {{=[:CC Expiry:]}}
Phone: {{=[:Telephone:]}}
Email: {{=[:Email:]}}
```

fournit la réponse suivante :

```
Marilyn M Mcguire
3071 Bailey Drive
Oelwein, IA
50662 US
SSN: 483-16-8190
Card: 4716-2361-6304-6128 Expires 3/2014
Phone: 319-283-0064
Email: Marilyn.C.Mcguire@spambob.com
```

Si vous exécutez l'étape à nouveau, vous obtenez un ensemble de données différent. DevTest suit le nombre de lignes de la base de données de test et sélectionne de manière aléatoire une ligne comprise entre 1 et N.

Pour ouvrir la fenêtre suivante, qui contient tous les modèles de chaîne existants et la documentation, cliquez sur l'onglet vertical Patterns (Modèles) toute à droite de la page.


Patterns
These patterns generate Strings based on the following definitions:
Pattern-based
D - digit (0-9)
H - hex digit (0-9, A-F)
h - hex digit (0-9, a-f)
X - hex digit (0-9, A-F, a-f)
L - letters (A-Z)
l - letters)a-z)
A - alphanumeric
P - punctuation
. - (dot) anything printable
[1,2,3] - randomly choose among the options shown
{0,9,8} - do not allow these on the previous spec
\ - take the following char as a literal
*(N[-M]) - repeat the pattern N times, or randomly N-M times if M is presen
Anything that is not a pattern char IS a literal, for example Jo\hnDDD woul
be "John123" or the like.
Built-in String Generator Patterns
Name
Pattern
Category
CC Verification Code
TestData
City
Country
Credit Card
Email
First Name
Last Name
Middle Initial
SSN
State Code
Street Address
Telephone
UPS Tracking Code
Zip Code
Find:
Custom String Generator Patterns
Name
Pattern
Category
Find:
Sample:
Exp:
Add

Informations sur l'implémentation

Les données sont stockées par défaut dans la base de données de rapports, dans la table TESTDATA.

DevTest Workstation vérifie si la table TESTDATA contient des données lorsqu'elle démarre. S'il n'y a aucune donnée, le fichier com/itko/lisa/test/data/TestData.csv de lisa-core.jar est lu pour charger la base de données. Si reports.db est supprimé, les données de test sont recréées. La lecture de la base de données est effectuée uniquement au démarrage et prend environ 15 secondes.

Création de votre propre modèle de chaîne

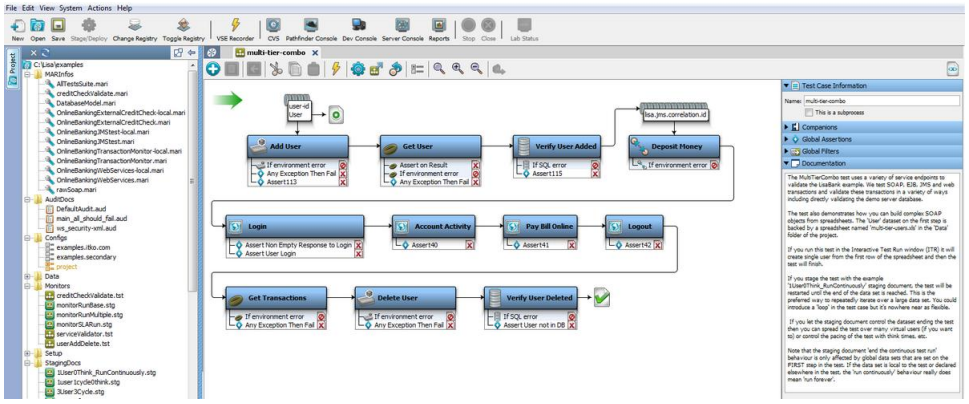
Lors de l'ajout de vos propres données, veillez à affecter l'ID correctement. Commencez par 1 et augmentez les valeurs sans espaces jusqu'à ce que vous arriviez au nombre de lignes souhaité.

Le code de générateur de chaîne sélectionne * à partir de testdata, où ID = 'n', après avoir obtenu la valeur n à partir d'un objet aléatoire. L'ID doit donc être la clé primaire de la table pour garantir des recherches efficaces.

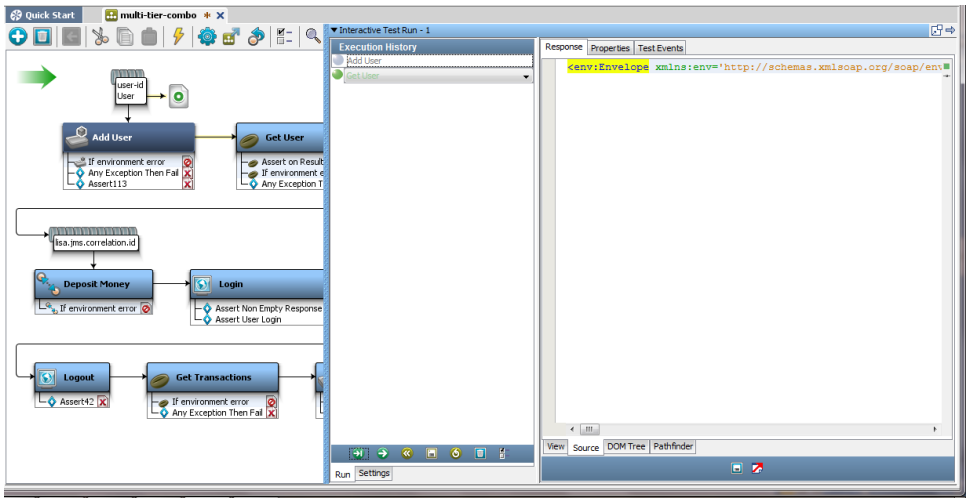
Exemple

Une bonne façon d'expérimenter avec les expressions de propriété est de créer un scénario de test simple avec une étape Outpt Log Message (Message de journal de sortie) unique. Cette étape écrit uniquement dans un journal et affiche la réponse dans le panneau Interactive Test Run Response (Réponse de l'exécution d'un test interactif). Par conséquent, vous pouvez mettre en pratique à l'aide d'expressions de propriété.

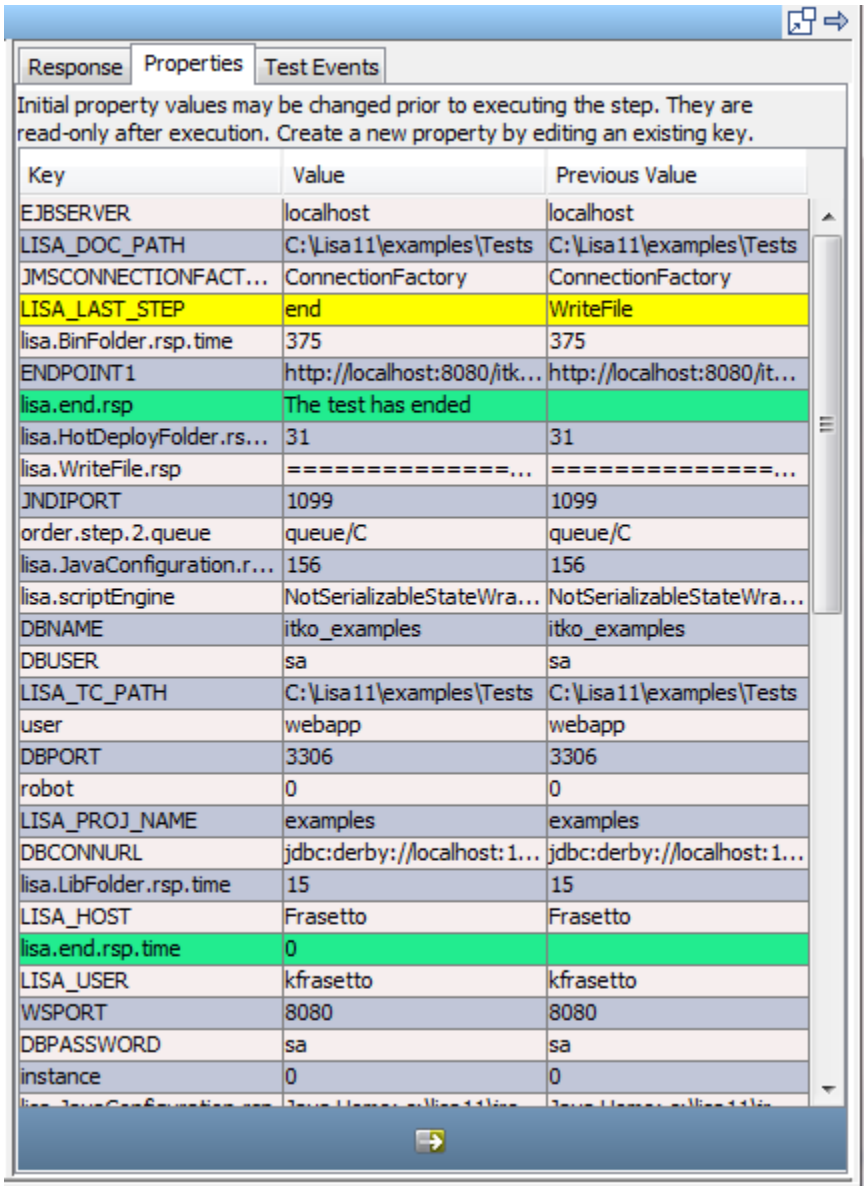
L'exemple suivant utilise le scénario de test à plusieurs niveaux du répertoire Examples (multi-tier-combo.tst) :



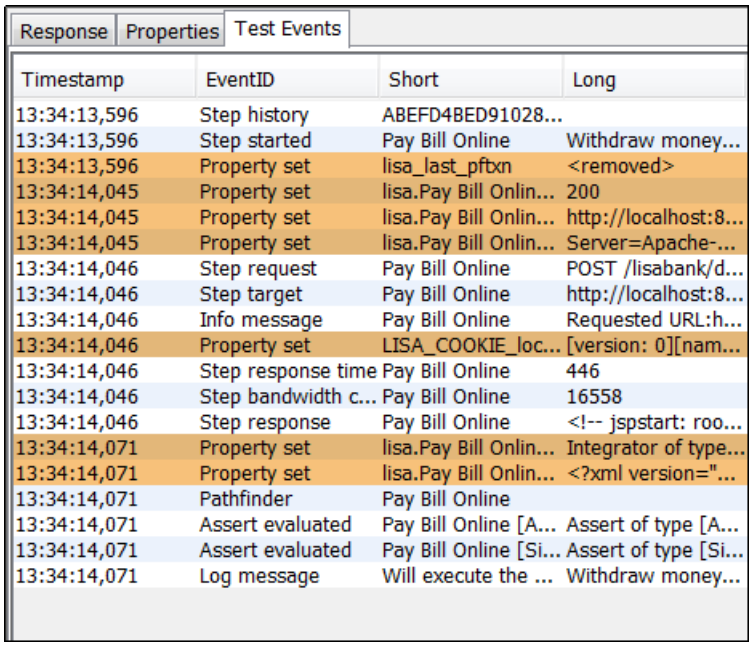
Ce graphique présente l'exemple dans l'utilitaire ITR.



Ces graphiques présentent l'onglet Properties (Propriétés) dans l'ITR. L'onglet affiché correspond à l'étape Get User (Obtenir le nom d'utilisateur).



Ce graphique présente les onglets Test Events (Evénements de test) dans l'ITR.



Recherchez l'événement **Property Set** (Propriété définie) dans l'onglet Test Events (Evénements de test).

Les développeurs Java peuvent également profiter de l'environnement BeanShell de l'étape JavaScript pour tester des expressions de propriété.

Sources des propriétés

Les propriétés peuvent être générées à partir de plusieurs sources qui incluent notamment :

- DevTest
- Variables d'environnement
- Variables de ligne de commande au démarrage
- Configurations
- Compagnons
- Etapes de test
- Filters (Filtres)
- Ensembles de données
- Modèles de chaîne

Les propriétés pouvant être remplacées, il est important de comprendre la hiérarchie des propriétés, c'est-à-dire leur ordre de lecture dans un scénario de test.

La hiérarchie suivante est utilisée :

1. Propriétés chargées pendant la configuration d'un test
2. Variables d'environnement de système d'exploitation (comme java.version ou os.user, par exemple)
3. Fichiers de propriété DevTest
4. Attributs de ligne de commande
5. Configuration par défaut
6. Propriétés de configuration alternatives (de la configuration active ou du fichier de configuration d'exécution)
7. Propriétés définies pendant l'exécution de test
8. Propriétés définies dans les compagnons
9. Propriétés définies pendant l'exécution de test (par exemple, dans des ensembles de données, des filtres et des étapes) Les propriétés définies pendant l'exécution de test remplacent les valeurs définies préalablement.

Propriétés communes et variables d'environnement

HOT_DEPLOY

Pointe vers un répertoire hotDeploy propre au projet.

LASTRESPONSE

Réponse à la dernière étape exécutée

LISA_HOME

Pointe vers le répertoire d'installation et est automatiquement défini. Cette valeur inclut une barre oblique finale. Pour, par exemple, référencer un répertoire Exemples, spécifiez :

```
{{LISA_HOME}}exemples
```

Aucune barre oblique n'est requise avant le nom du répertoire.

LISA_HOST

Nom du système sur lequel l'environnement de test s'exécute.

LISA_JAVA_HOME

Machine virtuelle Java à utiliser. Définissez cette propriété uniquement si vous ne voulez pas utiliser la machine virtuelle intégrée. Si Java n'est pas installé, DevTest utilisera le JRE fourni. Vous devez également renommer le répertoire jre dans le répertoire d'installation par jre_notinuse, par exemple.

LISA_POST_CLASSPATH

Variable utilisée pour ajouter des informations après la variable classpath de DevTest. DevTest n'utilise pas la variable CLASSPATH de l'environnement du système d'exploitation. Pour ajouter vos propres fichiers JAR après la variable classpath de DevTest, utilisez LISA_POST_CLASSPATH.

LISA_PRE_CLASSPATH

Variable utilisée pour ajouter des informations avant la variable classpath de DevTest. DevTest n'utilise pas la variable CLASSPATH de l'environnement du système d'exploitation. Pour ajouter vos propres fichiers JAR avant la variable classpath de DevTest, utilisez LISA_PRE_CLASSPATH.

LISA_PROJ_NAME

Nom du projet auquel le document actuel appartient. LISA_PROJ_NAME fait référence au projet de test appelant principal.

LISA_RELATIVE_PROJ_NAME

Nom du projet auquel le document actuel appartient. LISA_RELATIVE_PROJ_NAME fait référence au projet du test ou sous-processus en cours d'exécution. Si le test n'appelle aucun sous-processus, LISA_PROJ_NAME et LISA_RELATIVE_PROJ_NAME sont identiques.

LISA_PROJ_PATH

Chemin complet du répertoire de projet. La valeur dépend du système d'exploitation. Une barre oblique inversée (\) est utilisée comme caractère de séparation sur Windows. Une barre oblique (/) est utilisée comme caractère de séparation sur les autres systèmes d'exploitation. L'exemple suivant s'appuie sur une installation Windows :

```
C:\Program Files\LISA\examples
```

Une restriction limite l'utilisation de la variable LISA_PROJ_PATH dans une étape Java personnalisée : la syntaxe {{LISA_PROJ_PATH}} n'est pas prise en charge. L'étape Java personnalisée appelle un compilateur Java pour compiler le script et Java traite les barres obliques inversées comme des caractères d'échappement dans des chaînes. Par conséquent, cette chaîne spécifique déclenche une erreur de compilateur. La solution est d'utiliser LISA_PROJ_PATH en tant que variable. Par exemple :

```
File f = new File ( LISA_PROJ_PATH );
```

LISA_RELATIVE_PROJ_PATH

Il s'agit du chemin complet du répertoire de projet du test ou sous-processus en cours d'exécution. LISA_PROJ_PATH fait référence au test appelant principal. Consultez la propriété LISA_PROJ_PATH pour plus d'informations. Si le test n'appelle aucun sous-processus, LISA_PROJ_PATH et LISA_RELATIVE_PROJ_PATH sont identiques.

LISA_PROJ_ROOT

Chemin complet du répertoire de projet. La valeur ne dépend pas du système d'exploitation. Une barre oblique (/) est utilisée comme caractère de séparation sur tous les systèmes d'exploitation, y compris Windows. L'exemple suivant s'appuie sur une installation Windows :

```
C:/Program Files/LISA/examples
```

LISA_RELATIVE_PROJ_ROOT

Il s'agit du chemin complet du répertoire de projet du test ou sous-processus en cours d'exécution. LISA_PROJ_ROOT fait référence au test appelant principal. Consultez la propriété LISA_PROJ_ROOT pour plus d'informations. Si le test n'appelle aucun sous-processus, LISA_PROJ_ROOT et LISA_RELATIVE_PROJ_ROOT sont identiques.

LISA_PROJ_URL

URL du répertoire de projet. Par exemple :

```
file:/C:/Program%20Files/LISA/examples
```

LISA_RELATIVE_PROJ_URL

Il s'agit de l'URL du répertoire de projet du test ou sous-processus en cours d'exécution. LISA_PROJ_URL fait référence au test appelant principal. Consultez la propriété LISA_PROJ_URL pour plus d'informations. Si le test n'appelle aucun sous-processus, LISA_PROJ_URL et LISA_RELATIVE_PROJ_URL sont identiques.

LISA_TC_PATH

Chemin complet du répertoire dans lequel le scénario de test se trouve.

LISA_TC_URL

URL du répertoire dans lequel le scénario de test se trouve.

LISA_USER

Utilisateur qui a chargé le scénario de test.

Fichiers de propriété

Les principaux fichiers de propriété sont les suivants :

- lisa.properties
- local.properties
- site.properties

Des informations détaillées sur ces propriétés sont disponibles dans les annexes suivantes :

- Annexe A (page 485) : Fichier de propriétés DevTest (lisa.properties)
- Annexe B (page 531) : fichiers de propriété personnalisés (local.properties, site.properties)

Utilisation du volet Properties (Propriétés)

Le volet Properties vous permet d'afficher les propriétés associées à un scénario de test. Ce volet simplifie également le processus d'ajout de propriétés à une étape.

Procédez comme suit:

1. Ouvrez le scénario de test à modifier.
2. Pour ouvrir le volet Properties (Propriétés), effectuez l'une des opérations suivantes :
   - Cliquez sur l'onglet vertical Properties sur le côté droit de la page.
   - Cliquez sur Help (Aide), Properties dans le menu principal.
   - Cliquez sur  Show model properties (Afficher les propriétés du modèle) dans la barre d'outils du scénario de test.

   Le volet Properties (Propriétés) s'ouvre.
3. Accédez au champ de texte de l'arborescence d'éléments dans lequel vous voulez ajouter une propriété et cliquez dans le champ de texte.
4. Sélectionnez la propriété à ajouter dans le volet Properties (Propriétés).
   - Pour rechercher une propriété par nom, utilisez le champ Find (Rechercher) dans la partie inférieure du volet.
   - Pour limiter la liste à une catégorie spécifique de propriétés, sélectionnez une propriété dans la liste de la partie supérieure du volet.
   - Pour afficher les propriétés globales de cette instance DevTest, cliquez sur Global Properties (Propriétés globales) tout à droite de la page.
5. Cliquez sur Add (Ajouter) au bas du volet Property.

   La propriété est ajoutée au champ sélectionné.

Configurations

Une configuration est une collection nommée de propriétés qui spécifient en général des valeurs propres à un environnement pour le système testé.

En supprimant les données d'environnement codées de manière irréversible du scénario de test, vous pouvez exécuter le même test dans des environnements différents en utilisant simplement une configuration différente. Les configurations sont utilisées de manière généralisée dans DevTest ; par exemple, dans un document de scénario de test, un document de suite de tests, un document de simulation, une exécution de scénario de test ou une exécution de suite de tests.

Une configuration doit être définie au niveau du projet. Vous pouvez spécifier les valeurs de ces propriétés au début d'un scénario de test.

La configuration par défaut d'un projet est **project.config**. Vous pouvez créer d'autres configurations dans un projet et en activer une pour un scénario de test ou une suite.

Lorsque vous créez une configuration, vous ne pouvez pas lui ajouter de nouvelles clés. Pour ajouter des clés dans la nouvelle configuration, ajoutez-les au fichier project.config. Vous pouvez ensuite sélectionner les clés nouvellement définies ajoutées au fichier project.config à partir de la liste déroulante disponible dans le nouveau fichier de configuration.

Les propriétés sont ajoutées à votre configuration automatiquement au cours du développement du test.

Par exemple, lorsque vous entrez le nom du fichier WSDL dans un test, le nom du serveur et le port définis sont remplacés par des propriétés telles que WSSERVER et WSPORT. Les valeurs de ces propriétés sont automatiquement ajoutées à votre configuration de projet par défaut. Vous pouvez changer l'emplacement du service Web simplement en modifiant la configuration, au lieu de rechercher les valeurs codées de manière irréversible dans plusieurs étapes de test.

Dans un autre exemple, lorsque vous utilisez des objets Enterprise JavaBeans (EJB) ou Java, vous pouvez basculer des répertoires de déploiement à chaud ou ajouter des fichiers JAR supplémentaires à votre chemin de classe, pour utiliser des versions différentes du code Java. Deux propriétés standard existent pour ces emplacements : HOT_DEPLOY et MORE_JARS. Vous pouvez définir ces propriétés dans votre configuration.

Pour plus d'informations sur les autres propriétés, consultez la rubrique Propriétés (page 93).

Les configurations servent à stocker des propriétés associées au système testé. Evitez de les utiliser pour le stockage de paramètres globaux ou d'autres paramètres non relatifs au test. Vous pouvez stocker ces paramètres dans un compagnon (page 173).

Remarque : Les barres obliques inversées "\" ne sont pas conservées dans les fichiers de configuration. Si vous modifiez le fichier de configuration manuellement et ajoutez un élément avec une barre oblique inversée, le fichier est écrasé sans les barres obliques inversées.

Un fichier de configuration est un fichier texte avec l'extension .config dans lequel les propriétés sont indiquées sous la forme de paires clé-valeur. Les fichiers de configuration font partie intégrante de chaque exécution de test.

Vous pouvez créer des fichiers de configuration ou les modifier dans un éditeur de texte, puis les enregistrer avec l'extension .config.

Lors du démarrage d'une exécution de test, vous pouvez sélectionner un fichier de configuration. Pour plus d'informations, consultez la section Application d'une configuration au cours de l'exécution d'un scénario de test (page 115) de la rubrique *Utilisation de CA Application Test.*

Il est recommandé d'établir une convention d'attribution d'un nom pour les fichiers de configuration, afin de simplifier l'identification d'autres configurations.

Ces fichiers de configuration doivent être importés dans DevTest.

Les rubriques suivantes sont incluses.

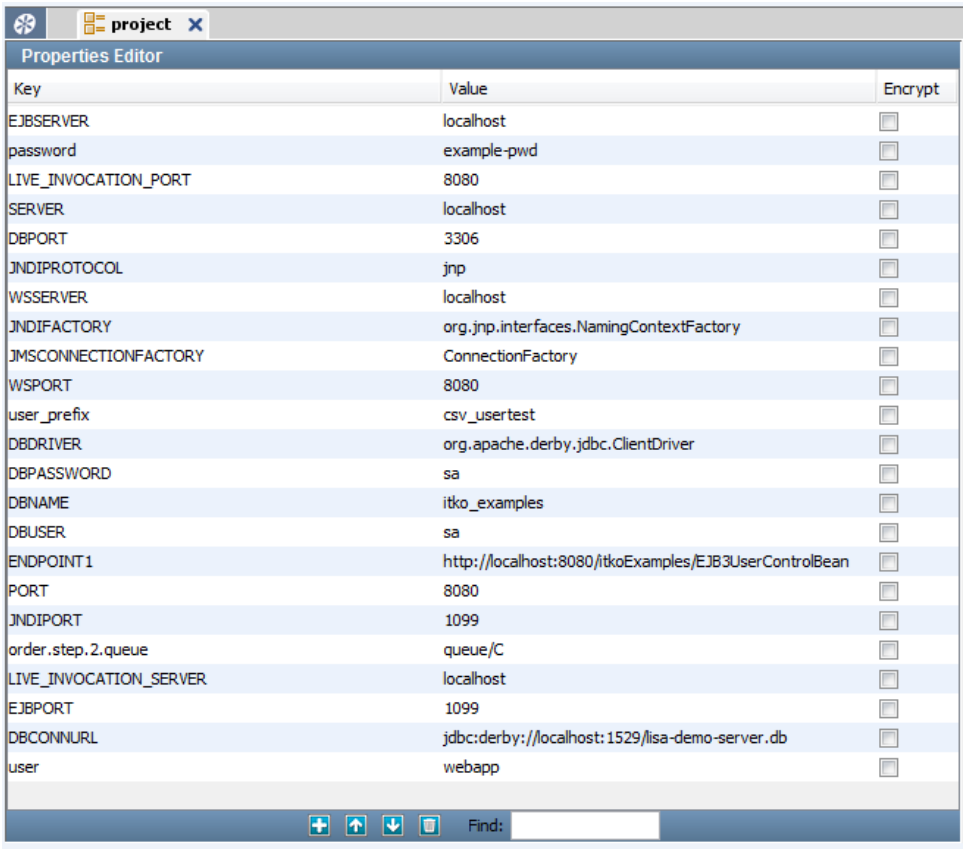
Configuration de projet

Tous les projets comportent un fichier de configuration **project.config**. Ce fichier est également appelé *configuration de projet*.

Dans le dossier Configs (Configurations) du panneau Project (Projet), la configuration de projet s'affiche en orange. L'extension de fichier n'est pas affichée.

Vous ne pouvez pas renommer ou supprimer la configuration de projet.

Lorsque vous double-cliquez sur la configuration de projet, la fenêtre Properties Editor (Editeur de propriétés) s'ouvre. L'illustration suivante représente la fenêtre Properties Editor. L'éditeur comprend trois colonnes : Key (Clé), Value (Valeur) et Encrypt (Chiffrer).



Ces paramètres sont des paramètres standard disponibles dans toutes les configurations.

La configuration de projet contient le surensemble de clés qui sont définies dans toutes les autres configurations. Pour ajouter des paramètres aux autres configurations, incluez-les dans la configuration de projet. Vous pouvez alors sélectionner une valeur dans la liste déroulante des autres configurations.

Par défaut, la configuration de projet est la configuration active d'un projet. Vous pouvez changer (page 112)la configuration active d'un projet.

Ajout d'une configuration

Un projet peut comprendre plusieurs configurations, mais seule une configuration peut être active. Vous pouvez activer une configuration alternative ou la configuration par défaut. Les configurations alternatives peuvent uniquement remplacer les propriétés par défaut.

Pour ajouter des clés dans la nouvelle configuration, ajoutez-les à la configuration de projet. Vous pouvez sélectionner les clés nouvellement définies à partir de la liste déroulante du nouveau fichier de configuration. Dans un nouveau fichier de configuration, vous ne pouvez pas ajouter de nouvelles clés.

Lorsque vous créez un fichier de configuration, les seules clés que vous pouvez ajouter sont celles qui sont déjà définies dans la configuration de projet. Certaines clés standard sont fournies avec l'installation.

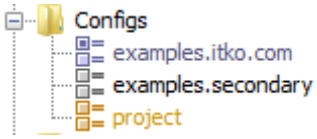
Procédez comme suit:

1. Cliquez avec le bouton droit de la souris sur le dossier Configs (Configurations) dans le panneau Project (Projet) et sélectionnez Create New Config (Créer une configuration).
2. Entrez le nom de la nouvelle configuration.
3. Cliquez sur OK.

Activation d'une configuration

Lorsque vous voulez appliquer une configuration différente à votre projet, activez-la. Dans le dossier Configs (Configurations) du panneau Project (Projet), vous pouvez activer une configuration. La configuration active s'applique au projet dans sa totalité, pas uniquement au scénario de test.

Lorsque la configuration de projet est active, elle est signalée en orange et les autres configurations en noir. Lorsqu'une configuration secondaire est activée, elle est signalée en violet et la configuration de projet reste orange.



Chaque scénario de test d'un projet unique partage la configuration active appliquée au projet. Vous ne pouvez pas affecter une configuration distincte à chaque scénario de test dans un projet.

Si une configuration active est sélectionnée, l'étape Stage a Quick Test (Simuler un test rapide) l'utilise. Dans le cas contraire, la configuration par défaut (project.config) est utilisée.

Pour activer une configuration :

1. Cliquez avec le bouton droit de la souris sur la configuration dans le dossier Configs et sélectionnez Make Active (Activer).

Modification d'une configuration

Dans une configuration autre que la configuration de projet, vous pouvez ajouter des propriétés uniquement si elles existent dans la configuration de projet.

Une bonne pratique est d'utiliser des propriétés dans les chemins d'accès stockés dans la configuration. Les propriétés, comme **LISA_HOME** ou **LISA_PROJ_ROOT**, permettent la portabilité des scénarios de test.

Une valeur de propriété peut contenir plusieurs lignes.

La vue étendue consiste en une boîte de dialogue permettant de modifier une valeur de propriété. Cette vue peut être utile lorsque la valeur est longue ou contient plusieurs lignes. Pour accéder à la vue étendue, cliquez avec le bouton droit de la souris sur la cellule de valeur de la propriété et sélectionnez Launch Extended View (Ouvrir la vue étendue).

Procédez comme suit:

1. Double-cliquez sur la configuration dans le dossier Configs.

   L'éditeur Properties Editor (Editeur de propriétés) s'affiche.

2. Ajoutez une propriété en cliquant sur  Add (Ajouter) au bas de l'éditeur de propriétés.

   Une nouvelle ligne est ajoutée à la liste de propriétés.

3. Cliquez sur la liste déroulante pour sélectionner la clé choisie.

   Les noms de propriété commune s'affichent dans la liste déroulante.

   Valeurs :

   HOT_DEPLOY

   Indique l'emplacement du répertoire de déploiement à chaud.

   MORE_JARS

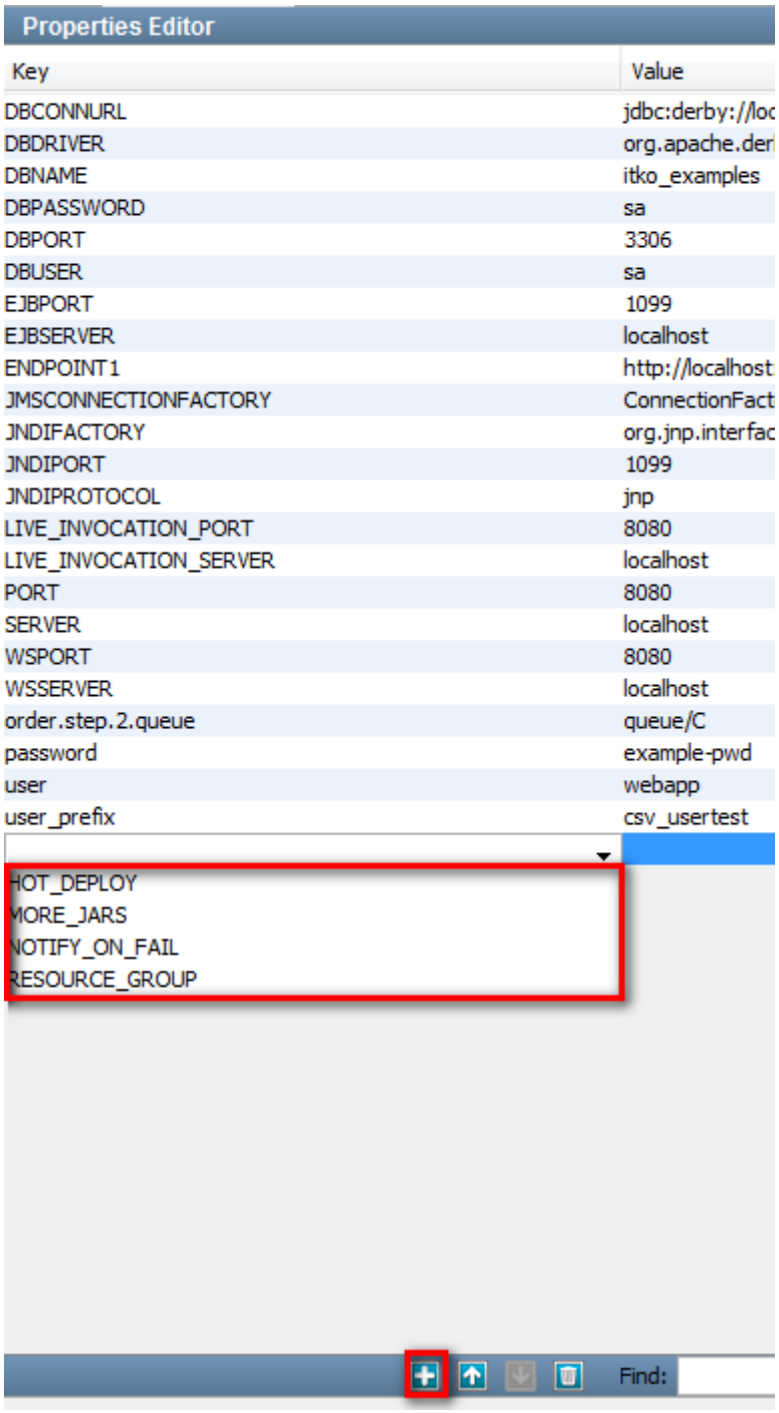
   Ajoute des fichiers JAR supplémentaires au chemin de classe.

   NOTIFY_ON_FAIL

   Définit une adresse électronique à notifier lorsqu'un échec du scénario de test se produit.

   RESOURCE_GROUP

   Filtre la sélection de coordinateurs et de VSE selon les ressources définies dans un groupe de ressources.


Properties Editor
Key
Value
DBCONNURL
jdbc:derby://loc
DBDRIVER
org.apache.der
DBNAME
itko_examples
DBPASSWORD
sa
DBPORT
3306
DBUSER
sa
EJBPORT
1099
EJBSERVER
localhost
ENDPOINT1
http://localhost
JMSCONNECTIONFACTORY
ConnectionFact
JNDIFACTORY
org.jnp.interfac
JNDIPORT
1099
JNDIPROTOCOL
jnp
LIVE_INVOCATION_PORT
8080
LIVE_INVOCATION_SERVER
localhost
PORT
8080
SERVER
localhost
WSPORT
8080
WSSERVER
localhost
order.step.2.queue
queue/C
password
example-pwd
user
webapp
user_prefix
csv_usertest
HOT_DEPLOY
MORE_JARS
NOTIFY_ON_FAIL
RESOURCE_GROUP
Find:

Application d'une configuration au cours de l'exécution d'un scénario de test

Pour plus d'informations sur l'application d'une configuration lors de l'exécution d'un scénario de test, consultez la section Simulation d'un scénario de test (page 316) de la rubrique *Utilisation de CA Application Test.*

Ressources

Un *actif* est un ensemble de propriétés de configuration groupées dans une unité logique.

En général, un actif représente un point de communication entre une application externe ou un composant de client/serveur intermédiaire et une application. Les types d'actifs peuvent être une fabrique de connexion JMS, une destination JMS, un contexte JNDI et une destination SAP JCo.

L'avantage principal des actifs est de pouvoir les réutiliser. Cela vous évite de devoir entrer les mêmes propriétés plusieurs fois. A la place, vous définissez les propriétés une fois dans un actif.

Les paramètres suivants sont communs à tous les actifs :

- Name (Nom)
- Description
- Run-time scope (Etendue d'exécution)

Les actifs sont associés à une configuration (page 108). Lorsque vous ouvrez une configuration, le navigateur d'actifs se trouve à droite de l'éditeur de propriétés. La configuration de projet contient le surensemble composé de tous les actifs dans les autres configurations. Vous pouvez configurer un actif dans l'une des autres configurations pour remplacer l'actif correspondant dans la configuration de projet.

Vous pouvez créer des actifs de zéro ou à partir d'étapes de test.

Navigateur d'actifs

Lorsque vous ouvrez une configuration, le navigateur d'actifs se trouve à droite de l'éditeur de propriétés.

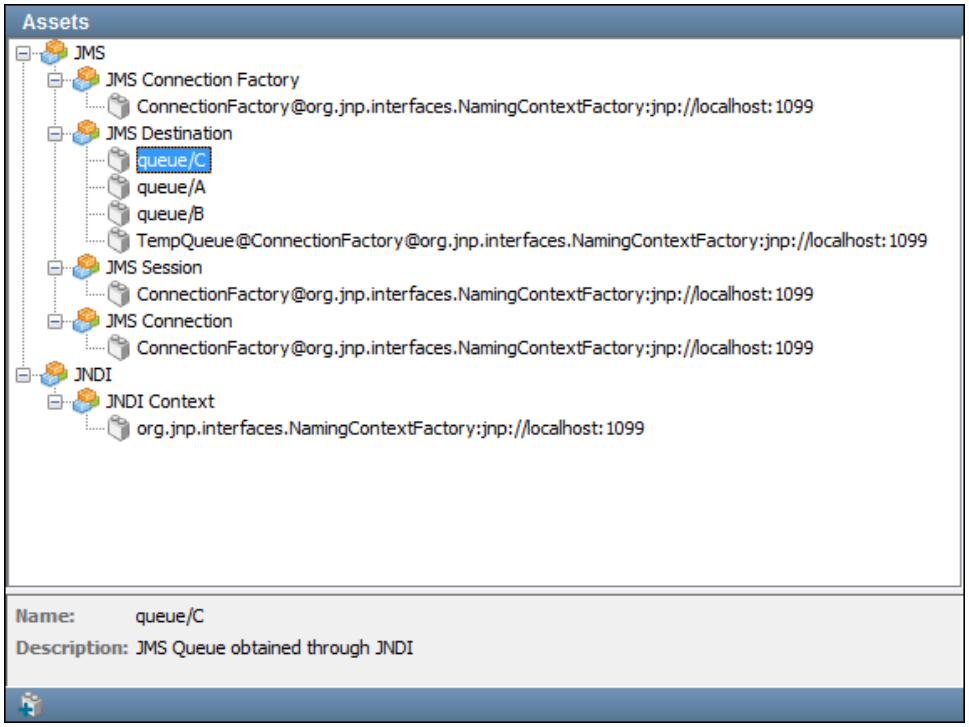
Remarque : Si le navigateur d'actifs est vide, aucun actif n'a été créé.

La zone principale contient une arborescence. Les noeuds de niveau supérieur correspondent aux *catégories d'actif*. Les noeuds de niveau secondaire correspondent aux *types d'actif*.

Les paramètres de nom et de description s'affichent sous la zone principale.

Un bouton Add (Ajouter) s'affiche dans le coin inférieur gauche.

L'illustration suivante représente le navigateur d'actifs. Les actifs de cet exemple sont divisés en deux catégories : JMS et JNDI. La catégorie JMS a quatre types d'actif : JMS Connection Factory (Fabrique de connexion JMS), JMS Destination, JMS Session et JMS Connection (Connexion JMS). La catégorie JNDI a un type d'actif : JNDI Context (Contexte JNDI).



Si un actif est associé à une configuration de projet, vous pouvez créer une copie de l'actif en le cliquant avec le bouton droit de la souris et en sélectionnant l'option Duplicate (Dupliquer).

Si un actif est associé à une configuration supplémentaire, vous pouvez créer une copie de l'actif en le cliquant avec le bouton droit de la souris et en sélectionnant l'option Duplicate in project config (Dupliquer dans la configuration de projet).

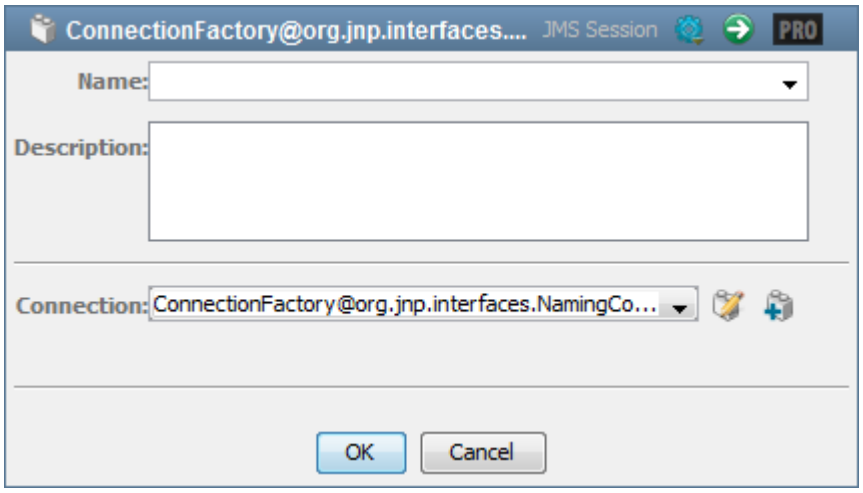
Editeur d'actifs

Utilisez l'éditeur d'actifs pour effectuer les tâches suivantes :

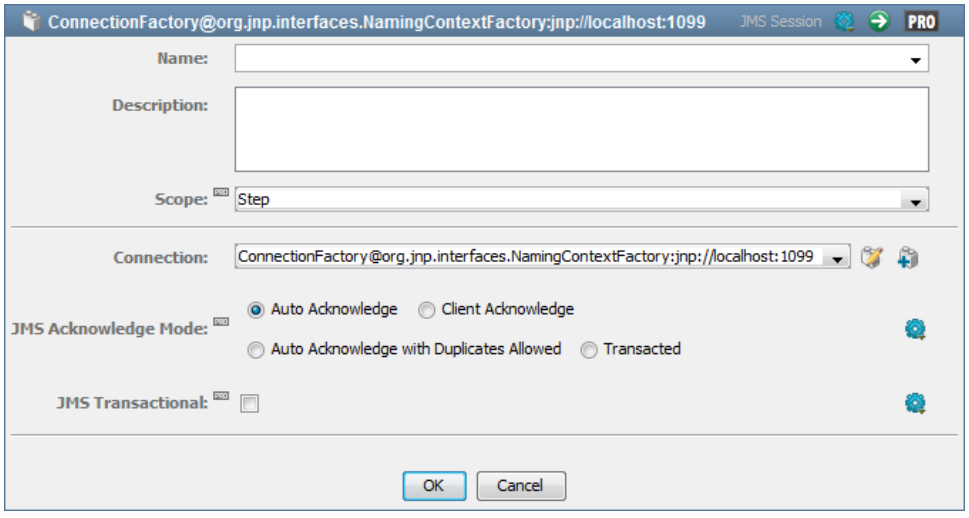
- Créer des actifs de zéro
- Modifier des actifs

L'éditeur d'actifs comprend deux modes : de base et avancé.

L'illustration suivante représente le mode de base. Les paramètres de nom et de description sont communs à tous les actifs. Les paramètres situés sous le premier séparateur horizontal varient selon le type d'actif.



L'illustration suivante représente le mode avancé. Vous pouvez afficher le mode avancé en cliquant sur PRO dans le coin supérieur droit.



Un actif peut avoir un paramètre dont la valeur est un autre type d'actif. Par exemple, l'actif de session JMS a un paramètre dans lequel vous spécifiez un actif de connexion JMS. L'actif de session JMS a un paramètre dans lequel vous spécifiez un actif de fabrique de connexion JMS.

Certains paramètres vous permettent de modifier l'éditeur de sorte à saisir une propriété comme valeur.

D'autres paramètres fournissent un ensemble discret de valeurs et vous permettent de modifier l'éditeur de sorte à pouvoir entrer la valeur directement.

Vous pouvez utiliser le bouton vert de la barre de titre pour vérifier (page 123) l'actif.

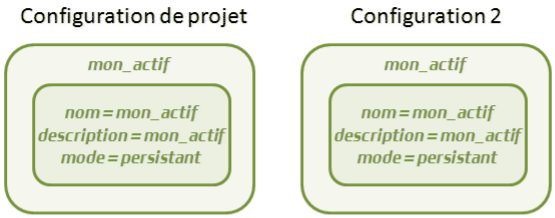
Héritage d'actif

Chaque projet comprend une configuration de projet. Un projet peut également avoir une ou plusieurs configurations supplémentaires.

La configuration de projet contient le surensemble composé de tous les actifs dans les autres configurations.

Vous pouvez configurer un actif dans une configuration supplémentaire pour remplacer l'actif correspondant dans la configuration de projet.

Le graphique suivant affiche un exemple. La configuration de projet a un actif nommé **myAsset**. Cet actif inclut les paramètres suivants : name (nom), description et mode. La deuxième configuration a le même actif. Toutefois, la valeur du paramètre mode dans la deuxième configuration est différente de la valeur du paramètre mode dans la configuration de projet. L'actif de la deuxième configuration remplace donc l'actif de la configuration de projet.



L'actif peut même être une classe différente, pourvu que le type soit le même.

Par exemple, supposez que la configuration de projet a un actif de fabrique de connexion IBM MQ. Dans la configuration supplémentaire, vous pouvez modifier l'actif de sorte à le définir comme un actif de fabrique de connexion TIBCO. Cette modification est possible, car l'actif de fabrique de connexion IBM MQ et l'actif de fabrique de connexion TIBCO ont le même type : JMS Connection Factory (Fabrique de connexion JMS).

Le navigateur d'actifs (page 116) utilise un code couleur pour indiquer le paramètre d'héritage des actifs dans chaque configuration supplémentaire :

- Si un actif est gris, il ne remplace aucun actif dans la configuration de projet.
- Si un actif est noir, il remplace un actif dans la configuration de projet.

Etendue d'exécution

L'étendue d'exécution définit le niveau minimum auquel une instance d'actif ouverte est mise en cache et réutilisée lors de l'exécution.

Chaque actif a une étendue d'exécution. Une opération dans une étape de test peut également avoir une étendue d'exécution.

Les valeurs valides sont les suivantes :

- **Step** (Etape) : une instance d'actif ouverte est mise en cache et réutilisée pendant l'étape qui permet d'y accéder. Lorsque l'étape est finie, l'instance d'actif est fermée.
- **Model** (Modèle) : une instance d'actif ouverte est mise en cache et réutilisée pendant la durée de l'exécution du modèle qui permet d'y accéder. Lorsque l'exécution du modèle est finie, l'instance d'actif est fermée.
- **Staged** (Simulé) : une instance d'actif ouverte est mise en cache et réutilisée par toutes les instances du modèle qui permet d'y accéder. Lorsque toutes les instances du modèle simulé sont finies, l'instance d'actif est fermée.
- **Global :** une instance d'actif ouverte est mise en cache et réutilisée globalement par tous les clients dans la machine virtuelle Java actuelle. Lorsqu'une instance d'actif a été inactive pendant un court instant, l'instance d'actif est fermée. Cette étendue est la plus grande possible.
- **Default (Valeur par défaut) :** une instance d'actif ouverte est affectée à la plus petite étendue disponible.

Si un actif est utilisé dans une opération, les étendues d'exécution pour l'actif et l'opération peuvent différer. DevTest évalue les étendues d'exécution et détermine la plus appropriée à utiliser.

Création d'actifs

Vous pouvez créer des actifs dans le navigateur d'actifs (page 116) ou l'éditeur d'actifs (page 118).

Procédez comme suit:

1. Effectuez l'une des opérations suivantes :
   - Dans le navigateur d'actifs, cliquez sur le bouton Add (Ajouter) dans le coin inférieur gauche et sélectionnez le type d'actif.
   - Dans l'éditeur d'actifs, cliquez sur l'icône Add New Asset (Ajouter un nouvel actif) et sélectionnez le type d'actif. Les types d'actif disponibles dépendent du paramètre dans lequel l'icône s'affiche.
2. Saisissez les informations requises. Si vous laissez le champ Name (Nom) vide, un nom est automatiquement généré.
3. Cliquez sur OK.

Création d'actifs à partir d'étapes de test

Vous pouvez créer des actifs à partir d'étapes de test JMS.

Vous pouvez créer plusieurs actifs à partir d'une étape de test unique. Par exemple, une étape JMS Messaging (JNDI) (Messagerie JMS (JNDI)) peut aboutir aux types d'actif suivants :

- JMS Connection Factory (Fabrique de connexion JMS)
- Connexion JMS
- JMS session
- JMS destination
- JNDI context (Contexte JMS)

Si vous effectuez une réexportation à partir d'une étape de test, toutes les modifications apportées à un actif à partir de l'exportation précédente sont remplacées.

Procédez comme suit:

1. Ouvrez un scénario de test.
2. Sélectionnez une ou plusieurs étapes de test.
3. Dans la barre d'outils de l'éditeur de modèles, cliquez sur l'icône Generate assets from the selected steps (Générer les actifs à partir des étapes sélectionnées)

Modification des actifs

Cette rubrique décrit la procédure de modification d'un actif existant.

Les modifications que vous pouvez effectuer concernent notamment la classe d'actifs. Par exemple, vous pouvez convertir un actif de session JMS en un actif de session de file d'attente JMS ou en un actif de session de rubrique JMS. L'icône permettant de modifier la classe d'actifs se trouve dans la barre de titre de l'éditeur d'actifs.

Procédez comme suit:

1. Effectuez l'une des actions suivantes :
    a. Dans le navigateur d'actifs, double-cliquez sur un actif.
    b. Dans le navigateur d'actifs, cliquez avec le bouton droit de la souris sur un actif et sélectionnez Edit (Modifier) ou Override (Remplacer).
    c. Dans un éditeur d'étapes, cliquez sur Edit Selected Asset (Modifier l'actif sélectionné).
    d. Dans l'éditeur d'un autre actif qui inclut l'actif sélectionné en tant que paramètre, cliquez sur Edit Selected Asset.
    e. Apportez les modifications appropriées.
2. Cliquez sur OK (Redéployer/déployer).

Vérification des actifs

Vous pouvez vérifier un actif à partir de l'éditeur d'actifs (page 118) au cours des procédures suivantes :

- Création d'actifs
- Modification d'actifs

Le processus de vérification tente d'accéder à l'objet que l'actif représente, mais n'effectue aucune opération fonctionnelle.

Procédez comme suit:

1. Vérifiez que l'application, ou les terminaux qu'elle utilise, sont en cours d'exécution et disponibles pour DevTest.
2. Dans l'éditeur d'actifs, cliquez sur le bouton Verify (Vérifier) vert.
3. Affichez la fenêtre de journal et recherchez un message de réussite ou d'échec.

Filtres

Un filtre est un élément qui s'exécute avant et après une étape de test, en vous donnant l'opportunité de changer les données dans les résultats ou de stocker les valeurs dans des propriétés. Vous pouvez utiliser un filtre pour extraire une valeur d'une page Web, de réponses XML et DOM, d'un objet Java, d'un document de texte et de plusieurs autres réponses d'étape de test.

La plupart des filtres s'exécutent après l'exécution de l'étape.

Une fois que les données ont été filtrées, vous pouvez utiliser les données dans une assertion ou dans une étape de test ultérieure. Les filtres s'appliquent en général à la réponse du système testé. Par exemple, les filtres sont utilisés pour analyser des valeurs d'une page HTML ou pour effectuer des conversions avec la réponse. Les filtres peuvent également être utiles à d'autres endroits. Vous pouvez utiliser des filtres pour enregistrer une valeur de propriété dans un fichier ou convertir une propriété de sorte à être la dernière réponse. Les filtres sont surtout utilisés pour définir des propriétés.

Vous pouvez appliquer un filtre en tant que filtre global ou d'étape. Les types de filtre disponibles sont les mêmes, mais leur application diffère.

Filtre global

Un filtre global est un filtre défini au niveau du scénario de test. Il est exécuté avant ou après toutes les étapes de test qui ne sont pas définies pour ignorer les filtres globaux. Vous pouvez définir une étape pour ignorer les filtres globaux dans l'élément Step Information (Informations sur l'étape) de l'étape.

Filtre d'étape

Un filtre d'étape est un filtre défini au niveau de l'étape de test. Il est exécuté avant ou après chaque exécution de l'étape de test.

Vous pouvez ajouter autant de filtres globaux et d'étape que vous avez besoin. Les filtres sont exécutés dans l'ordre selon lequel ils s'affichent dans le scénario de test.

Les rubriques suivantes sont incluses.

Ajout d'un filtre (page 124)
Glisser-déposer d'un filtre (page 141)

Ajout d'un filtre

Les méthodes d'ajout de filtre sont les suivantes :

Ajout d'un filtre manuel (page 126)
Ajout d'un filtre à partir d'une réponse HTTP (page 128)
Ajout d'un filtre à partir d'un ensemble de résultats JDBC (page 131)
Ajout d'un filtre à partir d'un objet Java renvoyé (page 138)

Ajout d'un filtre manuel

Pour ajouter un filtre manuellement, sélectionnez le type de filtre dans une liste et entrez les paramètres pour le filtre.

Vous pouvez ajouter deux types de filtres manuellement : les filtres globaux et les filtres d'étape.

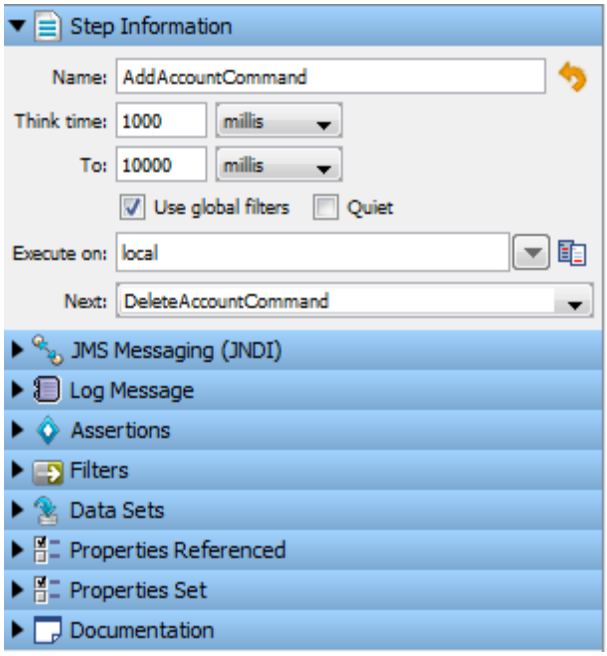
Les filtres globaux s'appliquent et sont exécutés automatiquement pour toutes les étapes du scénario de test, sauf instruction contraire pour une étape.

Un filtre d'étape est applicable uniquement à une étape et est exécuté pour cette étape uniquement.

Pour ajouter un filtre global manuellement :

1. Pour ouvrir le panneau Elements du scénario de test, ouvrez un scénario de test et cliquez n'importe où dans l'éditeur.
2. Dans l'élément Global Filters (Filtres globaux), cliquez sur Add (Ajouter)  pour ajouter un filtre global.
3. Vous êtes invité à sélectionner un type de filtre Integration Support for CAI (Prise en charge de CAI par intégration) ou Integration Support for webMethods Integration Server (Prise en charge de l'intégration avec le serveur d'intégration webMethods).

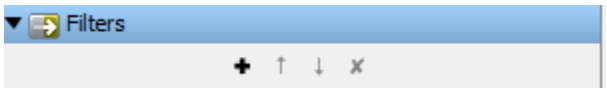
   Pour plus d'informations sur l'ajout de ces types de filtres, consultez les sections Prise en charge de l'intégration pour CAI ou Prise en charge de l'intégration pour le serveur d'intégration webMethods de la rubrique *Utilisation de CA Application Test.*
4. Lorsque vous avez au moins un filtre global dans un scénario de test, par défaut, la case à cocher Use Global Filters (Utiliser des filtres globaux) est sélectionnée pour chaque étape. Si vous ne voulez pas appliquer de filtre global à une étape, désélectionnez la case à cocher.



Pour ajouter un filtre d'étape manuellement :

Procédez comme suit:

1. Sélectionnez l'étape pour laquelle vous voulez appliquer le filtre et dans le panneau droit, cliquez sur l'élément Filter (Filtre).



2. Pour répertorier les filtres disponibles, cliquez sur Add (Ajouter)  dans l'élément de filtre ou cliquez avec le bouton droit de la souris sur l'étape, sélectionnez Add Filter (Ajouter un filtre), puis sélectionnez le filtre approprié pour cette étape.

   Le menu Filter s'ouvre et répertorie les filtres disponibles. Chaque filtre a son propre éditeur et ses paramètres. Pour plus d'informations sur chaque type de filtre, consultez la section Types de filtres de la rubrique *Référence.*

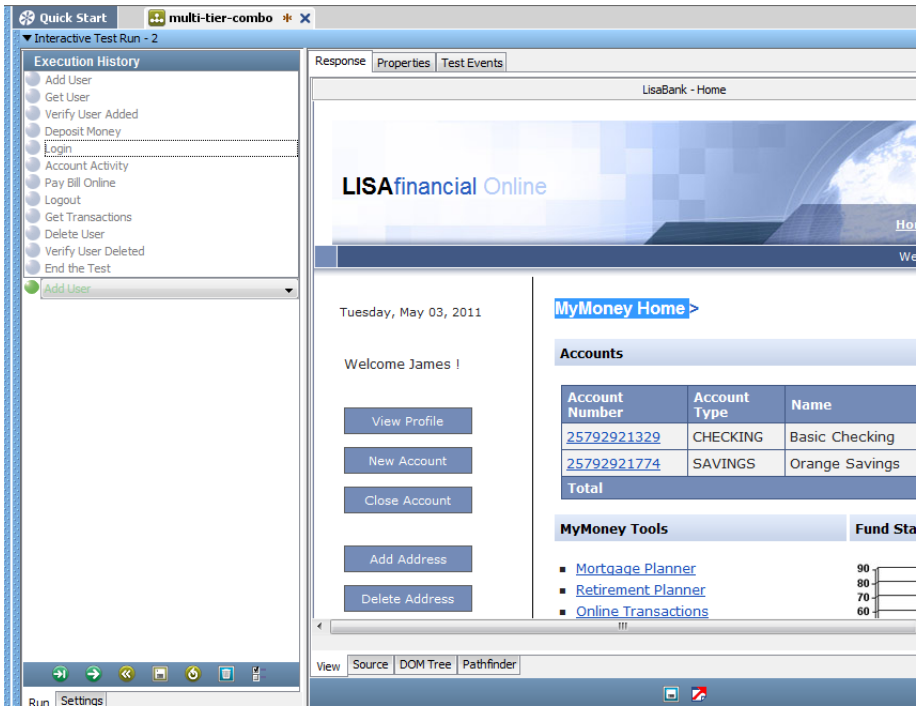
Ajout d'un filtre à partir d'une réponse HTTP

Lorsque vous avez accès à la réponse d'une étape HTTP, vous pouvez l'utiliser pour ajouter un filtre directement.

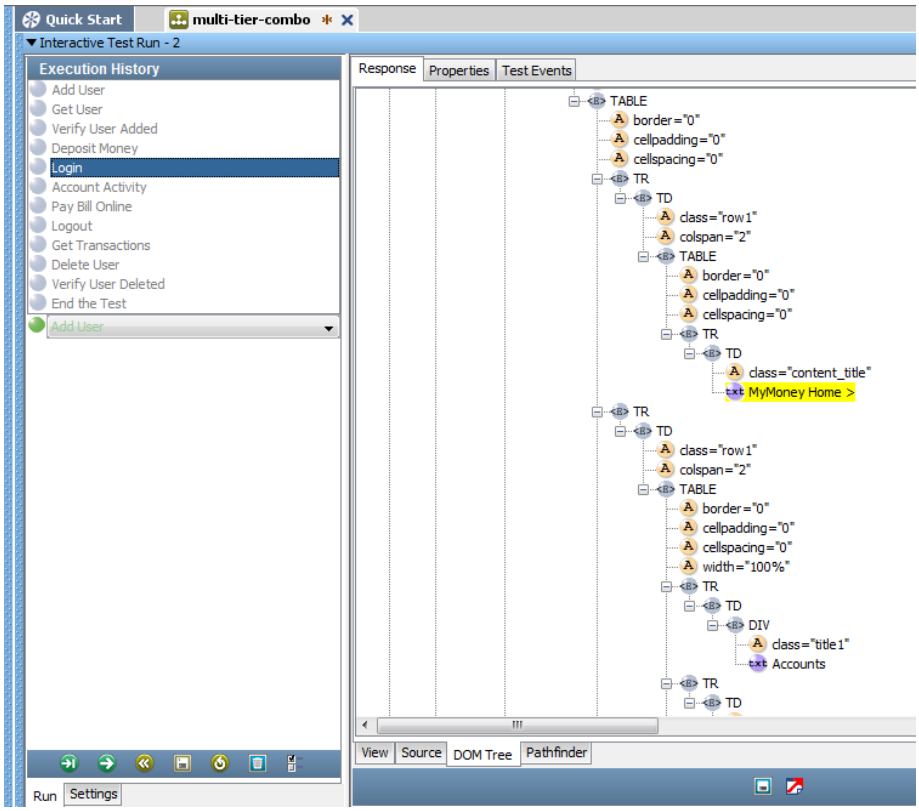
Cet exemple utilise la réponse de l'étape de connexion dans le scénario de test à plusieurs niveaux du répertoire d'exemples. Le but de cet exemple est de capturer le texte où la **page d'accueil de MyMoney** s'affiche actuellement dans la fenêtre. (Il ne s'agit pas toujours du même texte).

Procédez comme suit:

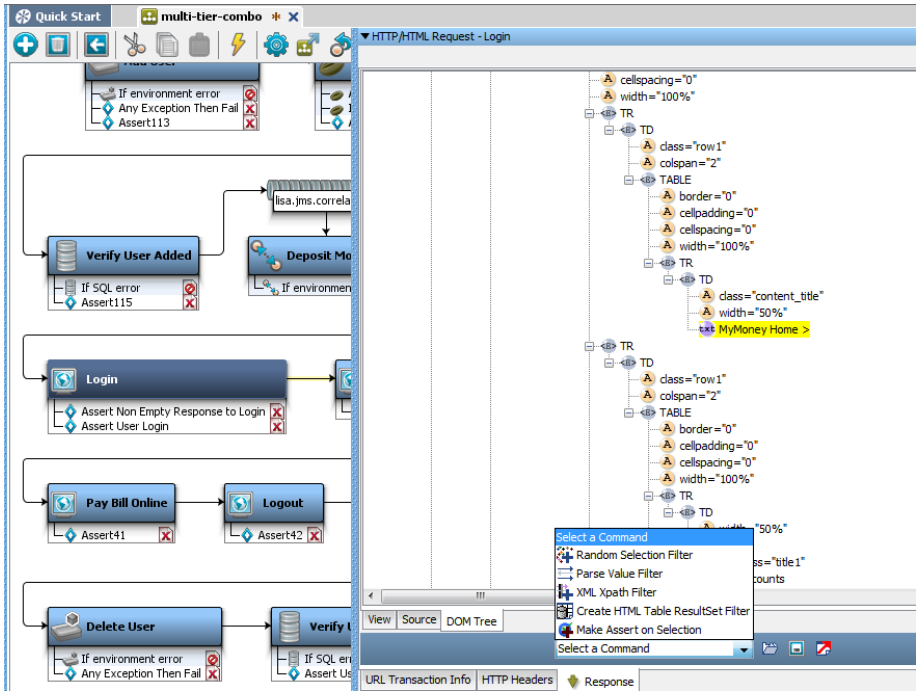
1. Exécutez le scénario de test à plusieurs niveaux dans l'ITR, puis sélectionnez l'étape de test Login (Connexion).



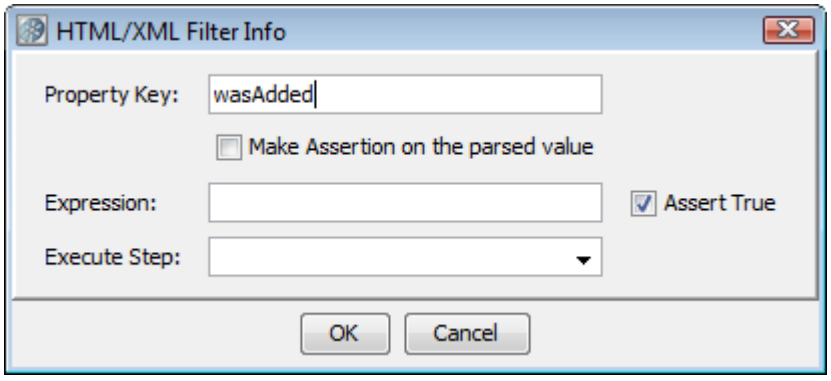
2. Sélectionnez le texte MyMoney Home (page d'accueil de MyMoney) dans l'onglet View (Afficher).
3. Pour confirmer que le texte est sélectionné dans l'arborescence, cliquez sur l'onglet DOM Tree (Arborescence DOM).



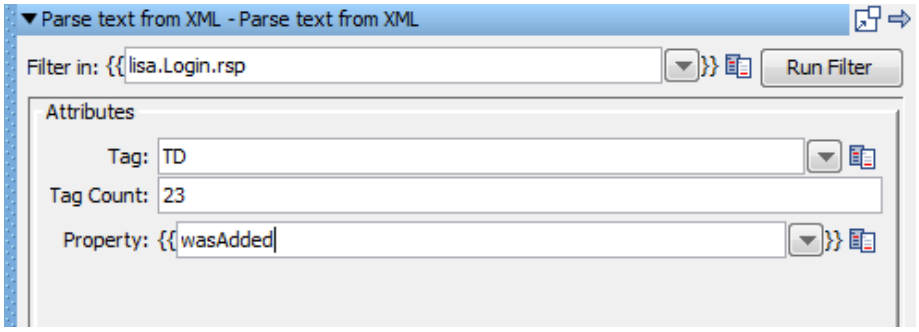
4. Pour appliquer un filtre intégré, double-cliquez sur l'étape de connexion dans l'éditeur de modèles pour ouvrir l'éditeur d'étapes HTTP/HTML.



5. Accédez à l'onglet DOM Tree, recherchez MyMoney Home dans l'arborescence, puis sélectionnez-la.
6. En bas de la fenêtre, dans la liste déroulante de la zone Select a Command (Sélectionner une commande), sélectionnez Parse Value Filter (Analyser le filtre de valeur).
7. Dans la boîte de dialogue qui s'affiche, entrez le nom pour la clé de propriété wasAdded :



8. Cliquez sur OK.



Vous pouvez également ajouter une assertion. Par exemple, vous voulez tester la valeur de la propriété **wasAdded**, afin de déterminer si sa valeur correspond à l'utilisateur ajouté. Pour plus d'informations, consultez la rubrique Ajout d'assertions.

Le filtre qui a été généré peut être affiché en tant que filtre dans l'étape de test de connexion.

Les mêmes fonctionnalités de filtre sont disponibles lorsqu'une réponse HTML est affichée dans l'éditeur d'étapes.

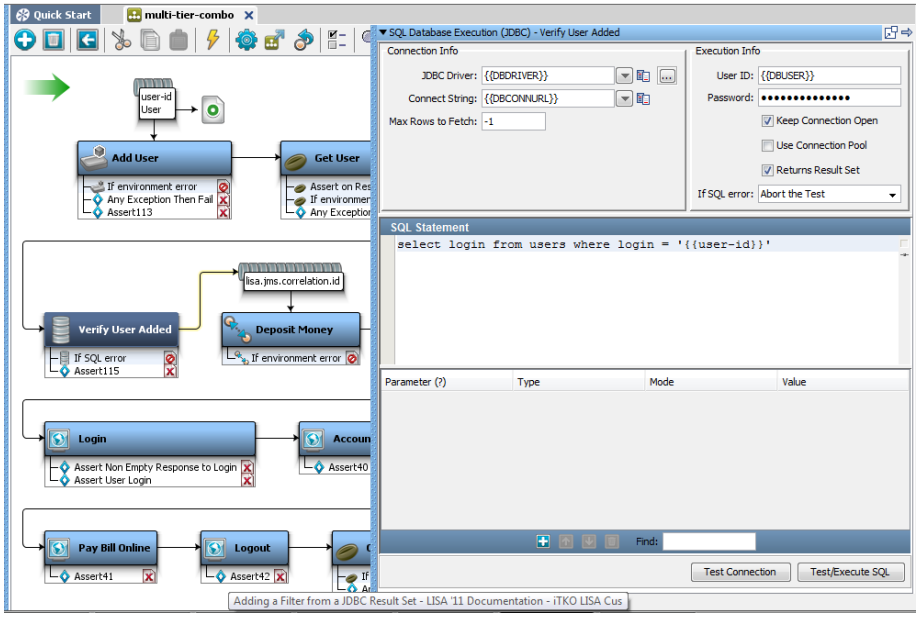
Ajout d'un filtre à partir d'un ensemble de résultats JDBC

Lorsque vous avez accès à la réponse d'un ensemble de résultats d'une étape JDBC, vous pouvez l'utiliser pour ajouter un filtre directement.

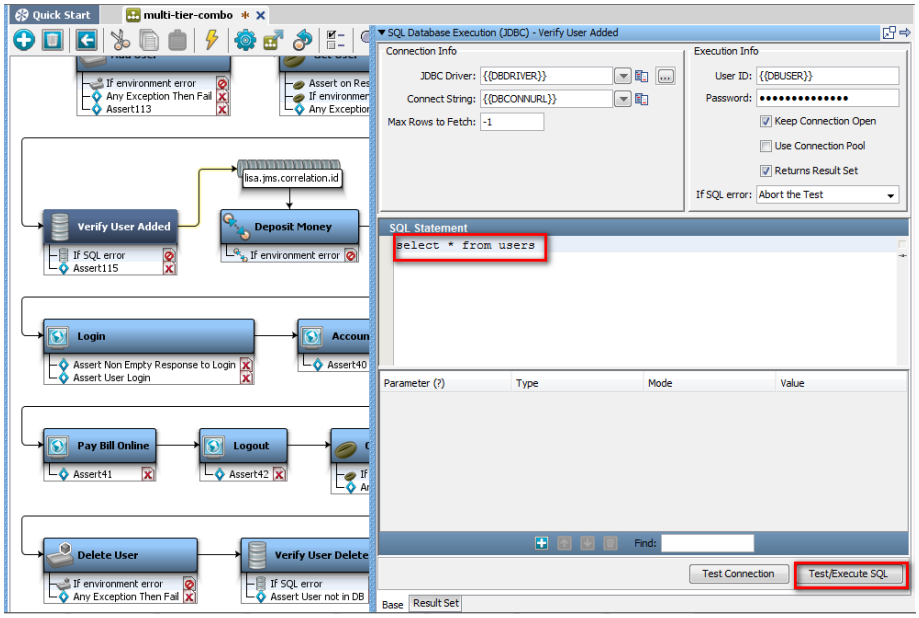
Cet exemple présente la procédure d'ajout d'un filtre à partir de la réponse d'ensemble de résultats JDBC, à l'aide de la réponse de l'étape Verify User Added (Vérifier l'ajout de l'utilisateur) du scénario de test à plusieurs niveaux.

Procédez comme suit:

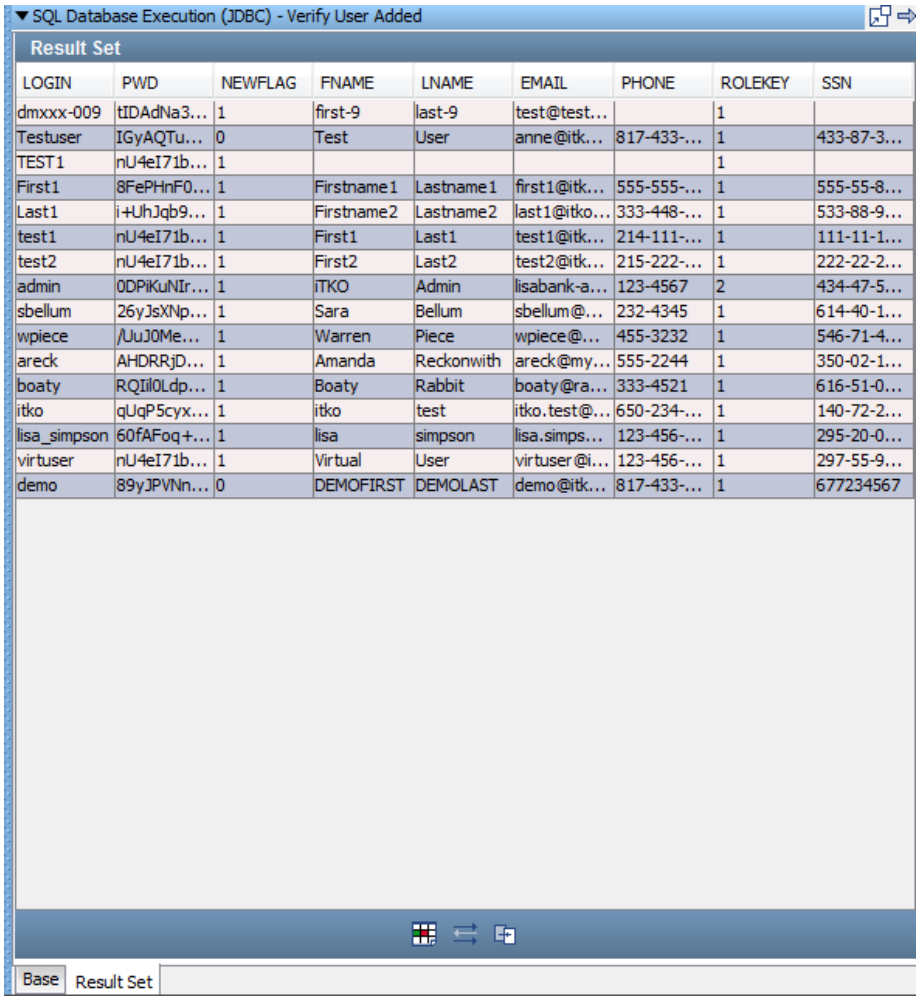
1. Pour ouvrir l'éditeur d'étapes, double-cliquez sur l'étape Verify User Added (Vérifier l'ajout de l'utilisateur).



2. Pour obtenir des valeurs dans l'ensemble de résultats, remplacez l'instruction SQL par **select * from users** et cliquez sur Test/Execute SQL (Tester/exécuter l'expression SQL).



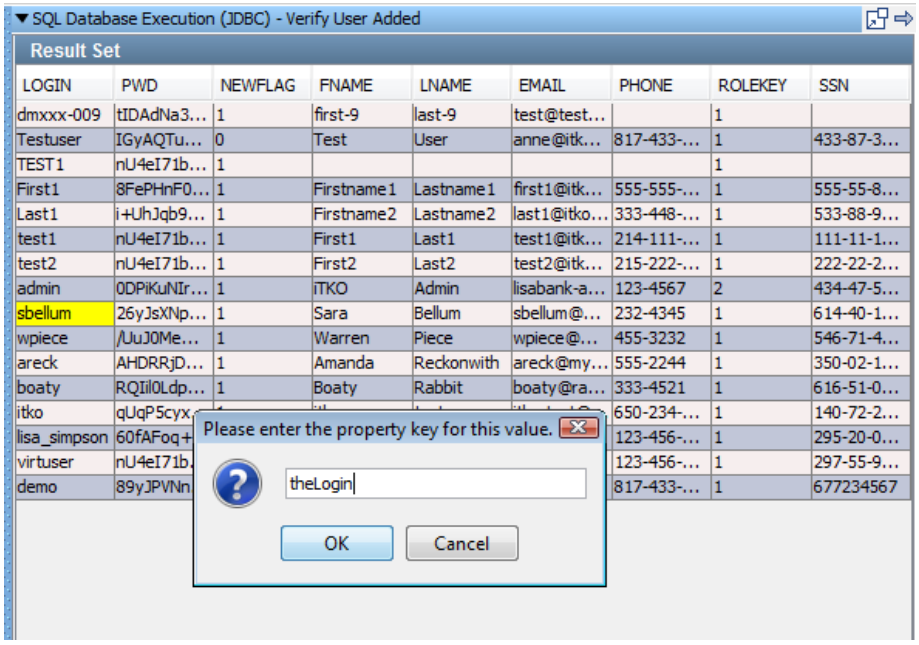
3. Pour obtenir des valeurs dans l'ensemble de résultats, cliquez sur l'onglet Result Set (Ensemble de résultats) et cliquez sur Test/Execute SQL (Tester/exécuter l'expression SQL).



LOGIN	PWD	NEWFLAG	FNAME	LNAME	EMAIL	PHONE	ROLEKEY	SSN
dmxxx-009	tIDAdNa3...	1	first-9	last-9	test@test...		1	
Testuser	IGyAQTu...	0	Test	User	anne@itk...	817-433-...	1	433-87-3...
TEST1	nU4eI71b...	1					1	
First1	8FePHnF0...	1	Firstname1	Lastname1	first1@itk...	555-555-...	1	555-55-8...
Last1	i+UhJqb9...	1	Firstname2	Lastname2	last1@itko...	333-448-...	1	533-88-9...
test1	nU4eI71b...	1	First1	Last1	test1@itk...	214-111-...	1	111-11-1...
test2	nU4eI71b...	1	First2	Last2	test2@itk...	215-222-...	1	222-22-2...
admin	0DPiKuNIr...	1	iTKO	Admin	lisabank-a...	123-4567	2	434-47-5...
sbellum	26yJsXNp...	1	Sara	Bellum	sbellum@...	232-4345	1	614-40-1...
wpiece	/UuJ0Me...	1	Warren	Piece	wpiece@...	455-3232	1	546-71-4...
areck	AHDRRjD...	1	Amanda	Reckonwith	areck@my...	555-2244	1	350-02-1...
boaty	RQIil0Ldp...	1	Boaty	Rabbit	boaty@ra...	333-4521	1	616-51-0...
itko	qUqP5cyx...	1	itko	test	itko.test@...	650-234-...	1	140-72-2...
lisa_simpson	60fAFoq+...	1	lisa	simpson	lisa.simps...	123-456-...	1	295-20-0...
virtuser	nU4eI71b...	1	Virtual	User	virtuser@i...	123-456-...	1	297-55-9...
demo	89yJPVNn...	0	DEMOFIRST	DEMOLAST	demo@itk...	817-433-...	1	677234567

4. Cliquez sur la cellule dans l'onglet Result Set (Ensemble de résultats) qui représente l'emplacement des informations à capturer (sbellum).
5. Cliquez sur Generate Filter for Current Col/Row Value (Générer un filtre pour la valeur de colonne ou de ligne actuelle). 
6. Dans la boîte de dialogue qui s'ouvre, entrez la clé de propriété *theLogin*.

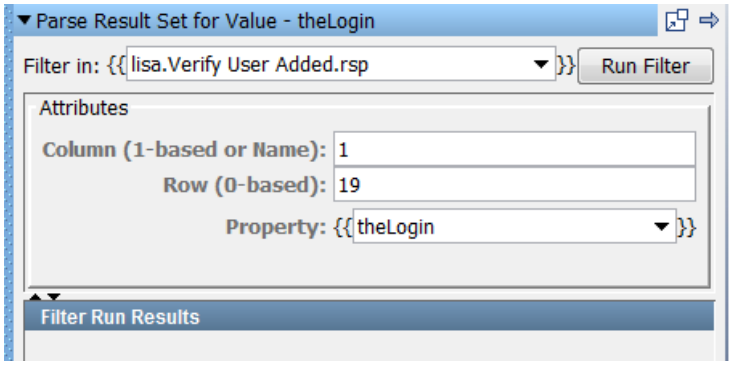
Figure 1: Ajout d'un filtre à partir d'un ensemble de résultats JDBC - Vérification de l'ajout d'un utilisateur - Saisie d'une clé de propriété pour la valeur


SQL Database Execution (JDBC) - Verify User Added
Result Set
LOGIN PWD NEWFLAG FNAME LNAME EMAIL PHONE ROLEKEY SSN
dmxxx-009 tIDAdNa3... 1 first-9 last-9 test@test... 1
Testuser IGyAQTu... 0 Test User anne@itk... 817-433-... 1 433-87-3...
TEST1 nU4eI71b... 1 1
First1 8FePHnF0... 1 Firstname1 Lastname1 first1@itk... 555-555-... 1 555-55-8...
Last1 i+UhJqb9... 1 Firstname2 Lastname2 last1@itko... 333-448-... 1 533-88-9...
test1 nU4eI71b... 1 First1 Last1 test1@itk... 214-111-... 1 111-11-1...
test2 nU4eI71b... 1 First2 Last2 test2@itk... 215-222-... 1 222-22-2...
admin 0DPiKuNIr... 1 iTKO Admin lisabank-a... 123-4567 2 434-47-5...
sbellum 26yJsXNp... 1 Sara Bellum sbellum@... 232-4345 1 614-40-1...
wpiece /UuJ0Me... 1 Warren Piece wpiece@... 455-3232 1 546-71-4...
areck AHDRRjD... 1 Amanda Reckonwith areck@my... 555-2244 1 350-02-1...
boaty RQIil0Ldp... 1 Boaty Rabbit boaty@ra... 333-4521 1 616-51-0...
itko qUqP5cyx... 650-234-... 1 140-72-2...
lisa_simpson 60fAFoq+ 123-456-... 1 295-20-0...
virtuser nU4eI71b 123-456-... 1 297-55-9...
demo 89yJPVNn 817-433-... 1 677234567
Please enter the property key for this value.
theLogin
OK
Cancel

7. Cliquez sur OK.

   DevTest ajoute un filtre appelé **Parse Result Set for Value** (Rechercher la valeur dans l'ensemble de résultats) dans l'étape List Users (Liste des utilisateurs).

8. Pour afficher le filtre, cliquez sur l'éditeur de filtres.



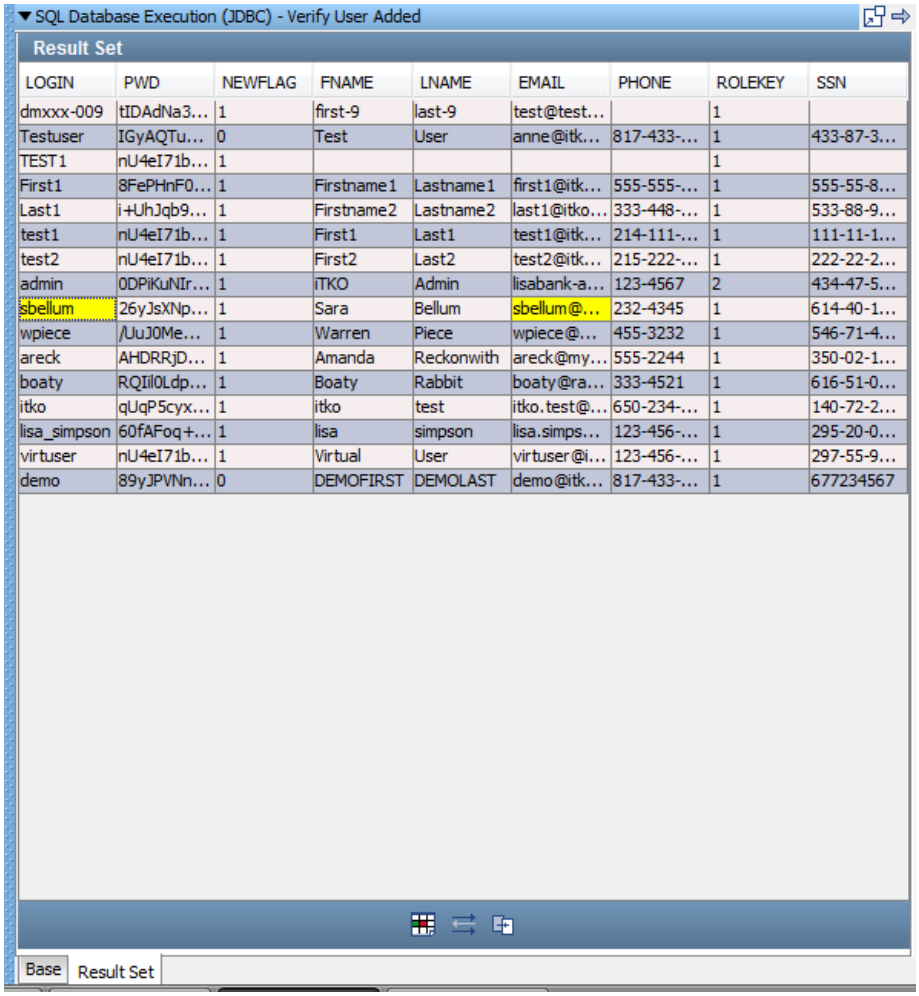
   Dans l'exemple, la valeur de la cellule de la première colonne et de la huitième ligne, **sbellum**, est stockée dans la propriété theLogin.

Application d'un deuxième filtre

Vous pouvez appliquer un deuxième filtre, rechercher une valeur dans une colonne de l'ensemble de résultats, puis capturer une valeur d'une autre colonne dans la même ligne.

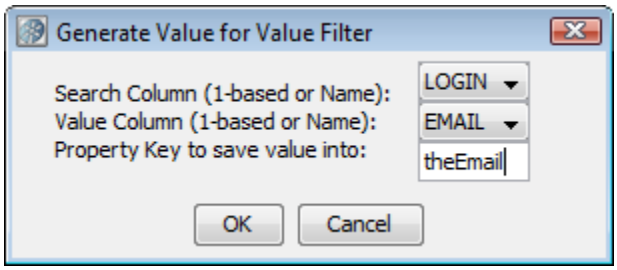
Procédez comme suit:

1. Dans l'ensemble de résultats, sélectionnez deux valeurs dans deux colonnes différentes de la même ligne à l'aide de la touche Ctrl.



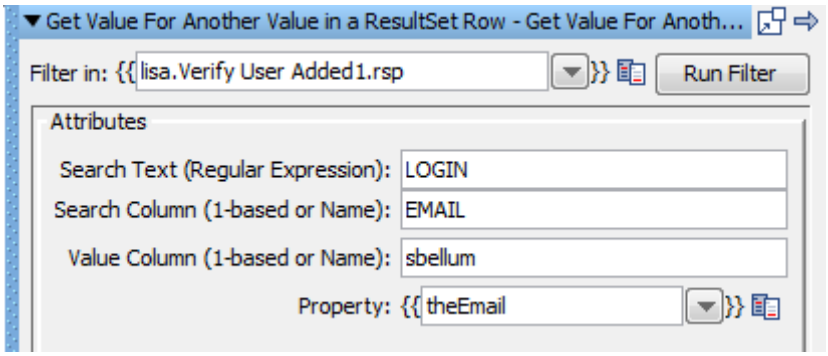
LOGIN	PWD	NEWFLAG	FNAME	LNAME	EMAIL	PHONE	ROLEKEY	SSN
dmxxx-009	tIDAdNa3...	1	first-9	last-9	test@test...		1	
Testuser	IGyAQTu...	0	Test	User	anne@itk...	817-433-...	1	433-87-3...
TEST1	nU4eI71b...	1					1	
First1	8FePHnF0...	1	Firstname1	Lastname1	first1@itk...	555-555-...	1	555-55-8...
Last1	i+UhJqb9...	1	Firstname2	Lastname2	last1@itko...	333-448-...	1	533-88-9...
test1	nU4eI71b...	1	First1	Last1	test1@itk...	214-111-...	1	111-11-1...
test2	nU4eI71b...	1	First2	Last2	test2@itk...	215-222-...	1	222-22-2...
admin	0DPiKuNIr...	1	iTKO	Admin	lisabank-a...	123-4567	2	434-47-5...
sbellum	26yJsXNp...	1	Sara	Bellum	sbellum@...	232-4345	1	614-40-1...
wpiece	/UuJ0Me...	1	Warren	Piece	wpiece@...	455-3232	1	546-71-4...
areck	AHDRRjD...	1	Amanda	Reckonwith	areck@my...	555-2244	1	350-02-1...
boaty	RQIil0Ldp...	1	Boaty	Rabbit	boaty@ra...	333-4521	1	616-51-0...
itko	qUqP5cyx...	1	itko	test	itko.test@...	650-234-...	1	140-72-2...
lisa_simpson	60fAFoq+...	1	lisa	simpson	lisa.simps...	123-456-...	1	295-20-0...
virtuser	nU4eI71b...	1	Virtual	User	virtuser@i...	123-456-...	1	297-55-9...
demo	89yJPVNn...	0	DEMOFIRST	DEMOLAST	demo@itk...	817-433-...	1	677234567

2. Sélectionnez le filtre Filter for a value and then get another column value (Rechercher une valeur à l'aide d'un filtre, puis obtenir une autre valeur de colonne), à l'aide de l'icône  . Pour créer ce filtre, sélectionnez deux cellules dans la même ligne. L'une correspondant à la colonne de recherche et l'autre à la colonne dont vous voulez extraire la valeur.
3. Dans la boîte de dialogue qui s'ouvre, sélectionnez ou réaffectez les colonnes pour la recherche et la valeur.
4. Entrez la clé de propriété *theEmail*.



5. Cliquez sur OK.

   DevTest ajoute le filtre **Get Value For Another Value in a ResultSet Row** (Obtenir la valeur d'une autre valeur dans une ligne d'ensemble de résultats) dans l'étape Verify User Added (Vérifier l'ajout de l'utilisateur).



   Le filtre Value For Another Value in a ResultSet Row (Obtenir la valeur d'une autre valeur dans une ligne d'ensemble de résultats) recherche **sbellum** dans la colonne LOGIN (Connexion). Si le filtre détecte **sbellum**, il stocke la valeur dans la colonne EMAIL (Courriel) de la même ligne d'une propriété nommée **theEmail** (Le courriel).

Remarque : Les mêmes fonctionnalités de filtre sont disponibles lorsqu'un ensemble de résultats JDBC est affiché dans l'éditeur d'étapes.

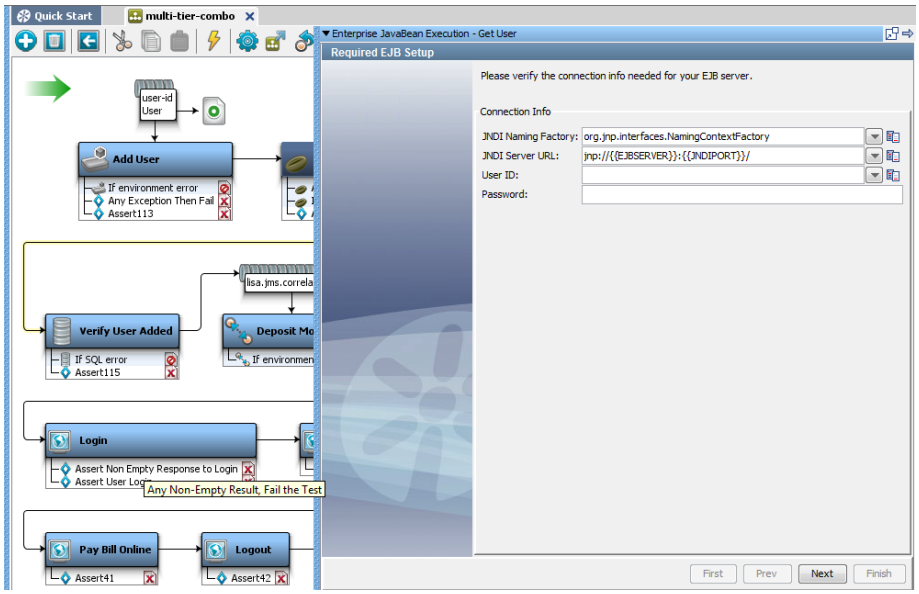
Ajout d'un filtre à partir d'un objet Java renvoyé

Lorsque le résultat de l'étape de test est un objet Java, vous pouvez utiliser le panneau Inline Filter (Filtres intégrés) de l'éditeur Complex Object Editor (page 175) (Editeur d'objets complexes) pour filtrer la valeur renvoyée à partir de l'appel de méthode directement.

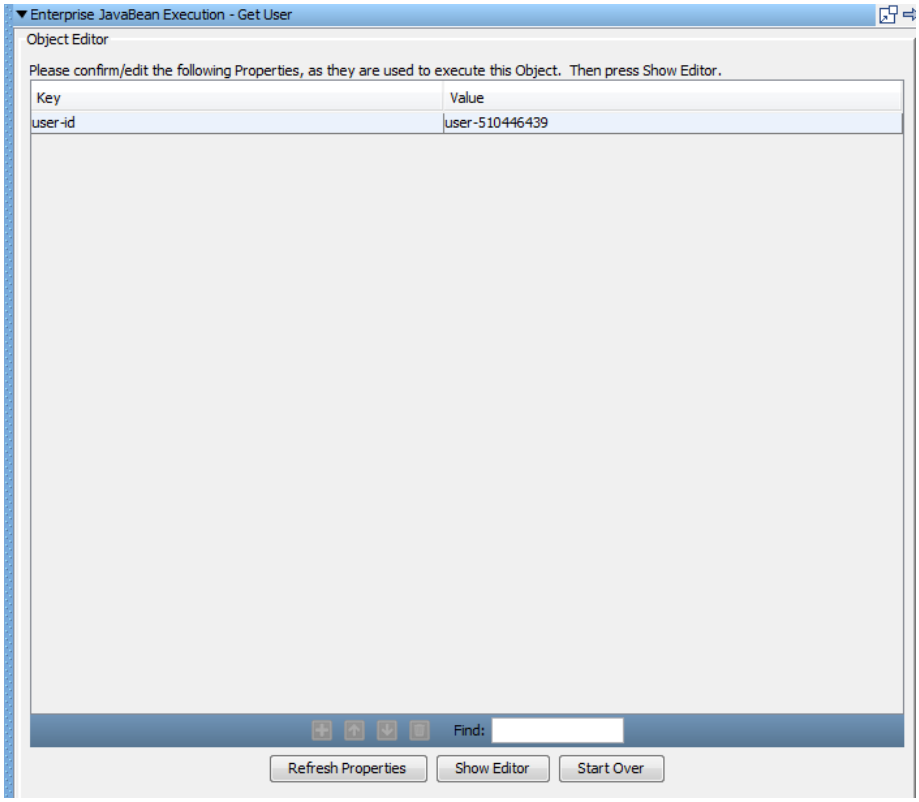
Cet exemple utilise l'étape Get User (Obtenir le nom d'utilisateur, une étape EJB) du scénario de test à plusieurs niveaux disponibles dans le répertoire Examples.

Procédez comme suit:

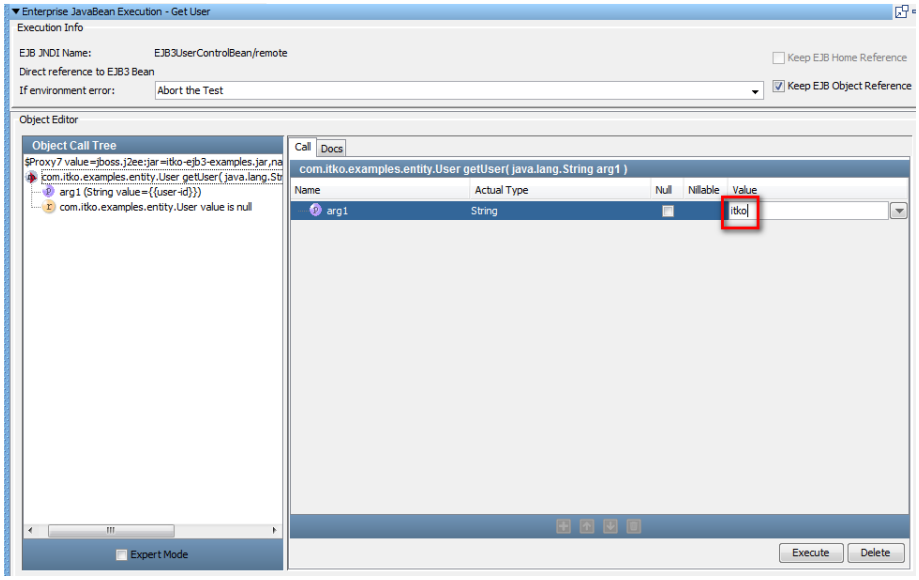
1. Pour ouvrir l'éditeur d'étapes pour l'étape Get User (Obtenir le nom d'utilisateur), double-cliquez sur l'étape.



2. Cliquez sur Next (Suivant). Dans la fenêtre suivante, cliquez sur Finish (Terminer).



3. Pour ouvrir l'arborescence des appels d'objet, cliquez sur Show Editor (Afficher l'éditeur).

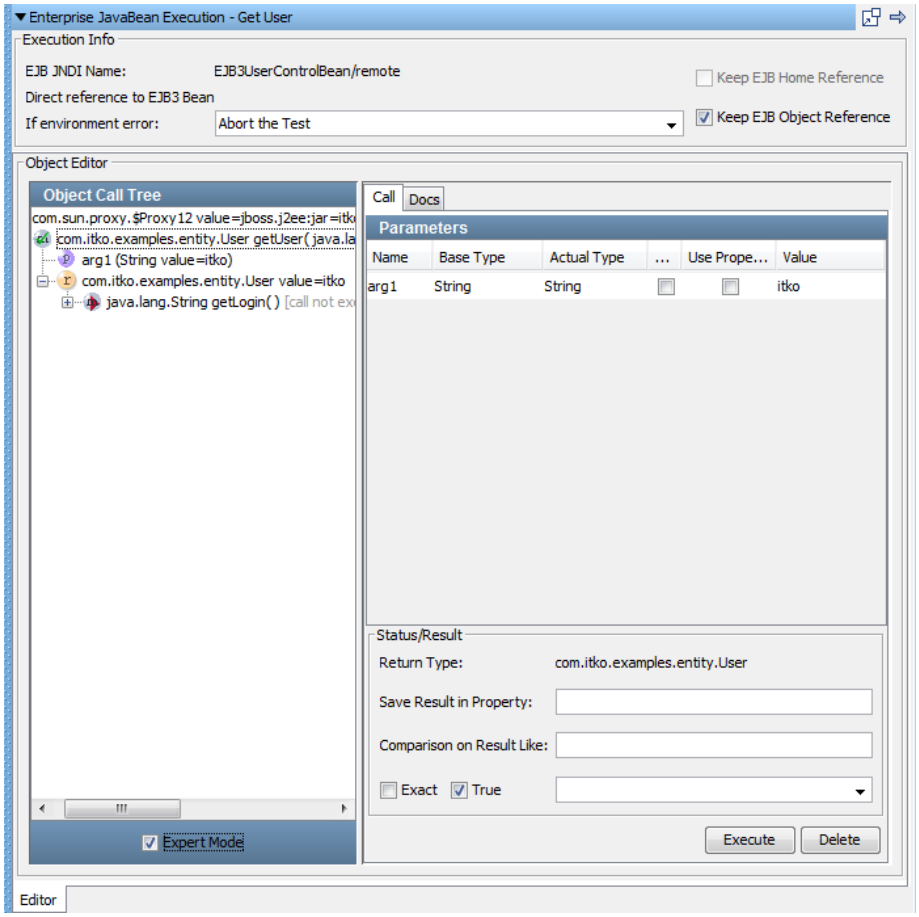


4. Entrez un paramètre d'entrée "itko" dans le champ de valeur.
5. Pour appliquer cette méthode, cliquez sur Execute (Exécuter).

La valeur renvoyée lors de l'exécution de la méthode getLogin est stockée dans la propriété getUserObject. Remarquez que dans ce cas, la valeur renvoyée est un objet de type UserState. Vous pouvez également ajouter une assertion.

6. Cochez la case Expert Mode (Mode Expert).

   L'onglet Call (Appel) affiche les paramètres de filtre.



Vous pouvez également appeler une méthode sur l'objet renvoyé pour obtenir la valeur de connexion pour l'utilisateur et enregistrer la connexion dans une autre propriété.

Les filtres intégrés (et les assertions) ne conduisent pas à l'ajout d'un filtre à l'étape de test dans l'arborescence des éléments. La gestion des filtres intégrés est toujours effectuée dans l'éditeur Complex Object Editor.

Pour plus d'informations sur l'éditeur Complex Object Editor, consultez la section Editeur Complex Object Editor (page 175)(COE, éditeur d'objets complexes) de la rubrique *Utilisation de CA Application Test*.

Glisser-déposer d'un filtre

Vous pouvez effectuer un glisser-déposer des filtres dans l'éditeur de modèles d'une étape de test à une autre.

Procédez comme suit:

1. Cliquez sur le filtre associé à une étape de test ; par exemple, Etape 1.
2. Effectuez un glisser-déposer du filtre sur une autre étape de test dans l'éditeur de modèles ; par exemple, Etape 2.

   Le filtre est appliqué à Etape 2.

Assertions

Une assertion est un élément de code DevTest exécuté après l'exécution d'une étape et de tous ses filtres. Les assertions vérifient que les résultats de l'étape sont conformes aux prévisions.

Le résultat d'une assertion est une valeur booléenne (true ou false).

Le résultat détermine la réussite ou l'échec d'une étape de test, ainsi que l'étape suivante à exécuter dans le scénario de test. Une assertion est utilisée pour altérer de façon dynamique le flux de travaux du scénario de test en introduisant une logique conditionnelle (création de branches) dans le flux de travaux. Le fonctionnement d'une assertion est similaire au bloc de programmation conditionnel If.

Par exemple, vous pouvez créer une assertion pour une étape JDBC qui garantit qu'une seule ligne dans l'ensemble de résultats contient un nom spécifique. Si les résultats de l'étape JDBC contiennent plusieurs lignes, l'assertion modifie l'étape suivante à exécuter. De cette façon, une assertion fournit une fonctionnalité conditionnelle.

Le flux de scénario de test est souvent modélisé sur l'une des deux possibilités suivantes :

- L'étape suivante définie pour chaque étape est l'étape logique suivante dans le scénario de test. Dans ce cas, les assertions pointent vers un échec.
- L'étape suivante est définie pour échouer et toutes les assertions pointent vers l'étape logique suivante.

En général, le choix dépend de la logique employée.

Remarque : Si une assertion référence une propriété non résolue, une erreur de définition de modèle est renvoyée. L'erreur de définition de modèle n'entraîne pas la fin du test, mais avertit l'auteur du test qu'une propriété non résolue a été rencontrée. Lorsqu'une propriété non résolue est rencontrée, l'assertion ne peut pas obtenir de résultat approprié, car elle ne dispose pas d'informations suffisantes. Ce manque d'informations aboutit à des faux positifs ou à des faux négatifs. (La plupart des assertions renvoient la valeur false en cas de détection d'une propriété non résolue, mais ce n'est pas une règle obligatoire.) L'exécution d'un test dans l'ITR et l'examen du panneau Test Events (Evénements de test) à la recherche d'erreurs de définition de modèle vous permettent de déterminer si des propriétés non résolues existent.

Vous pouvez ajouter autant d'assertions que nécessaires, ce qui vous permet de générer un flux de travaux d'une complexité quelconque. Les assertions sont les seuls objets qui peuvent modifier le flux de travaux DevTest.

Remarque : Les assertions sont exécutées dans l'ordre d'affichage et la logique de flux de travaux dépend habituellement de l'ordre d'application des assertions.

Après le déclenchement d'une assertion, vous pouvez configurer l'étape suivante tel que déterminé par l'assertion ; les autres assertions seront ignorées. Lors de chaque évaluation et déclenchement d'une assertion, un événement est généré.

Assertions globales et d'étape

A l'instar des filtres, vous pouvez appliquer des assertions de manière globale. En d'autres termes, les assertions peuvent s'appliquer au cycle de scénario de test complet ou comme assertion d'étape, c'est-à-dire uniquement à une étape spécifique.

Cette section comprend les rubriques suivantes :

Ajout d'une assertion

Plusieurs méthodes s'offrent à vous pour ajouter des assertions dans le scénario de test.

Ajout manuel d'une assertion

Pour ajouter une assertion manuellement, sélectionnez le type d'assertion dans une liste et entrez ses paramètres. Les assertions manuelles sont de deux types :

- Les assertions globales sont définies au niveau du scénario de test. Une assertion globale s'applique à toutes les étapes du scénario de test. Elle est automatiquement exécutée pour chaque étape, sauf si un noeud indique le cas contraire.
- Les assertions d'étape sont définies au niveau de l'étape de test. Ce type d'assertion est applicable uniquement à une étape et est exécuté pour cette étape uniquement.

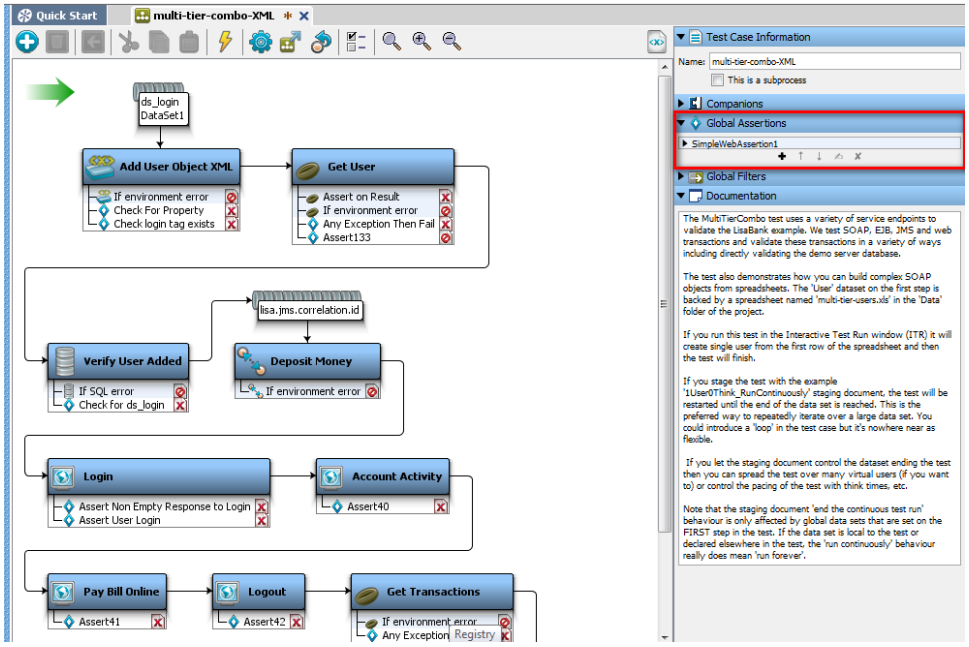
Pour ajouter une assertion globale :

1. Ouvrez un scénario de test et, dans le panneau droit, cliquez sur l'élément Global Assertions (Assertions globales).

Vous pouvez appliquer les types d'assertions globales suivants :

- **HTTP**
  - Assertion Simple Web Assertion (Assertion Web simple)
  - Assertion Check Links on Web Responses (Vérifier les liens dans les réponses Web)
- **XML**
  - Assertion Ensure Step Response Time (Vérifier le temps de réponse d'étape)
- **Autre**
  - Ensure Result Contains Expression (Vérifier que le résultat contient une expression)
  - Assertion Ensure Step Response Time (Vérifier le temps de réponse d'étape)
  - Scan a File for Content (Analyser le contenu d'un fichier)

Le graphique suivant illustre une assertion globale appliquée au scénario de test à plusieurs niveaux.

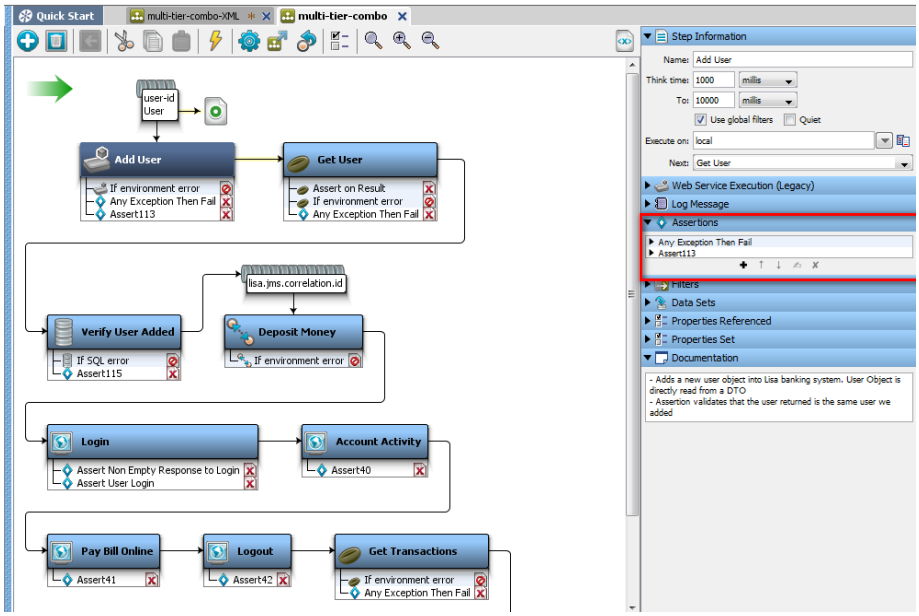


Pour ajouter une assertion d'étape :

1. Effectuez l'une des opérations suivantes :

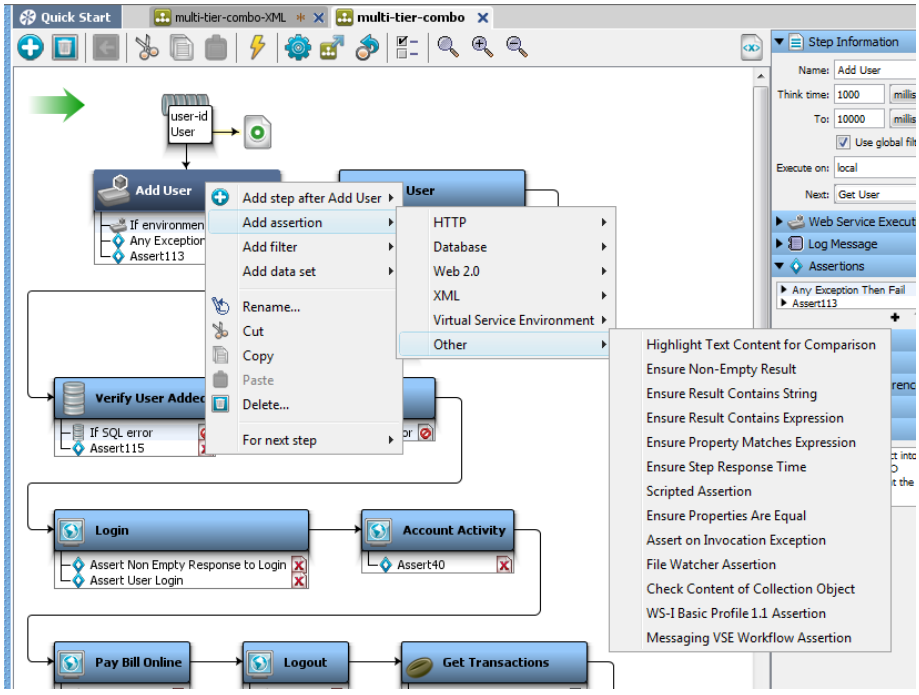
   - Sélectionnez l'étape pour laquelle vous voulez appliquer l'assertion et dans le panneau droit, cliquez sur l'élément Assertion.
   - Cliquez avec le bouton droit de la souris sur l'étape, sélectionnez Add Assertion (Ajouter une assertion) et sélectionnez l'assertion appropriée pour l'étape.

   Le graphique suivant illustre des assertions d'étape appliquées à l'étape Add User (Ajouter un utilisateur) dans le scénario de test à plusieurs niveaux.



2. Pour ajouter une assertion, cliquez sur Add (Ajouter) dans la barre d'outils d'assertion.

   Vous pouvez également cliquer avec le bouton droit de la souris sur une étape dans l'éditeur de modèles pour ajouter une assertion. Le panneau Assertion s'ouvre et affiche un menu d'assertions que vous pouvez appliquer à l'étape.



Pour sélectionner et modifier une assertion :

1. Effectuez l'une des opérations suivantes :
   - Cliquez sur l'étape à laquelle l'assertion est appliquée.
   - Cliquez sur l'assertion liée à cette étape dans l'onglet Assertion.
2. Pour ouvrir l'éditeur d'assertions, double-cliquez sur l'assertion. L'éditeur est propre à chaque type d'assertion.

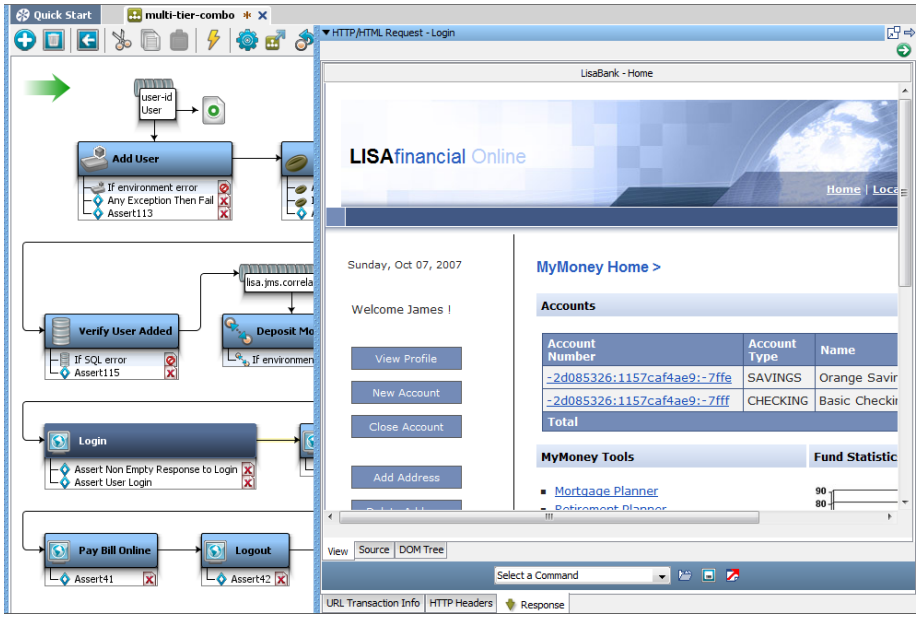
Ajout d'une assertion à partir d'une réponse HTTP

Lorsque vous avez accès à la réponse d'une étape HTTP, vous pouvez l'utiliser pour ajouter une assertion directement.

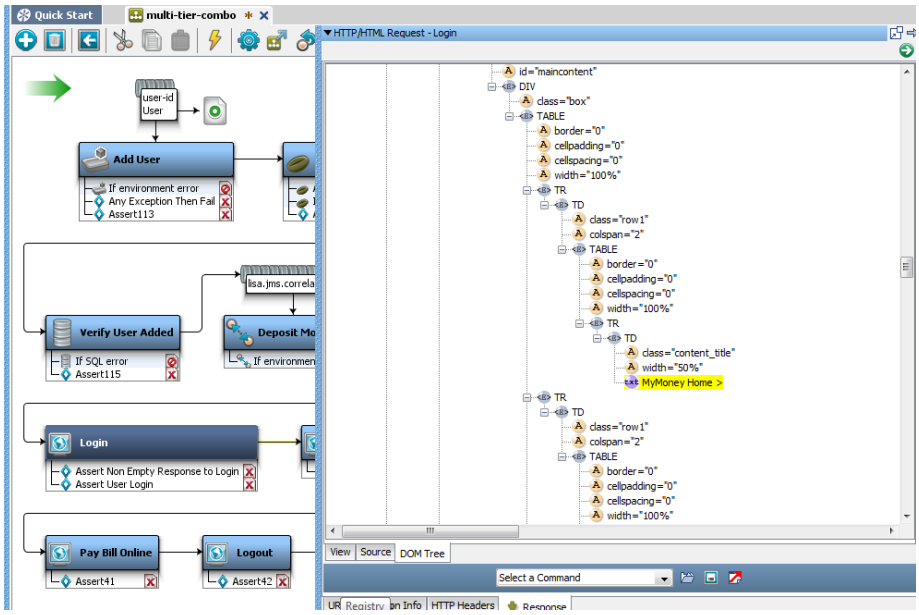
Cet exemple de réponse HTTP/HTML utilise l'étape Login (Connexion) dans le scénario de test à plusieurs niveaux. L'objectif de cet exemple est de tester si le texte MyMoney Home (page d'accueil de MyMoney) s'affiche dans la réponse.

Procédez comme suit:

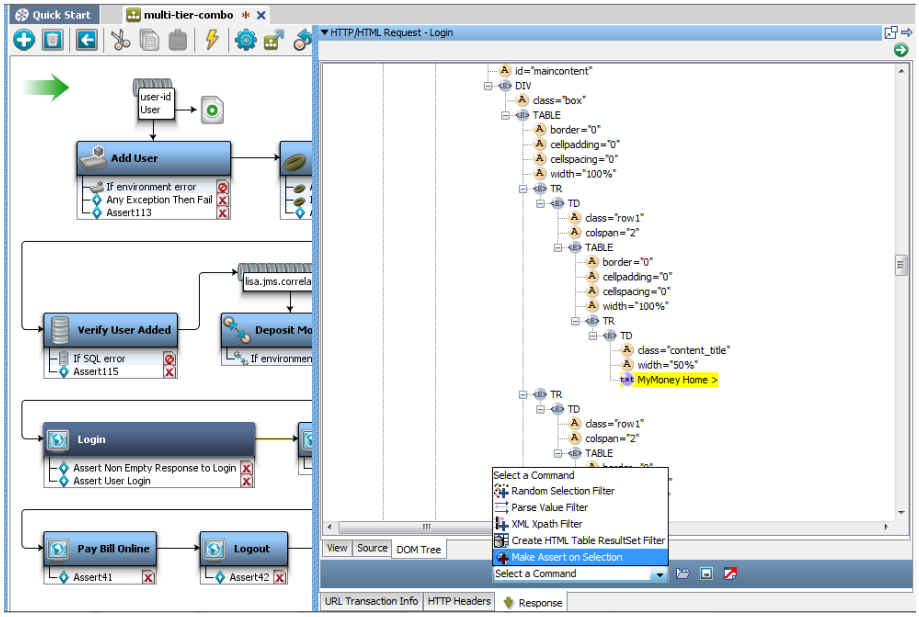
1. Exécutez le scénario de test à plusieurs niveaux dans l'ITR.
2. Double-cliquez sur l'étape de connexion dans l'éditeur de modèles.



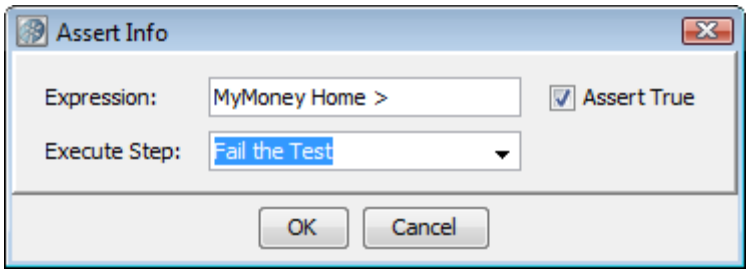
3. Sélectionnez le texte MyMoney Home (page d'accueil de MyMoney) dans l'onglet View (Afficher).
4. Pour vérifier que le texte est sélectionné dans l'arborescence, cliquez sur l'onglet DOM Tree (Arborescence DOM).



5. Dans le menu déroulant Select a command (Sélectionner une commande) au bas du panneau, sélectionnez Make Assert on Selection (Créer une assertion lors de la sélection).



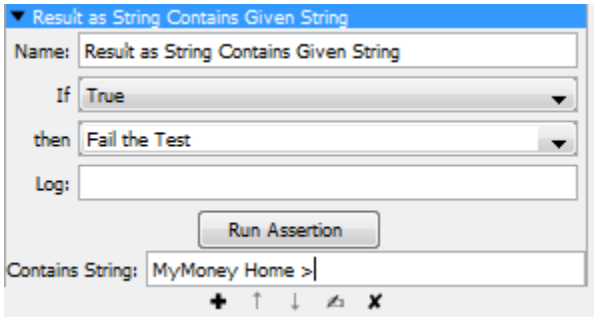
6. Dans la fenêtre qui est affichée, entrez l'expression à laquelle le texte sélectionné doit correspondre, puis sélectionnez le comportement d'assertion approprié.



Dans cet exemple, l'assertion se déclenche si le texte MyMoney Home n'est pas présent et renvoie à l'étape d'échec.

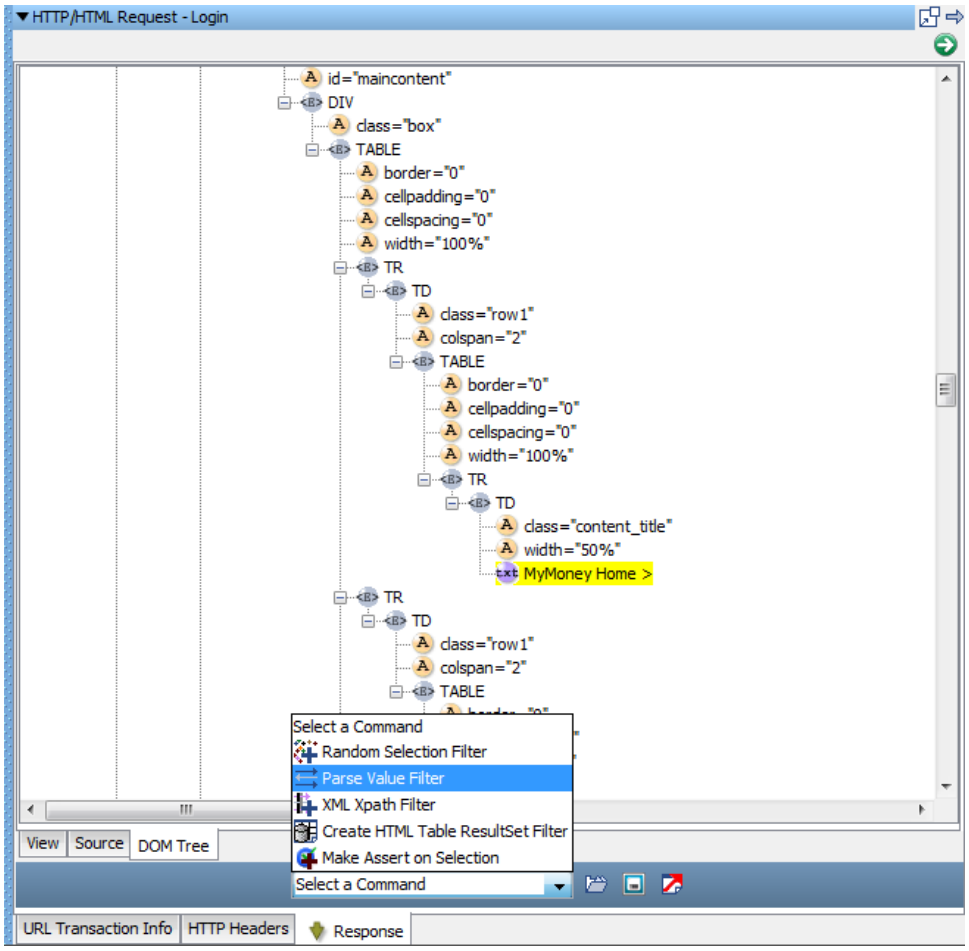
7. Cliquez sur OK pour enregistrer l'assertion.

   L'assertion qui a été générée peut être affichée en tant qu'assertion dans l'étape de connexion.

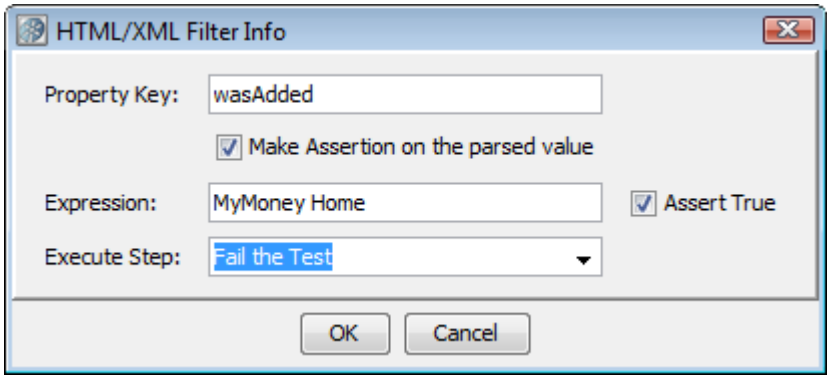


Exécution d'un filtre et d'une assertion

Si vous voulez qu'un filtre capture la valeur MyMoney Home, puis l'exécuter en tant qu'assertion, utilisez le filtre Parse Value Filter (Analyser le filtre de valeur) qui permet d'effectuer les deux opérations.



La fenêtre affichée par le filtre Parse Value Filter indique que la valeur Property Key (Clé de la propriété) est le filtre à appliquer et Expression est l'assertion à déclencher.



En conséquence, un filtre et une assertion sont ajoutés à l'étape de connexion et sont affichés dans l'éditeur de modèles.

Remarque : Les mêmes fonctionnalités d'assertion sont disponibles lorsqu'une réponse HTML est affichée dans l'éditeur d'étapes.

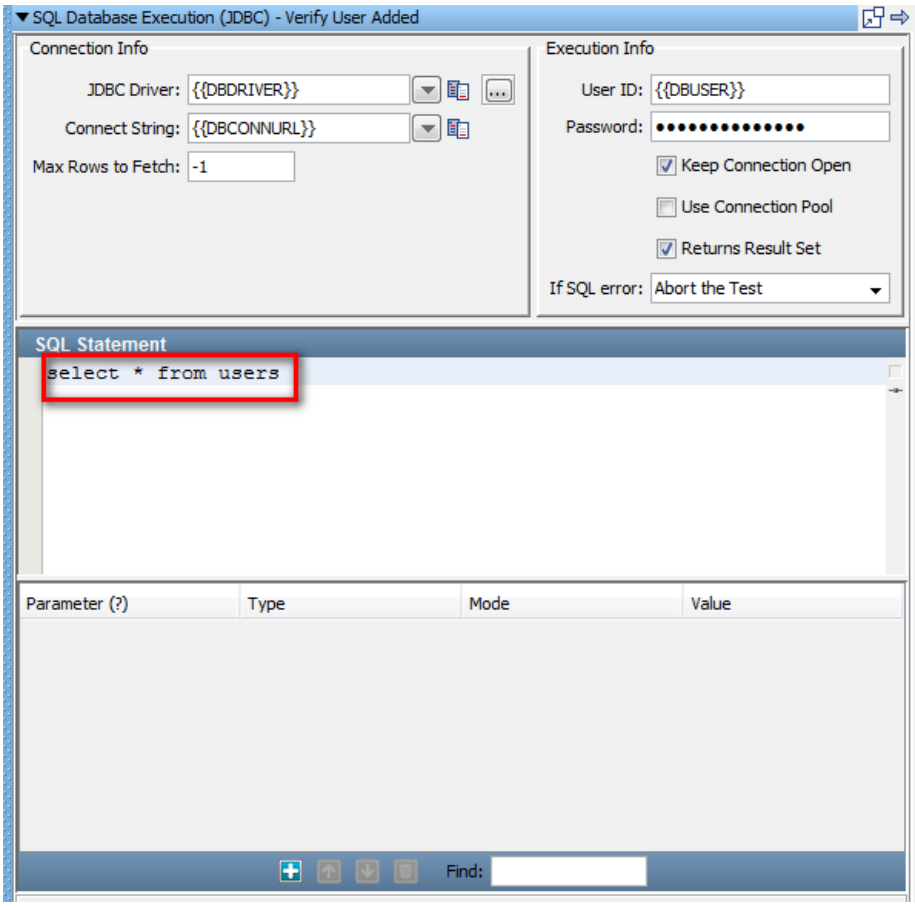
Ajout d'une assertion à partir d'un ensemble de résultats JDBC

Lorsque vous avez accès à la réponse d'un ensemble de résultats d'une étape JDBC, vous pouvez l'utiliser pour ajouter une assertion directement. L'exemple suivant illustre cette méthode d'ajout d'assertion.

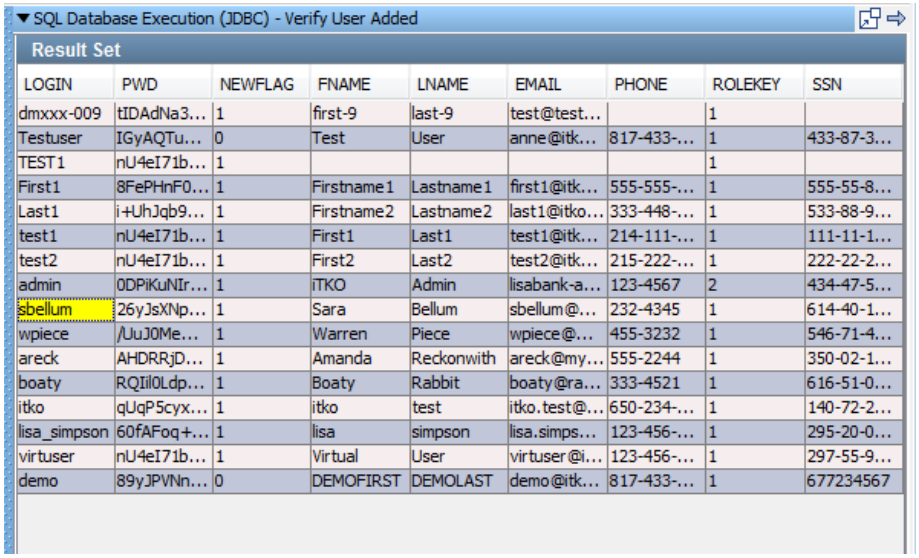
Cet exemple repose sur une réponse d'ensemble de résultats, utilisant la réponse de l'étape Verify User Added (Vérifier l'ajout de l'utilisateur) du scénario de test à plusieurs niveaux (multi-tier-combo.tst) dans le répertoire d'exemples.

Procédez comme suit:

1. Sélectionnez l'étape Verify User Added, puis double-cliquez pour ouvrir sa fenêtre d'éditeur. Remplacez l'instruction SQL par **select * from users**.



2. Pour exécuter la requête, cliquez sur le bouton Test/Execute SQL (Tester/Exécuter l'expression SQL).
3. Sélectionnez l'onglet Result Set (Ensemble de résultats), puis cliquez sur la cellule dans l'ensemble de résultats qui décrit les informations à analyser (par exemple, **sbellum**).

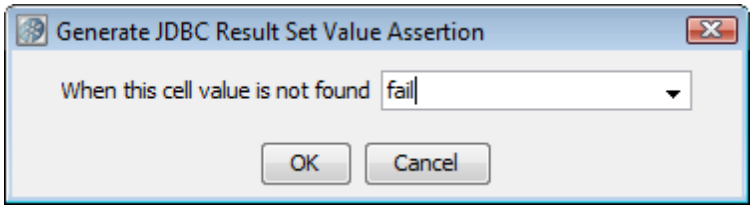


LOGIN	PWD	NEWFLAG	FNAME	LNAME	EMAIL	PHONE	ROLEKEY	SSN
dmxxx-009	tIDAdNa3...	1	first-9	last-9	test@test...		1	
Testuser	IGyAQTu...	0	Test	User	anne@itk...	817-433-...	1	433-87-3...
TEST1	nU4eI71b...	1					1	
First1	8FePHnF0...	1	Firstname1	Lastname1	first1@itk...	555-555-...	1	555-55-8...
Last1	i+UhJqb9...	1	Firstname2	Lastname2	last1@itko...	333-448-...	1	533-88-9...
test1	nU4eI71b...	1	First1	Last1	test1@itk...	214-111-...	1	111-11-1...
test2	nU4eI71b...	1	First2	Last2	test2@itk...	215-222-...	1	222-22-2...
admin	0DPiKuNIr...	1	iTKO	Admin	lisabank-a...	123-4567	2	434-47-5...
sbellum	26yJsXNp...	1	Sara	Bellum	sbellum@...	232-4345	1	614-40-1...
wpiece	/UuJ0Me...	1	Warren	Piece	wpiece@...	455-3232	1	546-71-4...
areck	AHDRRjD...	1	Amanda	Reckonwith	areck@my...	555-2244	1	350-02-1...
boaty	RQIil0Ldp...	1	Boaty	Rabbit	boaty@ra...	333-4521	1	616-51-0...
itko	qUqP5cyx...	1	itko	test	itko.test@...	650-234-...	1	140-72-2...
lisa_simpson	60fAFoq+...	1	lisa	simpson	lisa.simps...	123-456-...	1	295-20-0...
virtuser	nU4eI71b...	1	Virtual	User	virtuser@i...	123-456-...	1	297-55-9...
demo	89yJPVNn...	0	DEMOFIRST	DEMOLAST	demo@itk...	817-433-...	1	677234567

4. Dans la barre d'outils sous la fenêtre Result Set, cliquez sur  Generate Assertion for the Value of a Cell (Générer une assertion pour la valeur d'une cellule).

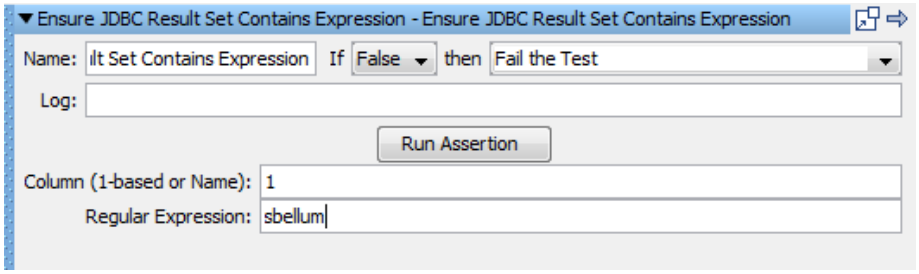
   L'objectif du test est de vérifier que **sbellum** s'affiche dans une cellule de la colonne LOGIN.

5. Dans la boîte de dialogue Generate JDBC Result Set Value Assertion (Générer une assertion de valeur d'ensemble de résultats JDBC), saisissez l'étape de test (échec) de redirection en cas de non-détection de la valeur :



   DevTest crée une assertion nommée Ensure JDBC Result Set Contains Expression (Vérifier que l'ensemble de résultats JDBC contient une expression) dans l'étape Verify User Added (Vérifier l'ajout d'utilisateur).

Equation 1: Ajout d'une assertion à partir d'un ensemble de résultats JDBC indiquant l'assertion ajoutée



Remarque : Les mêmes fonctionnalités d'assertion sont disponibles lorsqu'un ensemble de résultats JDBC est affiché dans l'éditeur d'étapes.

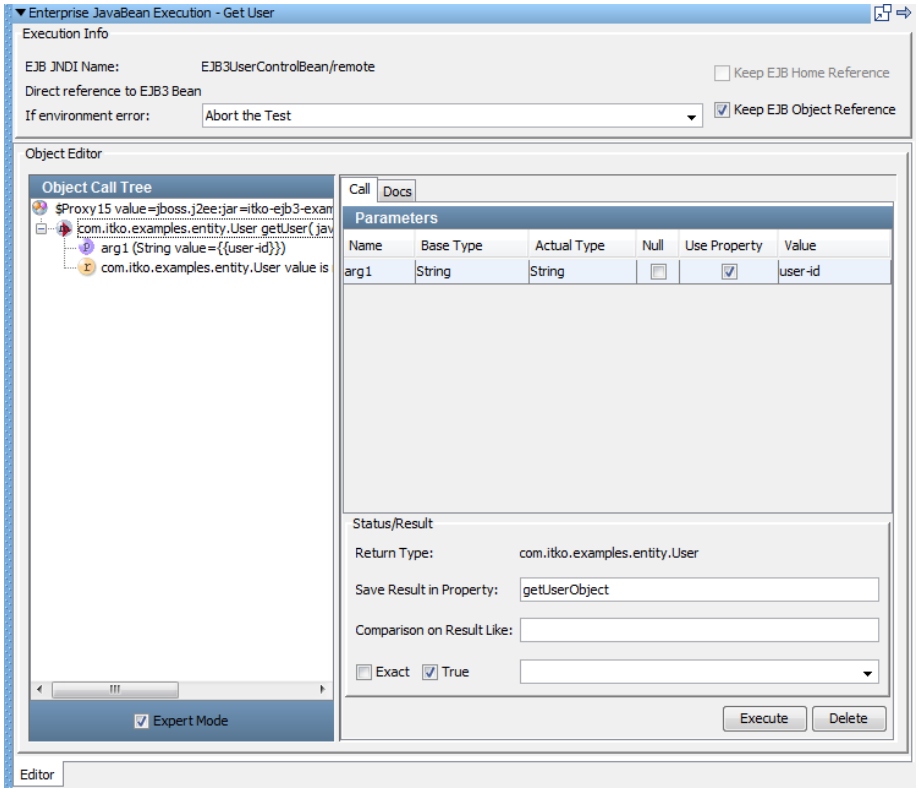
Ajout d'une assertion pour un objet Java renvoyé

Lorsque l'étape de test renvoie un objet Java, utilisez le panneau d'assertion en ligne Complex Object Editor (Editeur d'objets complexes) pour ajouter une assertion sur la valeur renvoyée directement à partir de l'appel de la méthode. L'exemple suivant illustre cette méthode d'ajout d'assertion.

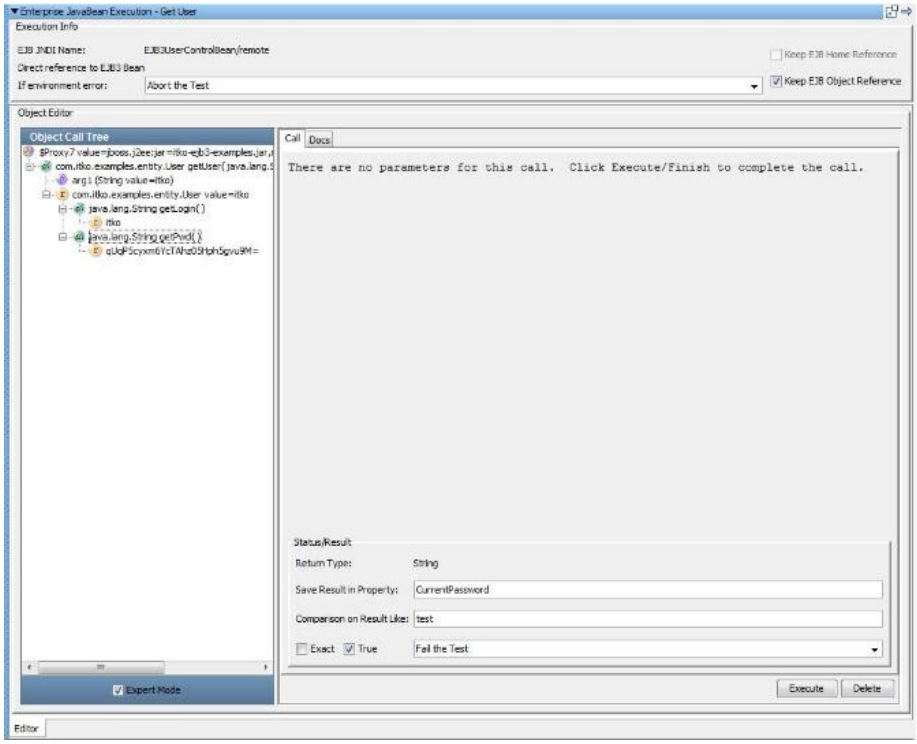
Cet exemple utilise l'étape Get User (Obtenir le nom d'utilisateur, une étape EJB) du scénario de test à plusieurs niveaux se trouvant dans le répertoire d'exemples.

Procédez comme suit:

1. Entrez le paramètre d'entrée **itko**, puis exécutez l'appel de méthode getUser. Exécutez l'appel getPwd sur l'objet UserState qui a été renvoyé à partir de cet appel.



2. Pour ouvrir le volet Status/Result (Statut/résultat) dans lequel vous pouvez ajouter l'assertion, cochez la case Expert Mode (Mode Expert) dans le volet gauche.



La valeur renvoyée lors de l'exécution de la méthode getPwd est stockée dans la propriété CurrentPassword.

3. Ajoutez une assertion qui teste si la valeur renvoyée est égale à la chaîne test. Si ce n'est pas le cas, elle redirige vers l'étape Fail (Echec).
4. Pour exécuter cette étape, cliquez sur Execute (Exécuter).

Remarque : Les assertions (et les filtres) intégrées ne sont pas ajoutées à l'étape de test. La gestion des assertions intégrées est toujours effectuée dans le Complex Object Editor (Editeur d'objets complexes).

Pour plus d'informations sur l'éditeur Complex Object Editor, consultez la section Editeur Complex Object Editor (COE, éditeur d'objets complexes) (page 175) de la rubrique *Utilisation de CA Application Test.*

Ajout d'une assertion à une étape de test d'applications mobiles

Vous pouvez appliquer des assertions à :

- Une étape de test d'applications mobiles
- Chaque action contenue dans une étape de test

  Remarque : Une étape de test contient autant de gestes (appuyer, glisser) que nécessaires pour terminer l'étape. Ces sous-étapes, ou actions, s'affiche dans la table Actions lorsque vous double-cliquez sur une étape de test.

Procédez comme suit:

1. Ouvrez le scénario de test d'applications mobiles à mettre à jour.
2. Pour afficher les détails de chaque étape :
   a. Cliquez sur l'étape de test à vérifier.
   b. Cliquez sur l'étape Mobile testing step (Etape de test d'application mobile) dans l'arborescence d'éléments à droite pour afficher les détails de l'étape. Vous pouvez également double-cliquer sur l'étape de test.

      L'onglet Mobile Testing Step (Etape de test d'application mobile) s'ouvre et affiche une capture d'écran de l'application de test. La section Actions, en haut de l'onglet, indique les actions effectuées dans l'étape de test.
   c. Pour afficher la capture d'écran associée à une action, cliquez sur l'action dans la section Actions.
3. Cliquez avec le bouton droit de la souris sur la capture d'écran ou un élément de fenêtre spécifique dans la partie inférieure de l'onglet.
4. Cliquez sur Add Assertion (Ajouter l'assertion), puis sélectionnez l'assertion à ajouter.

   L'assertion sélectionnée est ajoutée à l'étape de test.
5. Cliquez sur Save (Enregistrer).

Informations complémentaires :

Ajout d'une assertion (page 143)

Sélection et modification d'une assertion

Procédez comme suit:

1. Fermez l'éditeur d'étapes de test d'applications mobiles.
2. Pour afficher les assertions de l'étape de test sélectionnée, cliquez sur Assertions dans l'arborescence d'éléments à droite.

   L'onglet Assertions se développe.
3. Double-cliquez sur l'assertion à modifier.

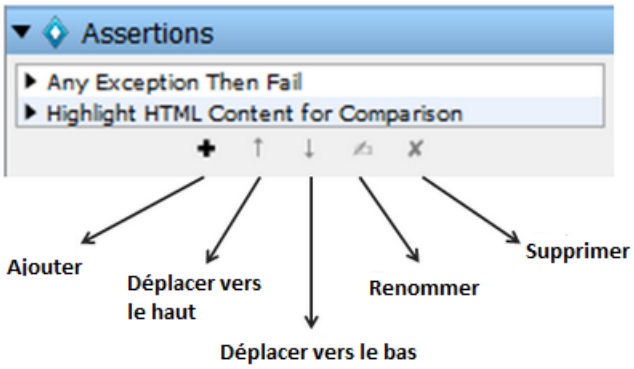
   L'éditeur d'assertions s'ouvre.
4. Remplissez les champs pour l'assertion sélectionnée.

   L'éditeur est propre à chaque type d'assertion. Pour plus d'informations sur les types d'assertion, consultez les rubriques suivantes :

   - Application d'une fenêtre mobile identique
   - Vérification de la correspondance de l'élément de fenêtre à l'expression
   - Vérification de la suppression d'élément de fenêtre
5. Cliquez sur Save (Enregistrer).

Barre d'outils Assertions

Tous les éléments ont leur propre barre d'outils permettant de les ajouter, de les supprimer ou de les réorganiser au bas de l'élément.



Réorganiser une assertion

Vous devrez peut-être réorganiser les assertions, car elles sont évaluées dans l'ordre dans lequel elles s'affichent. Modifier l'ordre des assertions peut affecter le flux de travaux.

Pour réorganiser une assertion, procédez comme suit :

- Sélectionnez l'assertion dans l'onglet Elements (Elements) et cliquez sur Move Up (Déplacer vers le haut) ou Move Down (Déplacer vers le bas) la barre d'outils.
- Effectuez un glisser-déposer de l'assertion dans l'éditeur de modèles vers la destination cible.

Changement de nom d'une assertion

Procédez comme suit:

- Sélectionnez l'assertion et cliquez-la avec le bouton droit de la souris pour ouvrir un menu. Pour renommer l'assertion, cliquez sur Rename (Renommer).
- Sélectionnez l'assertion et cliquez sur l'icône Rename dans la barre d'outils.
- Pour changer le nom de l'assertion sur le nom d'assertion par défaut, sélectionnez l'assertion et cliquez sur Revert to Default Name (Rétablir le nom par défaut).

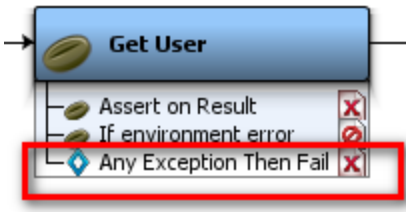
Glisser-Déposer d'une assertion

Pour effectuer un glisser-déposer des assertions dans l'éditeur de modèles d'une étape de test à une autre :

1. Cliquez sur l'assertion dans une étape de test ; par exemple, Etape 1.
2. Sélectionnez l'assertion et faites-la glisser vers une autre étape de test dans l'éditeur de modèles ; par exemple, Etape 2.
3. L'assertion est appliquée à l'étape 2.

Configuration de l'étape suivante d'une assertion

Les assertions ajoutées à une étape peuvent être affichées dans l'éditeur de modèles.



Une fois que l'assertion est ajoutée à une étape, vous pouvez sélectionner l'étape suivante à exécuter, si vous voulez que le flux de travaux soit modifié.

Pour configurer l'étape suivante d'une assertion :

1. Pour ouvrir le menu, cliquez avec le bouton droit de la souris sur l'assertion dans l'éditeur de modèles.
2. Sélectionnez If triggered, then (Opération à effectuer en cas de déclenchement) et effectuez l'une des opérations suivantes :
   - Indiquez si générer un avertissement ou une erreur.
   - Indiquez si terminer, mettre en échec ou interrompre l'étape.
   - Sélectionnez l'étape suivante à exécuter.

Ensembles de données

Un ensemble de données (page 568) est une collection de valeurs que vous pouvez utiliser pour définir des propriétés dans un scénario de test pendant l'exécution d'un test. Cette capacité permet d'introduire des données de test externes à un scénario de test.

Les ensembles de données sont souvent des lignes de données que vous pouvez insérer dans des propriétés en tant que paires nom-valeur. Toutefois, il arrive qu'un ensemble de données renvoie une valeur de propriété unique.

Vous pouvez créer des ensembles de données internes à DevTest, ou de manière externe ; par exemple, dans un fichier ou une table de base de données.

Lors de l'exécution d'un test, DevTest affecte des propriétés aux étapes spécifiées dans l'éditeur d'ensembles de données. Lorsque la ou les dernières valeurs de données sont lues à partir de l'ensemble de données, l'une des actions suivantes peut se produire :

- Vous pouvez réutiliser les données en commençant par la partie supérieure de l'ensemble de données.
- Vous pouvez rediriger le test vers une étape du scénario de test.

Les ensembles de données peuvent être globaux ou locaux.

Les rubriques suivantes sont incluses.

Ensembles de données globaux et locaux

Les ensembles de données peuvent être globaux ou locaux.

Ensembles de données globaux

Par défaut, un ensemble de données est global.

Le serveur de coordination est chargé de fournir des données à toutes les étapes d'un test.

Dans ce cas, toutes les instances de modèle partagent une instance unique de l'ensemble de données.

L'ensemble de données global est partagé et appliqué à toutes les instances du modèle, même si elles sont exécutées dans des simulateurs différents.

Ensembles de données locaux

Vous pouvez rendre un ensemble de données local en sélectionnant la case à cocher Local lors de sa création.

Chaque instance obtient essentiellement sa propre copie de l'ensemble de données.

Un ensemble de données local fournit une copie de l'ensemble de données à chaque instance exécutée.

Exemple

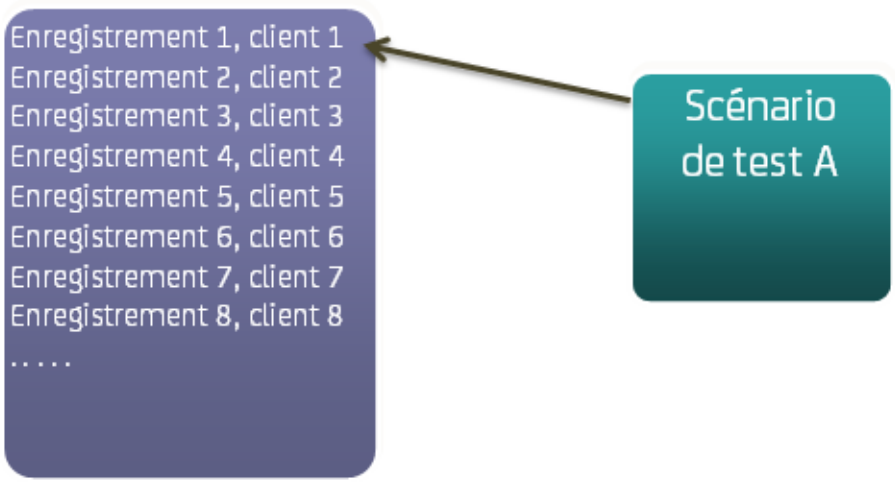
Cet exemple contient les composants suivants :

- Trois utilisateurs virtuels simultanés
- Un ensemble de données local contenant 100 lignes de données
- Un scénario de test exécutant plus de 100 lignes de données en boucle et s'arrête

Chaque utilisateur virtuel voit les 100 lignes de données dans l'ensemble de données.

Pour un ensemble de données local :

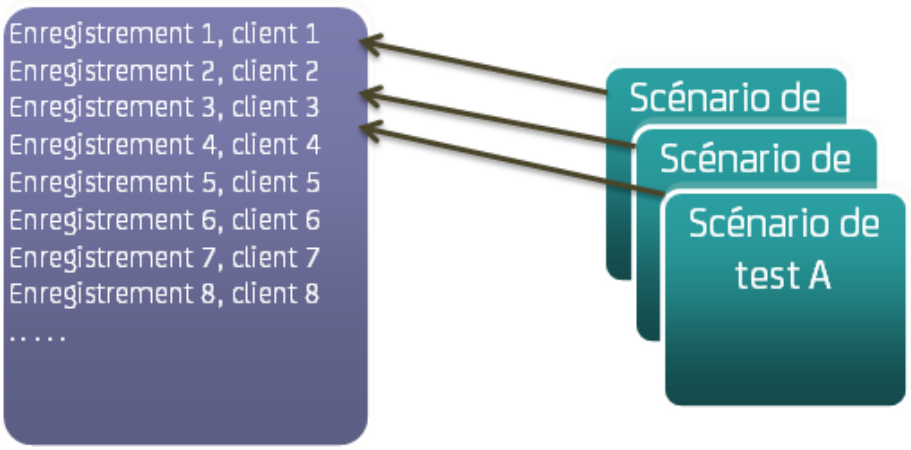
Exécution unique (1 utilisateur virtuel) : scénario de test A, récupère la première ligne de données, enregistrement 1, client 1.



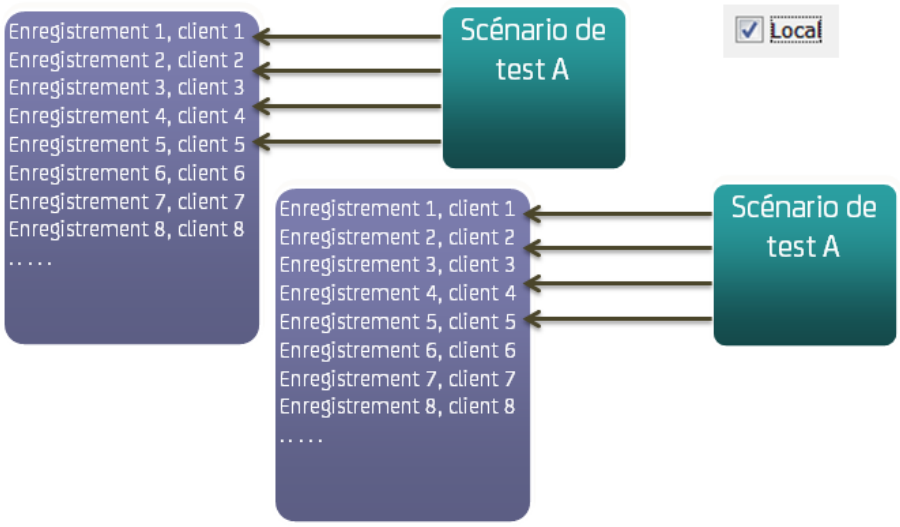
Un ensemble de données partagé par les utilisateurs virtuels est global.

Pour un ensemble de données global :

Lorsque le scénario de test A est simulé avec trois utilisateurs virtuels ou pour une exécution continue, chaque scénario de test contiendra la ligne de données suivante. DevTest partage un ensemble de données sur plusieurs exécutions selon les instructions spécifiées dans le document de simulation.



Chaque utilisateur virtuel (instance) obtient sa propre copie de l'ensemble de données local.



Lorsque l'option Local est sélectionnée et le scénario de test A contient une boucle, le test lit toutes les lignes de données de l'ensemble de données. Chaque exécution obtient sa propre copie de l'ensemble de données.

Exécution en boucle de scénario de test

En général, les données d'un ensemble de données déterminent le nombre de fois qu'un scénario de test est exécuté.

Vous pouvez implémenter l'exécution en boucle de plusieurs façons :

- Un test est défini pour se terminer lorsque toutes les données de l'ensemble de données sont épuisées.
- Un test est défini pour réutiliser l'ensemble de données lorsque les données sont épuisées.
- Une étape de test peut s'appeler ou vous pouvez configurer une série d'étapes dans une boucle qui s'exécute jusqu'à ce que les données de l'ensemble de données soient épuisées.

Utilisez un ensemble de données de compteur numérique pour exécuter une étape spécifique un certain nombre de fois. Vous pouvez définir la fréquence au niveau de l'étape ou au niveau du scénario de test.

Par exemple, vous voulez tester la fonctionnalité de connexion de votre application à l'aide de 100 paires d'ID d'utilisateur et de mots de passe. Pour réaliser ce test, définissez une seule étape qui s'appellera jusqu'à épuiser l'ensemble de données (contenant 100 lignes de données). Lorsque l'ensemble de données est épuisé, le test peut être redirigé vers l'étape ordinaire suivante dans le scénario, ou vers l'étape de fin. De même, vous pouvez utiliser un ensemble de données de compteur avec l'ensemble de données d'ID d'utilisateur et de mots de passe.

Si vous configurez un ensemble de données global dans la première étape d'un scénario de test de sorte à terminer le test lorsque l'ensemble de données est épuisé, toutes les instances de test d'une exécution simulée se terminent.

D'autres paramètres de simulation, tels que la durée de l'état stable, seront remplacés. Les ensembles de données locaux ne mettent pas fin à l'exécution simulée de cette façon, ni les ensembles de données dans les étapes autres que la première étape.

Ensembles de données aléatoires

Un ensemble de données aléatoire est un type spécial d'ensemble de données, qui peut être considéré comme un encapsulateur d'un autre ensemble de données.

Vous pouvez sélectionner un ensemble de données à rendre aléatoire pour des étapes spécifiques et le nombre maximum d'enregistrements pour la randomisation.

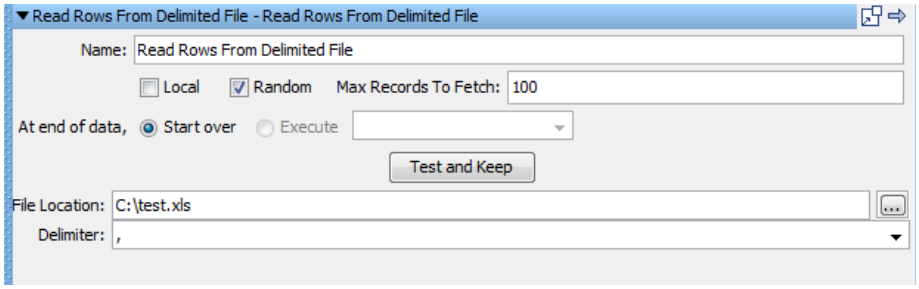
Lorsqu'un ensemble de données aléatoire encapsule un ensemble de données, un nombre *n* de copies de l'ensemble de données est ajouté à la liste de l'ensemble de données aléatoire, où *n* = nombre maximum de lignes. Ainsi, lorsque *n* = 10, DevTest effectue 10 copies des lignes de l'ensemble de données.

Lorsqu'une étape référence l'ensemble de données aléatoire, une ligne aléatoire est sélectionnée à partir de l'ensemble de données.

Si un ensemble de données aléatoire nommé **RAND1** lit une colonne nommée **Fruit** et que la valeur de Max Records to Fetch (Nombre maximum d'enregistrements à extraire) est définie sur 10, le nombre aléatoire **RAND1** sera généré pour sélectionner une ligne. **RAND1** est compris dans la plage 0 à *n*-1. Fruit est la valeur de la colonne Fruit détectée dans cette ligne.

Pour définir un ensemble de données comme aléatoire :

1. Sélectionnez l'ensemble de données que vous souhaitez définir comme aléatoire. Cet exemple utilise l'ensemble de données Read Rows from Delimited File (Lire les lignes à partir d'un fichier délimité).



2. Cochez la case Random (Aléatoire).
3. Entrez une valeur dans le champ Max Records To Fetch (Nombre maximum d'enregistrements à extraire).

   Cette valeur est le nombre maximum d'enregistrements que vous voulez récupérer à partir de l'ensemble de données. Si le nombre est supérieur aux nombre d'enregistrements figurant dans l'ensemble de données, un nombre plus petit est utilisé pour créer l'ensemble de données aléatoire.

Exemples de scénarios

Pour afficher le comportement des tests à l'aide d'ensembles de données, considérez les scénarios suivants :

Un test a 15 utilisateurs virtuels et un ensemble de données contenant deux lignes de données.

Scénario 1 : recommencer à la fin des données

Le premier utilisateur lit la première ligne de données et le deuxième utilisateur lit la deuxième ligne. Le troisième utilisateur recommence du début et lit la première ligne, etc. Ce processus se poursuit jusqu'à ce que les 15 utilisateurs aient exécuté le test. La première ligne a été lue huit fois et la deuxième ligne sept fois.

Scénario 2 : à la fin des données, exécuter l'étape End (Fin)

Le premier utilisateur lit la première ligne de données et le deuxième utilisateur lit la deuxième ligne. A partir du troisième utilisateur, tous les utilisateurs commence le test et passe immédiatement à l'étape de fin.

Un test a 100 utilisateurs virtuels et un ensemble de données contenant 1 500 lignes de données. Les tests s'exécutent simultanément.

Scénario 1 : recommencer à la fin des données

Lorsque les 100 utilisateurs démarrent, ils lisent les 100 premières lignes de données. Selon le document de simulation, à la fin des cycles les utilisateurs lancent de nouvelles exécutions du scénario de test et consomment plus de lignes de l'ensemble de données. Si toutes les lignes sont consommées et l'exécution du test n'est pas terminée, il recommence à partir de la première ligne, jusqu'à ce que l'exécution se termine.

Scénario 2 : à la fin des données, exécuter l'étape End (Fin)

L'exécution du test se termine après 1 500 cycles.

Un test a 10 utilisateurs virtuels et un ensemble de données contenant 10 000 lignes de données. Le document de simulation spécifie que le test doit s'exécuter pendant 2 minutes.

Les dix utilisateurs démarrent et lisent les dix premières lignes de données. Ils continuent de consommer les 10 000 lignes de données et la ligne qu'ils atteignent dépend de leur vitesse d'exécution. Après deux minutes, le test se termine.

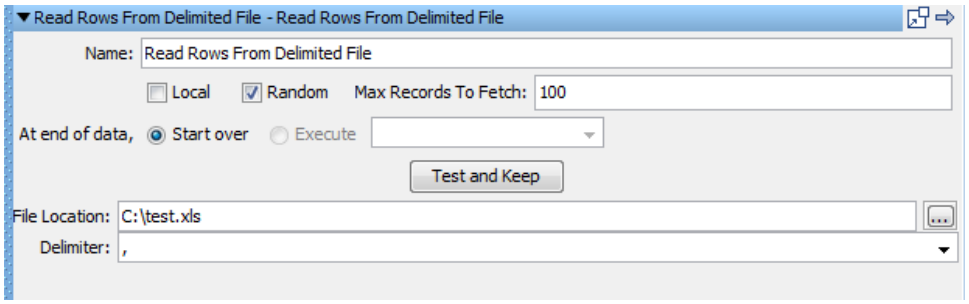
Ajout d'un ensemble de données

Procédez comme suit:

1. Dans l'éditeur de modèles, sélectionnez un scénario de test.
2. Développez l'onglet Data Sets (Ensembles de données) dans le panneau droit.
3. Cliquez sur Ajouter pour ouvrir le panneau d'ensemble de données répertoriant les ensembles de données communs.
4. Cliquez sur l'ensemble de données requis pour ouvrir l'éditeur Data Set Editor (Editeur d'ensembles de données) approprié. L'éditeur est spécifique à chaque ensemble de données.

Exemple d'éditeur d'ensembles de données

L'image suivante affiche l'éditeur pour l'ensemble de données Read Rows From a Delimited Data File (Lire les lignes à partir d'un fichier de données délimité).



Le panneau supérieur de l'éditeur d'ensembles de données est commun à tous les types d'ensemble de données. Les options disponibles sont les suivantes :

Name (Nom)

Nom de l'ensemble de données.

Local

Cochez cette case si vous voulez définir un ensemble de données local. La valeur par défaut est globale (option désélectionnée).

Random (Aléatoire)

Cochez la case Random si vous voulez définir l'ensemble de données en tant qu'ensemble de données aléatoire et entrez le nombre d'enregistrements maximum à extraire.

At End Of Data (A la fin des données)

Instructions indiquant la marche à suivre une fois que toutes les données ont été lues. Vous pouvez utiliser les options suivantes :

Start over (Recommencer)

Poursuivre la lecture des données à partir du début de l'ensemble de données.

Execute (Exécuter)

Sélectionnez l'étape à exécuter une fois que toutes les données ont été lues. Le menu déroulant a été prérempli avec toutes les étapes disponibles dans le scénario de test.

Test and Keep (Tester et conserver)

Une fois que tous les paramètres sont entrés, cliquez sur ce bouton pour tester l'ensemble de données et le charger dans les étapes du scénario de test.

Le panneau inférieur de l'éditeur d'ensembles de données est spécifique à l'ensemble de données créé. Pour un ensemble de données Read Rows from Delimited File (Lire les lignes à partir d'un fichier délimité), entrez :

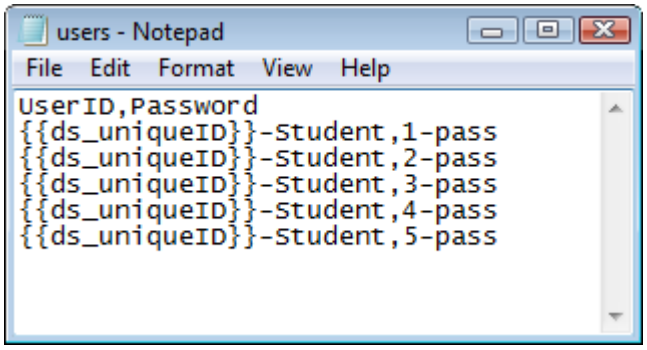
File Location (Emplacement du fichier)

Entrez le nom de chemin complet du fichier texte ou utilisez le bouton Browse (Parcourir). Vous pouvez utiliser une propriété dans le chemin d'accès (par exemple, LISA_HOME).

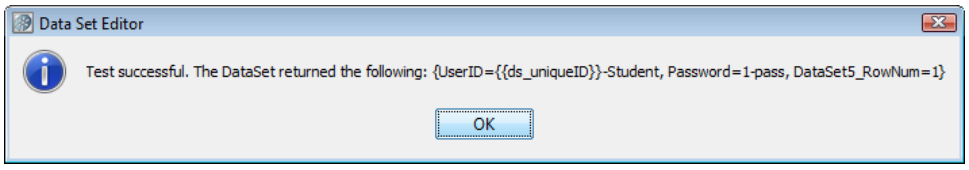
Delimiter (Délimiteur)

Entrez le délimiteur utilisé. Toutes les valeurs sont autorisées comme délimiteur. Certains délimiteurs communs sont inclus dans la liste déroulante.

Cet ensemble de données requiert un fichier texte délimité. Dans l'exemple suivant, la première ligne spécifie les noms de propriété userid et password. Les lignes ultérieures répertorient les valeurs de données à utiliser pour ces propriétés.



Lorsque vous cliquez sur Test and Keep (Tester et conserver), la première ligne de données est chargée. Un message confirme que l'ensemble de données peut être lu et indique la première ligne de données.



Terminer un scénario de test par un ensemble de données

Les conditions suivantes doivent être remplies pour qu'un ensemble de données termine une exécution de test :

- L'ensemble de données doit être global.
- L'ensemble de données doit être défini sur At End of Data: Execute end (A la fin des données : Exécuter la fin).
- L'ensemble de données doit être augmenté dans la première étape du scénario de test.

Soyez prudent lorsque vous appliquez un ensemble de données global à la première étape d'un scénario de test, car il réduit la totalité de l'exécution simulée avant de terminer les cycles.

Changement de nom d'un ensemble de données

Pour renommer un ensemble de données :

- Sélectionnez l'ensemble de données dans le panneau Elements (Eléments) et cliquez sur l'icône Rename (Renommer) dans la barre d'outils.
- Sélectionnez l'ensemble de données dans l'éditeur de modèles et cliquez avec le bouton droit de la souris pour ouvrir un menu contextuel et sélectionner l'option Rename.
- Sélectionnez l'ensemble de données dans le panneau Elements et cliquez sur  Revert to Default Name (Rétablir le nom par défaut) pour rétablir le nom de l'ensemble de données par défaut.

Déplacement d'un ensemble de données

Vous pouvez déplacer des ensembles de données dans l'éditeur de modèles d'une étape de test vers une autre à l'aide de l'option de glisser-déposer.

Procédez comme suit:

1. Cliquez sur l'ensemble de données associé à une étape de test, par exemple, l'étape 1.
2. Sélectionnez l'ensemble de données et faites-le glisser sur l'étape de test cible, par exemple, l'étape 2 dans l'éditeur de modèles.
3. L'ensemble de données déplacé est appliqué à l'étape 2.

Sélection de l'étape suivante de l'ensemble de données

Vous pouvez sélectionner l'étape suivante ou l'étape de fin à exécuter après le déclenchement de l'ensemble de données.

Procédez comme suit:

1. Pour ouvrir le menu, cliquez avec le bouton droit de la souris sur l'ensemble de données dans l'éditeur de modèles.
2. Cliquez sur le menu At end (A la fin) et sélectionnez l'étape suivante à exécuter.

Ensembles de données et propriétés

Les ensembles de données peuvent utiliser des propriétés.

Exemples d'utilisations indispensables des propriétés :

- Lorsque vous spécifiez l'emplacement d'un ensemble de données externe, vous pouvez utiliser une propriété, LISA_HOME par exemple, plutôt qu'une valeur codée de manière irréversible pour le chemin d'accès. Par exemple : **LISA_HOME\myTests\myDataset.csv**

  L'utilisation d'une propriété au lieu d'une valeur codée de manière irréversible améliore la portabilité d'un ensemble de données. En règle générale, ce type d'utilisation des propriétés implique qu'elles doivent être disponibles à un stade précoce de l'exécution du test, c'est pourquoi vous devez les définir dans l'une des manières suivantes :

  - Comme propriété système
  - Dans une configuration

  Vous pouvez définir une valeur factice dans votre configuration et la modifier ultérieurement. Cette valeur permet au fichier d'être trouvé lors de la conception. Cette utilisation des propriétés est importante lorsque vous exécutez des tests sur DevTest Server.
- Les valeurs d'ensemble de données peuvent contenir des propriétés. Ces valeurs sont évaluées lorsque la valeur est lue. Par exemple, vous pouvez configurer une valeur de connexion étudiant_1 dans l'ensemble de données. Si la valeur actuelle de la propriété étudiant est **Bart**, la valeur de données résultante devient **Bart_1**.
- Les ensembles de données locaux peuvent utiliser des propriétés à partir du scénario de test. Ce n'est cependant pas le cas des ensembles de données globaux, car tous les utilisateurs virtuels les partagent.

Compagnons

Un compagnon est un élément de LISA qui s'exécute avant et/ou après toutes les exécutions de scénario de test. Les compagnons sont utilisés pour configurer le comportement global dans le scénario de test. Ce comportement peut inclure la simulation de la bande passante de navigateur et du type de navigateur, la définition des points de synchronisation dans les tests de charge et la création d'un chargeur de classes de bac à sable. Dans un sens, les compagnons définissent un certain contexte pour l'exécution de scénario de test.

Ces compagnons vous fournissent une aide précieuse pour le scénario de test, avant même l'exécution des étapes.

Les rubriques suivantes sont incluses.

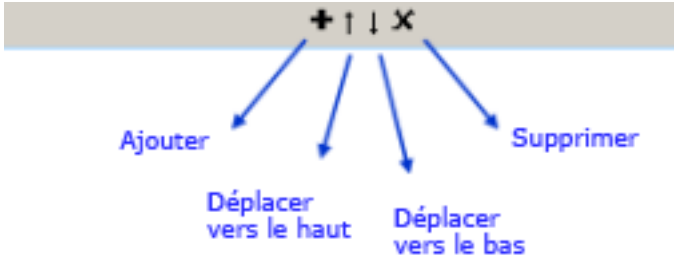
Ajout d'un compagnon

Procédez comme suit:

1. Ouvrez un scénario de test et cliquez sur l'élément Companion (Compagnon) dans le panneau droit.
2. Cliquez sur + Add (Ajouter). Le menu Companion s'ouvre.
3. Chaque compagnon a un éditeur différent. Pour ouvrir l'éditeur approprié, sélectionnez le compagnon requis. Chaque type de compagnon ainsi que l'ensemble de leurs paramètres sont décrits en détail dans la section Descriptions des compagnons de la rubrique *Référence.*

Barre d'outils de compagnon

Au bas de chaque élément se trouve une barre d'outils qui comprend des icônes vous permettant d'ajouter, de supprimer ou de réorganiser les éléments.



Hooks de DevTest

Les hooks de DevTest constituent un moyen automatique global d'exécuter la logique au début ou à la fin d'un scénario de test. Les hooks appliquent cette logique à tous les scénarios de test dans l'environnement sur lequel le hook est déployé.

Les hooks fonctionnent de la même façon que les compagnons. Ils s'exécutent avant le début d'un test et après la fin d'un test.

La seule différence réside dans le fait que les hooks sont définis au niveau de l'application et que tous les scénarios de test de DevTest les appliquent. Si un hook existe, il est utilisé pour tous les scénarios de test.

Un *hook* est un mécanisme qui permet d'inclure automatiquement la logique de configuration et/ou la logique de fin de test, pour tous les tests exécutés dans DevTest. Pour clarifier, un hook est un compagnon qui s'applique à l'échelle du système. Toutes les opérations qu'un hook peut effectuer peuvent être modélisées sous la forme de compagnon.

Les hooks sont utilisés pour configurer des environnements de test, eviter l'exécution de tests configurés de manière incorrecte ou non conformes aux recommandations définies ; ils permettent également d'effectuer des opérations communes.

Les hooks sont des classes Java qui se trouvent dans le classpath de DevTest. Les hooks ne sont pas définis dans le fichier .lisaextensions, mais dans les fichiers local.properties ou lisa.properties :

```
lisa.hooks=com.mypackage1.MyHook1, com.mypackage2.MyHook2
```

Les hooks sont exécutés ou appelés au démarrage et à la fin de tous les sous-processus dans un scénario de test, en plus du scénario de test lui-même. Si un scénario de test comprend trois sous-processus, la logique de hook est exécutée quatre fois. La logique est exécutée une fois pour le test principal et une fois pour chacun des trois sous-processus.

Pour empêcher un sous-processus d'exécuter **startHook et/ou endHook**, incluez l'action suivante :

```
String marker =
(String)testExec.getStateValue(TestExec.FROM_PARENT_TEST_MARKER
_KEY)

marker est défini sur true uniquement lorsqu'il s'agit d'un
sous-processus.

if (!"true".equals(marker))

Définit toute les logiques de hook de début et de fin qui ne doivent
pas être exécutées dans un sous-processus.
```

Différences entre hooks et compagnons

- La portée des hooks est globale. Les testeurs n'incluent généralement pas de hook dans leur scénario de test de la même manière qu'ils incluent des compagnons. Les hooks sont enregistrés au niveau du système. Si les tests doivent tous inclure une logique et vous ne voulez pas que les utilisateurs l'ignorent par mégarde, définir un hook est une meilleure solution.
- Les hooks sont déployés au niveau de l'installation, non au niveau du scénario de test. Si un test est exécuté sur deux ordinateurs et qu'un hook est enregistré sur l'un d'eux, le hook sera exécuté uniquement lors de la simuation du test sur l'ordinateur sur lequel le hook est déployé. Les compagnons qui sont définis dans le scénario de test s'exécutent indépendamment de la configuration de niveau installation.
- Les hooks sont pratiquement transparents pour les utilisateurs. Ils ne peuvent pas requérir par conséquent de paramètres personnalisés auprès des utilisateurs. Leurs paramètres proviennent des propriétés de la configuration ou du système. Les compagnons peuvent avoir des paramètres personnalisés, car ils sont rendus dans l'éditeur de modèles.

Editeur Complex Object Editor (COE) (Editeur d'objets complexes)

L'éditeur Complex Object Editor vous permet d'interagir avec des objets Java sans devoir écrire de code Java.

Vous pouvez modifier la valeur actuelle d'un paramètre d'entrée, effectuer des appels de méthode et examiner des valeurs renvoyées. Vous pouvez également appliquer un simple filtre intégré et ajouter des assertions simples à la valeur renvoyée.

Plusieurs étapes de test impliquent la manipulation d'objets Java. Si vous travaillez directement avec des objets Java, comme dans une étape Dynamic Java Execution (Exécution Java dynamique) ou une étape Enterprise JavaBean (EJB), ou indirectement, comme avec des paramètres d'entrée pour un service Web ou des messages de retour à partir d'un bus ESB (Enterprise Service Bus), vous utilisez l'éditeur Complex Object Editor (Editeur d'objets complexes).

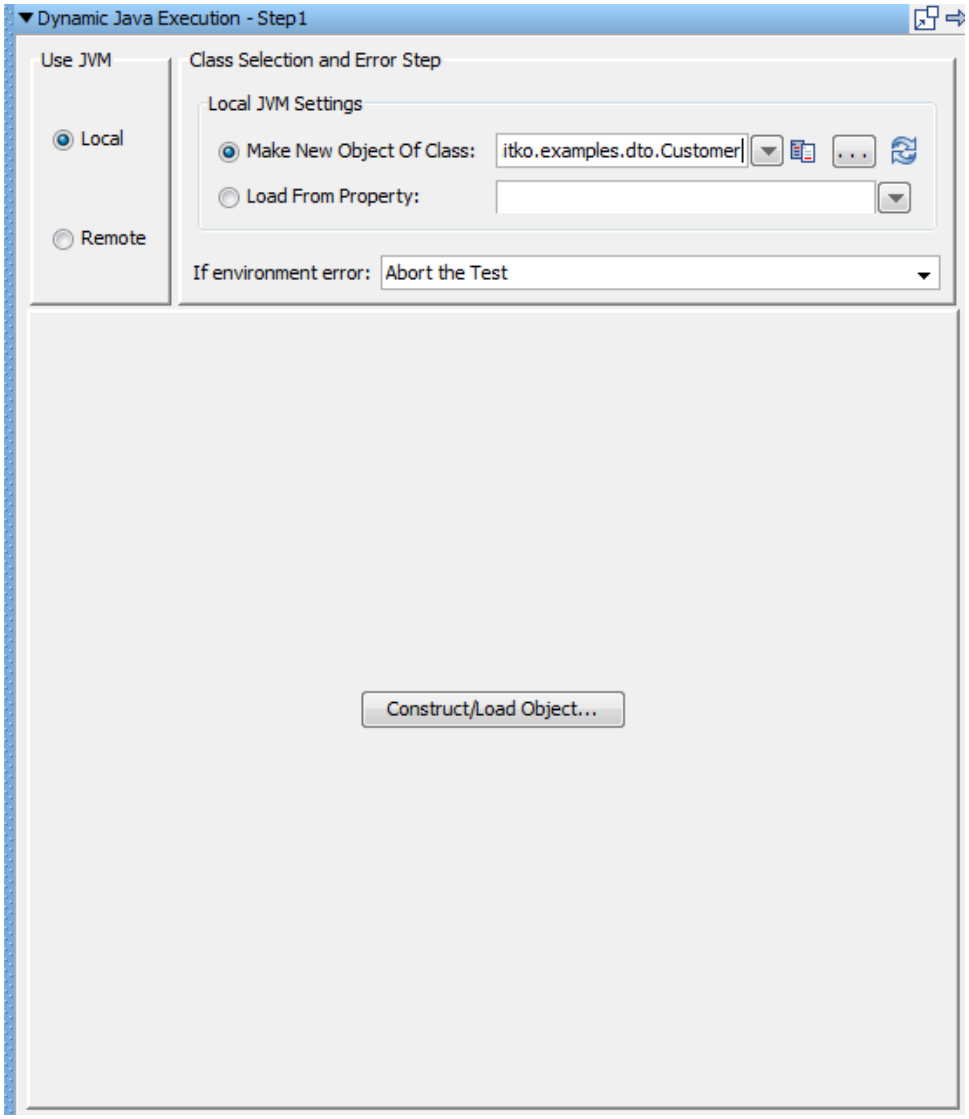
Ce chapitre vous permet d'abord de vous familiariser avec l'interface utilisateur, puis de consulter plusieurs scénarios d'utilisation de plus en plus complexes.

Les rubriques suivantes sont incluses.

Appel de l'éditeur Complex Object Editor (Editeur d'objets complexes)

L'apparence de l'éditeur Complex Object Editor (Editeur d'objets complexes) est identique, où qu'il soit.

Par exemple, lorsque vous appelez l'étape Dynamic Java Execution (Exécution Java dynamique) à l'aide de la classe Customer, la fenêtre suivante s'affiche :

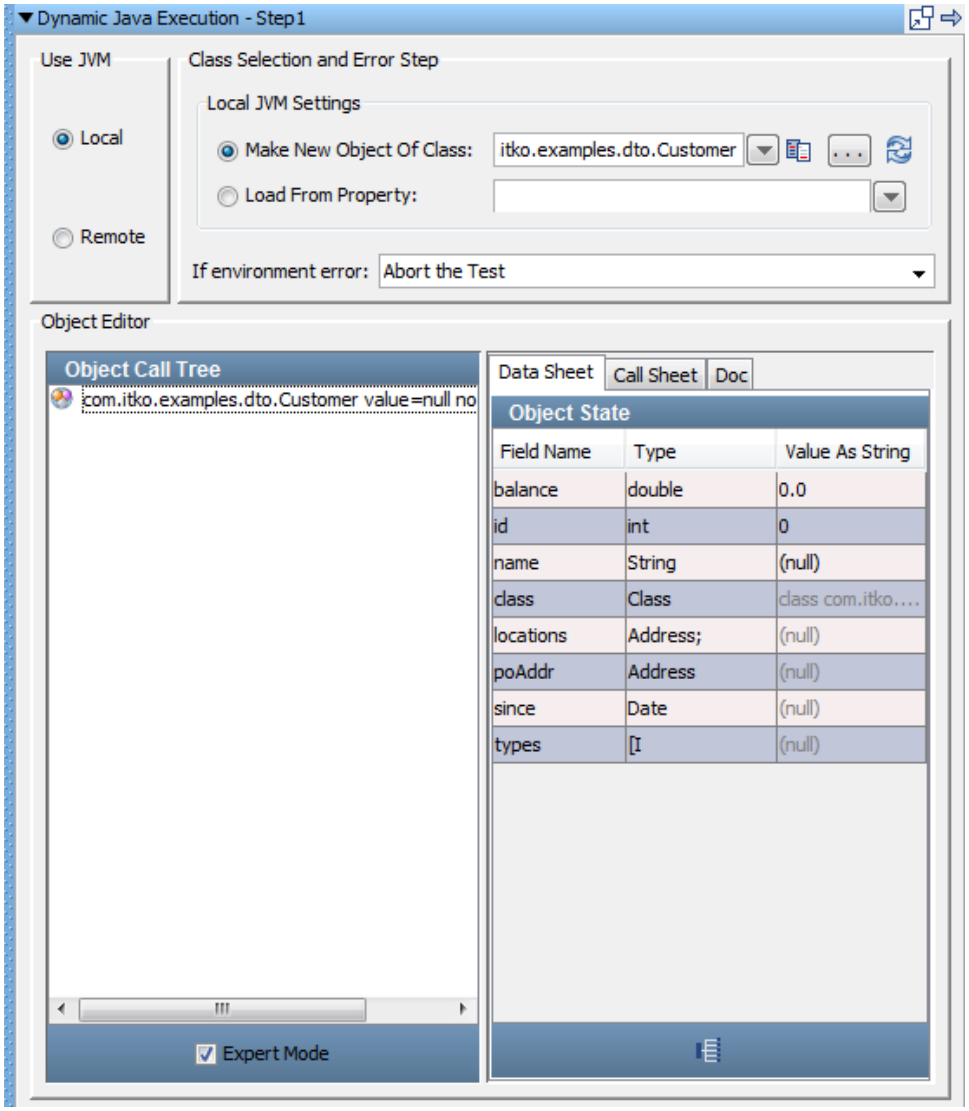


Sélectionnez la machine virtuelle Java locale et entrez *com.itko.examples.dto.Customer* dans le champ Make New Object Of Class (Créer un objet de classe).

Pour charger l'objet dans l'éditeur d'objets, cliquez sur Construct/Load Object (Générer/charger un objet).

Editeur Complex Object Editor (Editeur d'objets complexes)

Une fois que l'objet est chargé, l'éditeur Complex Object Editor est appelé.



L'éditeur d'objets est divisé en deux panneaux. Le panneau de gauche, l'arborescence Object Call Tree (Arborescence des appels d'objet), suit les appels de méthode, ainsi que leurs paramètres d'entrée et les valeurs renvoyées. Le panneau Object Call Tree (page 179) est décrit en détail dans la rubrique suivante.

Le panneau droit, nommé Object State (Etat de l'objet), comporte un ensemble d'onglets dynamiques (panneau Data Sheet (Feuille de données), panneau Call Sheet (Feuille d'appel), panneau Doc (Document) (page 181)) qui vous indiquent les options disponibles.

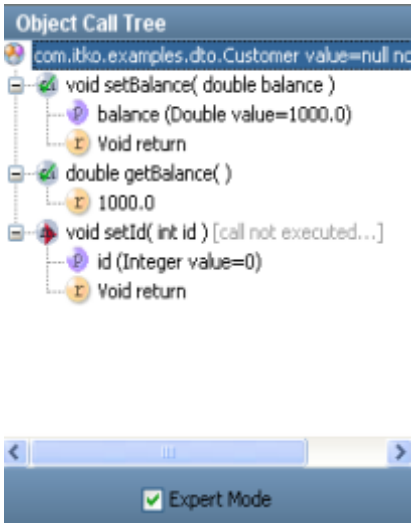
Remarque : Si la réponse est un fichier XML supérieur à 5 Mo, il est affiché en texte brut sans vue DOM. Vous pouvez ajuster cette limite de 5 Mo avec la propriété **gui.viewxml.maxResponseSize.**

La fenêtre précédente indique un objet Java de type **Customer loaded** dans l'éditeur. Cet objet a été chargé à l'aide de l'étape de test Dynamic Java Execution (Exécution Java dynamique), mais d'autres opérations auraient pu également le charger. Aucun appel n'a été effectué sur l'objet.

Panneau Object Call Tree (Arborescence des appels d'objet)

Lorsque vous manipulez un objet Java (Customer), l'arborescence Object Call Tree se développe pour vous permettre d'effectuer un suivi des appels effectués et des valeurs de paramètre associées.

Exemple d'arborescence Object Call Tree après plusieurs appels de méthodes :



Icônes du panneau Object Call Tree (Arborescence des appels d'objet)

Les icônes suivantes sont utilisées pour identifier les branches dans l'arborescence Object Call Tree :

Icône	Description
	Type (classe) de l'objet actuellement chargé, suivi par la réponse à l'appel de la méthode toString de l'objet.
	Constructeur appelé. Cette icône est affichée si plusieurs constructeurs existent.
	Appel de méthode qui n'a pas été exécuté.

Appel de méthode qui a été exécuté.

Paramètres d'entrée (type et valeur actuelle) pour la méthode englobante.

Valeur renvoyée (valeur actuelle si l'appel a été exécuté) pour la méthode englobante.

Cliquer sur un élément dans l'arborescence Object Call Tree affiche l'ensemble d'onglets approprié de la feuille Data Sheet (Feuille de données) et Call Sheet (Feuille d'appel) dans le panneau droit.

Cliquer avec le bouton droit de la souris sur un élément dans l'arborescence Object Call Tree permet d'afficher un menu.

Vous pouvez exécuter tous les appels ou vous pouvez marquer tous les appels comme non exécutés dans l'arborescence Object Call Tree.

Panneaux Data Sheet (Feuille de données) et Call Sheet (Feuille d'appel)

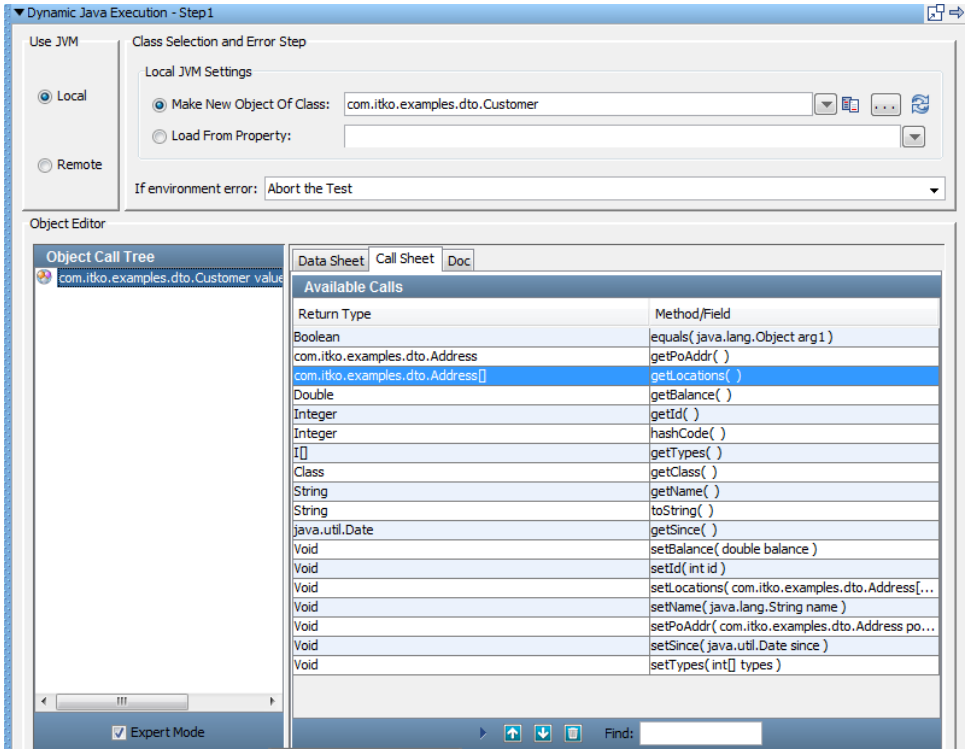
Panneau Data Sheet (Feuille de données)

Le panneau droit comprend trois onglets. L'onglet Data Sheet (Feuille de données) est actif par défaut.

Vous pouvez modifier les données affichées en noir dans cet onglet. Les valeurs de données sont modifiées dans la colonne Value as String (Valeur en tant que chaîne). Ces valeurs sont toujours des primitives ou des chaînes. Vous ne pouvez pas modifier les valeurs grisées dans le panneau Data Sheet (Feuille de données), mais vous le pourrez dans d'autres fenêtres. Le champ Address (Adresse), par exemple, est un objet de type Address que vous ne pouvez pas modifier dans cet onglet.

Panneau Call Sheet (Feuille d'appel)

Dans l'onglet Call Sheet, l'éditeur Complex Object Editor (Editeur d'objets complexes) affiche les appels de méthodes ou de champs qui sont disponibles, et leurs types de retour.

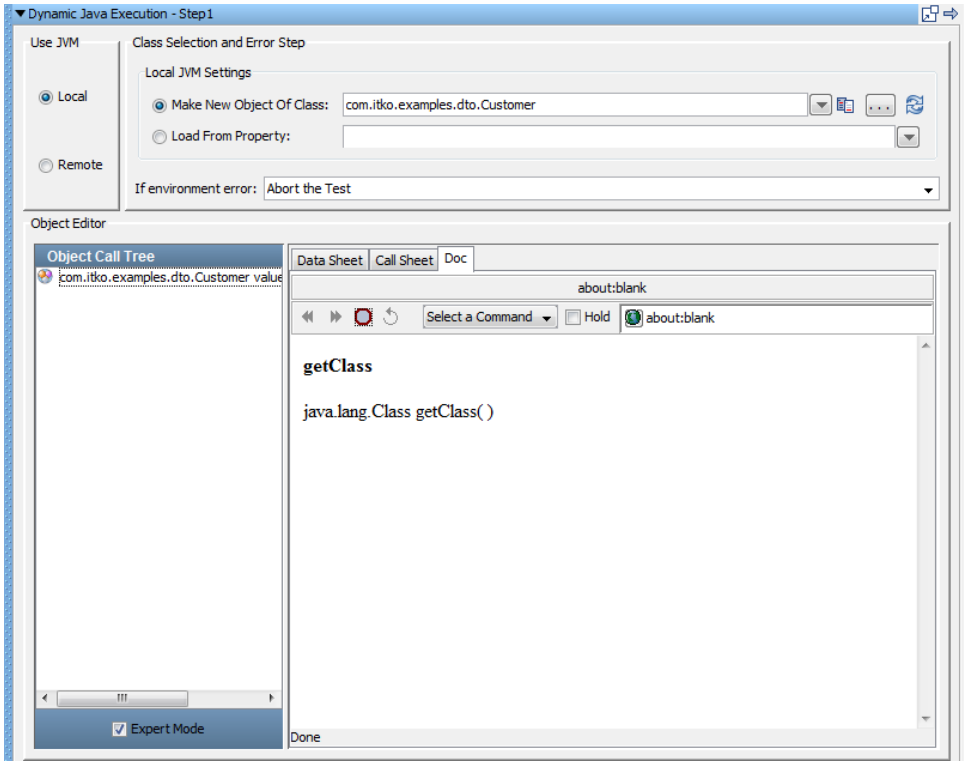


Pour ajouter une méthode, sélectionnez-la dans la liste Methods/Fields (Méthodes/Champs) et cliquez sur  Add Method (Ajouter une méthode) en bas de l'onglet. La méthode sélectionnée s'affiche dans l'arborescence Object Call Tree (dans le panneau gauche).

Lorsque vous êtes dans l'arborescence des appels d'objet, vous pouvez fournir des paramètres d'entrée et appeler la méthode.

Panneau Doc (Document)

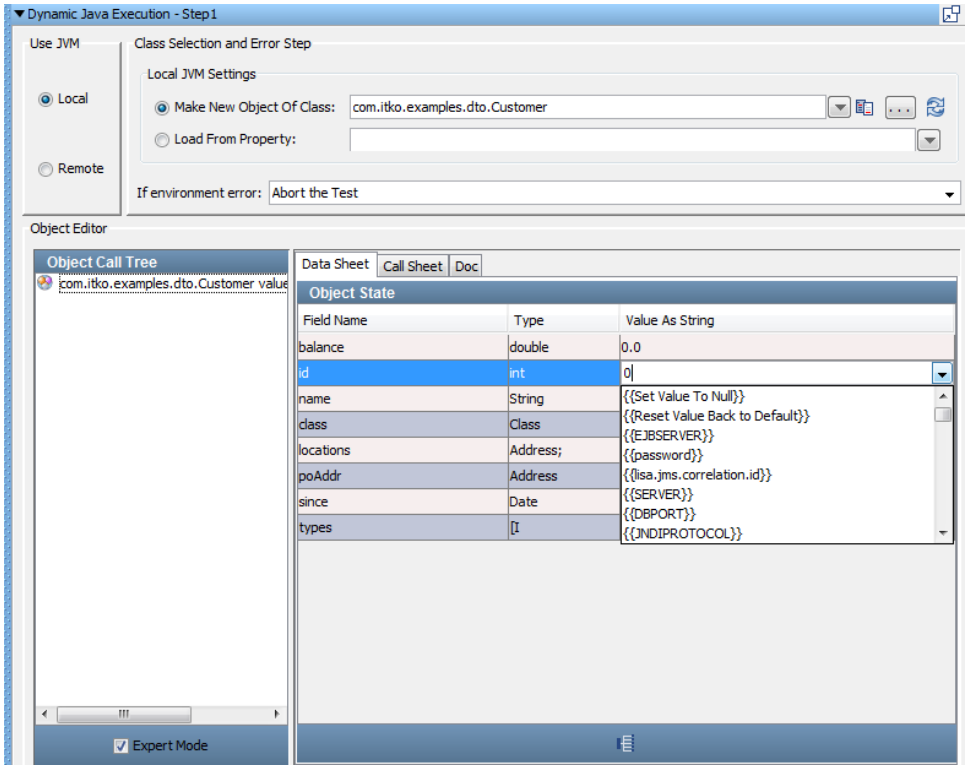
L'onglet Doc affiche toute la documentation d'API Java disponible pour cette classe.



Panneaux d'interaction d'objet

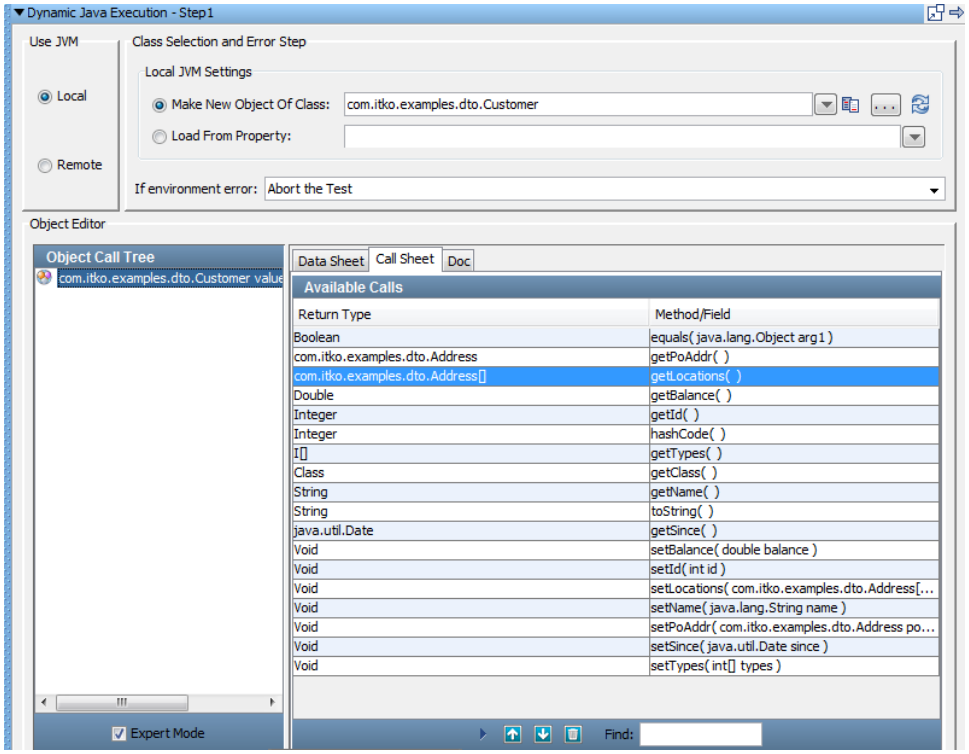
Chaque action affichée dans les feuilles Data Sheet (Feuille de données) et Call Sheet (Feuille d'appel) a une interface différente.

L'onglet et son interface changent en fonction de la sélection effectuée dans l'arborescence Object Call Tree, dans le panneau gauche.



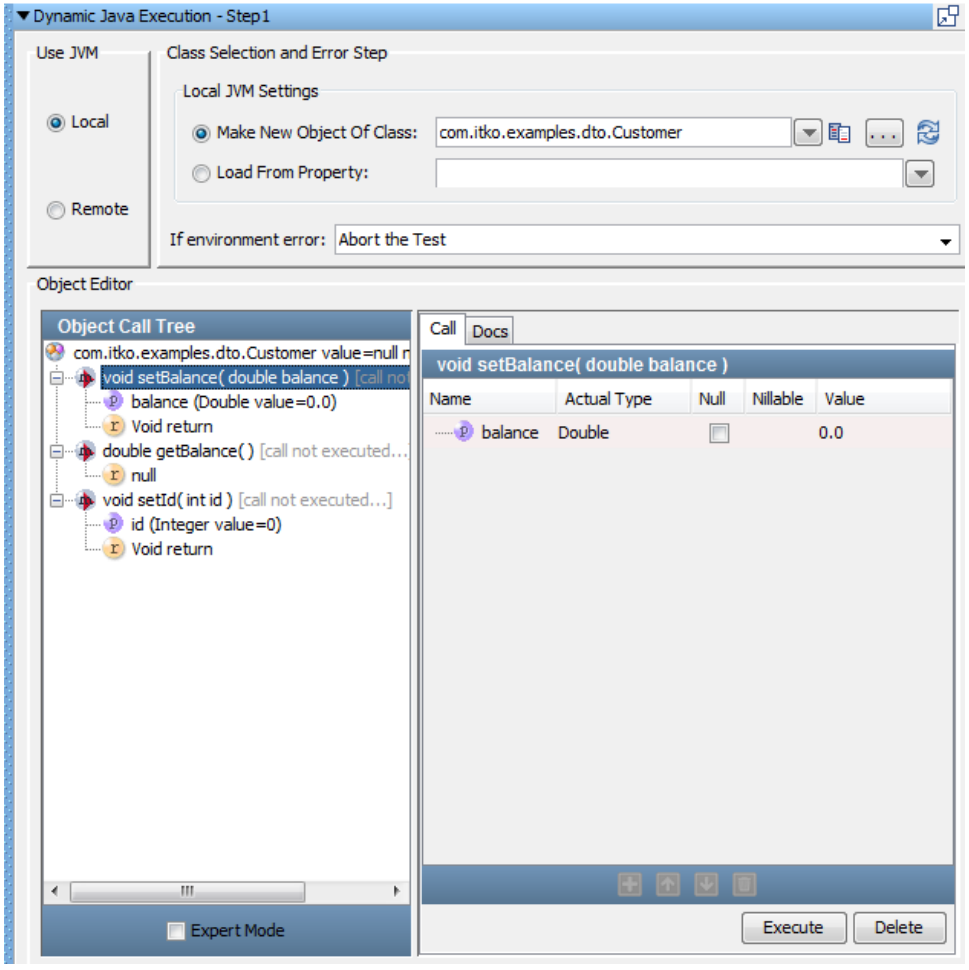
Panneaux d'objet

Les onglets Data Sheet, Call Sheet et Doc sont disponibles lorsqu'un objet est sélectionné dans l'arborescence Object Call Tree.



Panneaux d'appel de méthode

Si vous sélectionnez un appel de méthode dans l'arborescence Object Call Tree, les onglets Call Sheet et Doc sont disponibles dans le panneau droit.



Fournissez des valeurs pour les paramètres d'entrée dans ce panneau.

Les informations sur chaque paramètre sont fournies, vous n'avez donc qu'à fournir la valeur. L'exemple suivant est simple. Le paramètre unique est de type double, vous pouvez donc entrer une valeur ou un nom de propriété dans la colonne Value.

Le menu déroulant maintient une liste des propriétés actuelles. Si vous devez entrer un objet en tant que paramètre d'entrée, des opérations supplémentaires sont nécessaires. Plusieurs approches permettant de fournir des objets sont décrites dans les sections suivantes.

Remarquez que le mode Expert Mode (Mode Expert) au bas du panneau gauche n'est pas sélectionné. Cela indique que le mode Simple est sélectionné.

Mode simple et mode expert

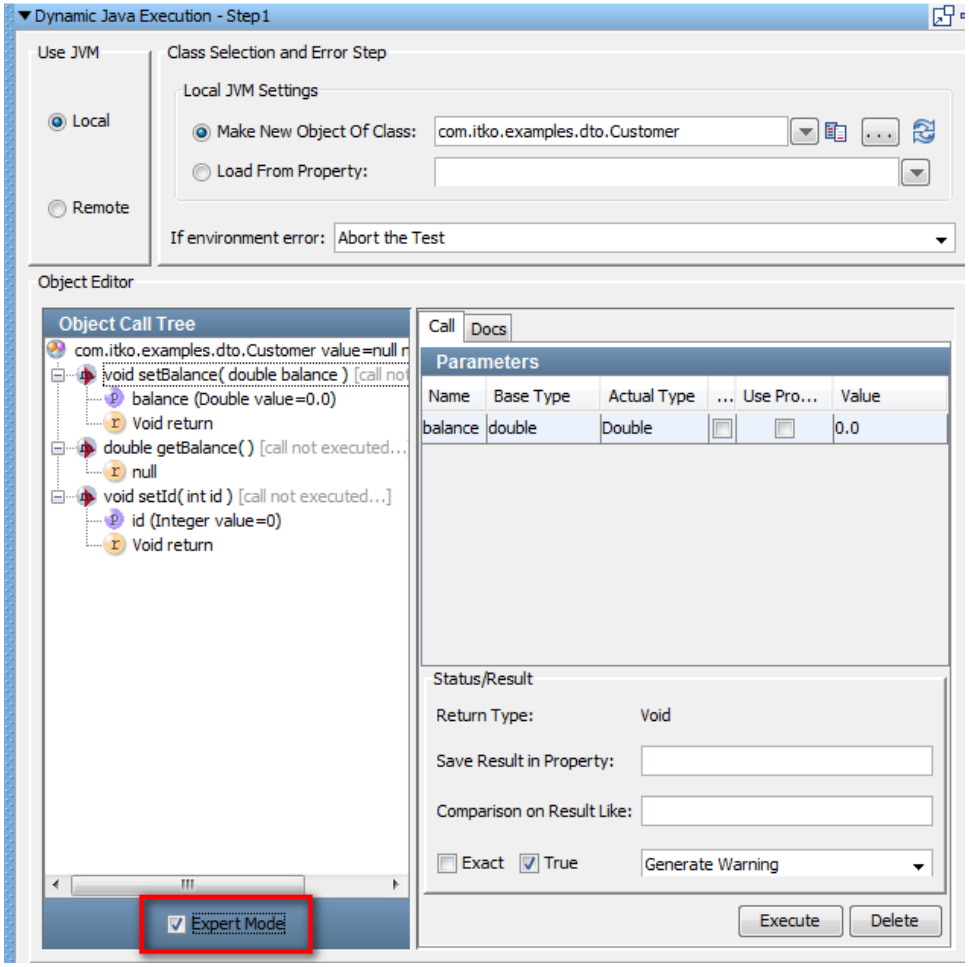
Deux modes d'édition sont disponibles : simple et expert.

Le mode simple est utile lorsque l'objet est un objet simple, comme un Bean Java avec un constructeur unique par défaut et plusieurs méthodes d'accesseur setter/getter. C'est l'objet de transfert de données classique (objet DTO). Ces objets sont communs comme entrée pour les appels de service Web. Avec des objets de ce type, vous pouvez basculer entre le mode Simple et Expert. Un objet de transfert de données qui contient un objet de transfert de données comme propriété peut être manipulé en mode Simple. Des exemples sont fournis ultérieurement.

Utilisez le mode Expert pour des objets plus complexes, comme les objets qui ont plusieurs constructeurs. Certains objets composites qui contiennent d'autres objets complexes ne peuvent pas utiliser le mode Simple, et si l'objet actuel requiert le mode Expert, l'option de mode Simple est désactivée.

Tous les graphiques affichés précédemment utilisaient le mode Simple.

Le graphique suivant représente l'exemple utilisé dans le graphique précédent, mais avec le mode Expert sélectionné.



Un nouveau panneau Status/Result (Statut/résultat) s'ouvre, comme affiché.

Vous pouvez ajouter des filtres intégrés (Save Result Property In (Enregistrer le résultat dans la propriété)) et des assertions (Comparison On Result Like (Comparaison en cas de similitude du résultat)) dans l'éditeur d'objets.

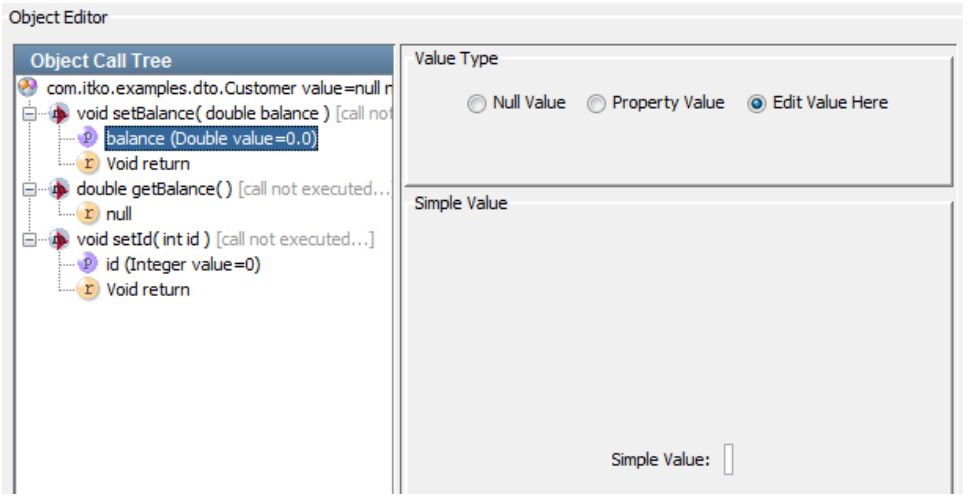
Remarque : Les filtres et les assertions intégrés qui sont appliqués ne sont pas affichés dans la liste de filtres ou d'assertions.

Plusieurs autres différences se manifestent dans les exemples ultérieurs.

Panneaux de paramètre d'entrée

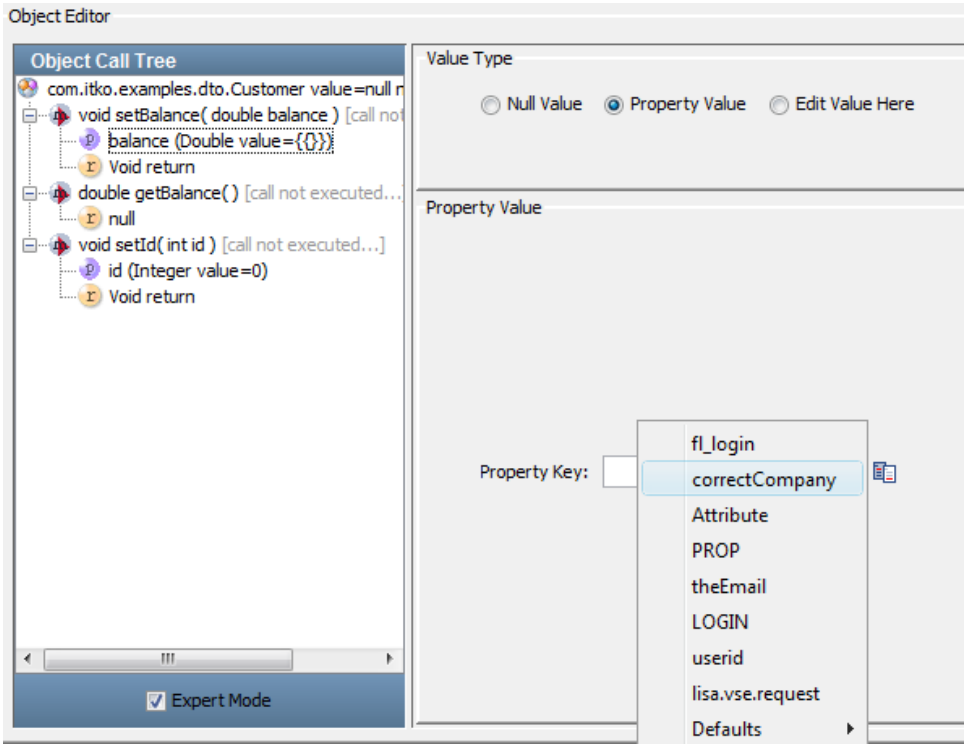
Si vous sélectionnez un paramètre d'entrée dans l'arborescence Object Call Tree, les onglets disponibles dans le panneau droit varient selon le type du paramètre d'entrée : primitives, chaînes ou objets.

Pour les paramètres d'entrée qui sont des primitives et des chaînes, le panneau suivant est affiché.



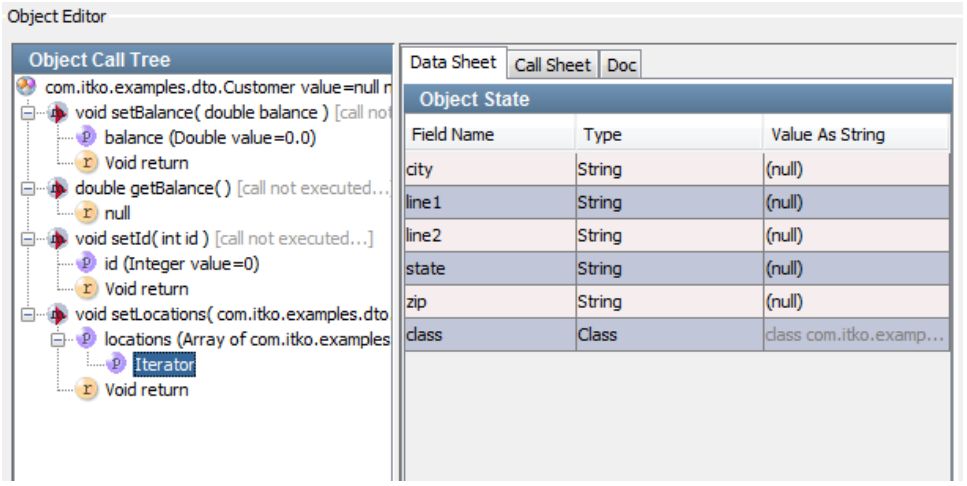
Vous pouvez modifier la valeur dans ce panneau dans le champ Simple Value (Valeur simple).

Pour utiliser une propriété comme valeur, cliquez sur Property Value (Valeur de la propriété) pour afficher le panneau suivant.



Pour ouvrir les clés de propriété disponibles, saisissez le nom de la propriété, utilisez le menu déroulant ou cliquez sur List (Liste) .

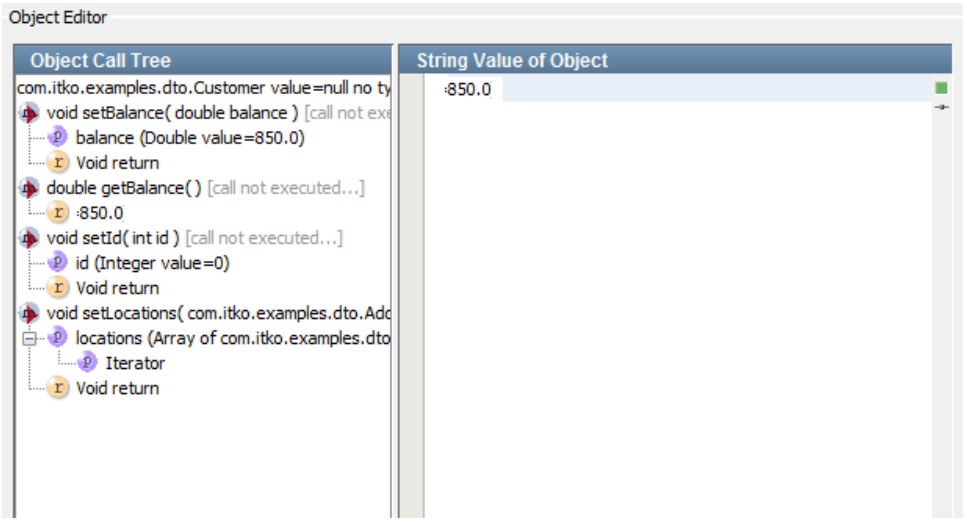
Pour les paramètres d'entrée qui sont des objets, le panneau suivant est affiché.



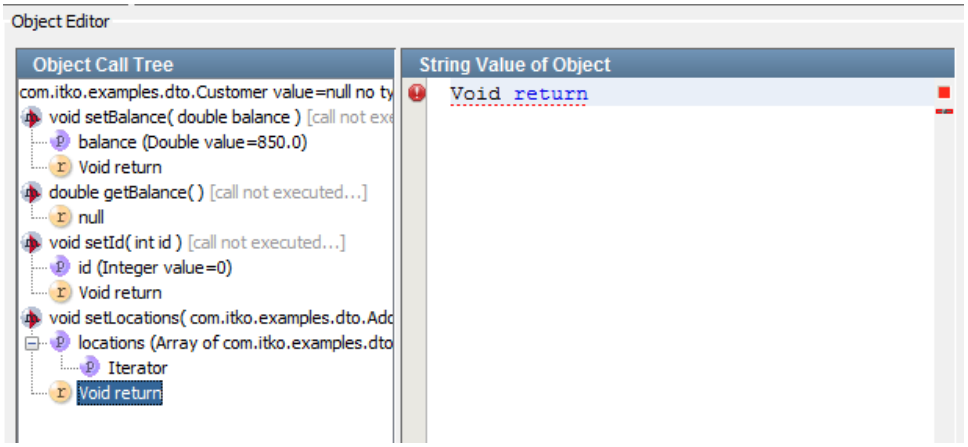
Panneaux de valeur renvoyée

Si vous sélectionnez une valeur renvoyée dans l'arborescence Object Call Tree, les onglets disponibles dans le panneau droit varient selon le type du paramètre d'entrée.

Pour les paramètres d'entrée qui sont des primitives et des chaînes, le panneau suivant est affiché.



Pour les paramètres d'entrée qui sont des objets, le panneau suivant est affiché.

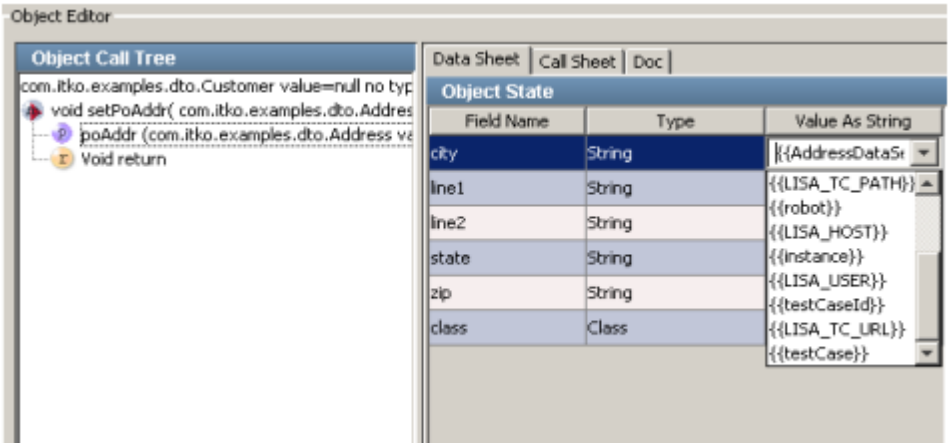


Utilisation d'ensembles de données dans l'éditeur Complex Object Editor (Editeur d'objets complexes)

Une méthode très répandue de fournir des données à un objet DTO Java consiste à utiliser un ensemble de données.

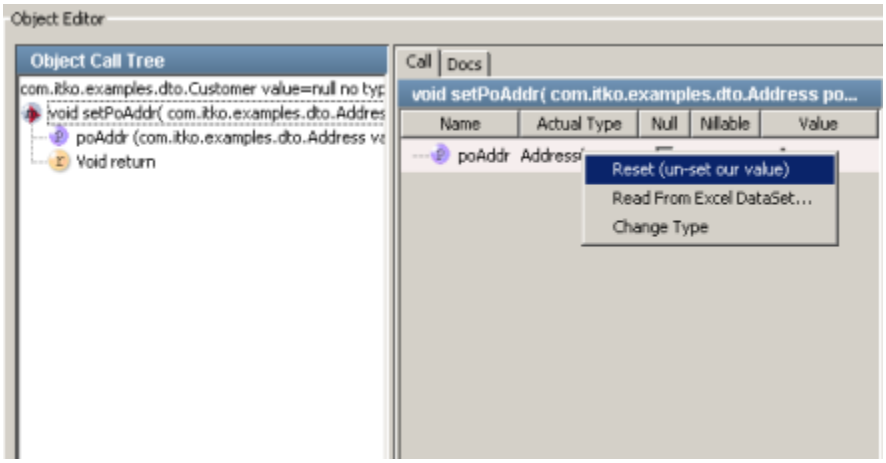
L'ensemble de données Read DTOs from Excel File (Lire les objets DTO à partir d'un fichier Excel) est fourni à cet effet.

Si l'ensemble de données Excel existe, vous pouvez entrer la propriété qui contient l'ensemble de données comme valeur pour l'objet DTO.

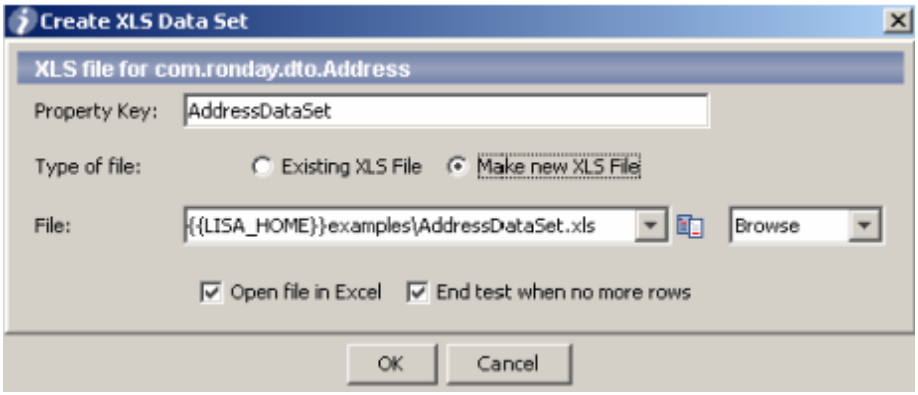


Pour initialiser la création d'un ensemble de données d'objet de transfert de données à partir de l'éditeur d'objets :

1. Lorsqu'un objet DTO s'affiche dans la liste d'appels, cliquez avec le bouton droit de la souris sur Name (Nom) ou Actual Type (Type réel) pour ouvrir un menu.



2. Sélectionnez l'ensemble de données Read From Excel Data Set (Lire à partir d'un ensemble de données Excel) pour afficher la fenêtre suivante.



3. Les paramètres de cette fenêtre sont requis pour initialiser la création de l'ensemble de données.

Entrez les paramètres suivants.

Property Key (Clé de la propriété)

Propriété qui stocke les valeurs actuelles à partir de l'ensemble de données.

Type of File (Type de fichier)

Sélectionnez cette option s'il s'agit d'un fichier XLS existant ou pour créer un nouveau fichier XLS.

File (Fichier)

Nom du fichier Excel qui sert de modèle pour les données d'objet DTO.

Open file in Excel (Ouvrir le fichier dans Excel)

Permet d'ouvrir la feuille de calcul dans Excel.

End test when no more rows (Terminer le test lorsque plus aucune ligne n'est disponible)

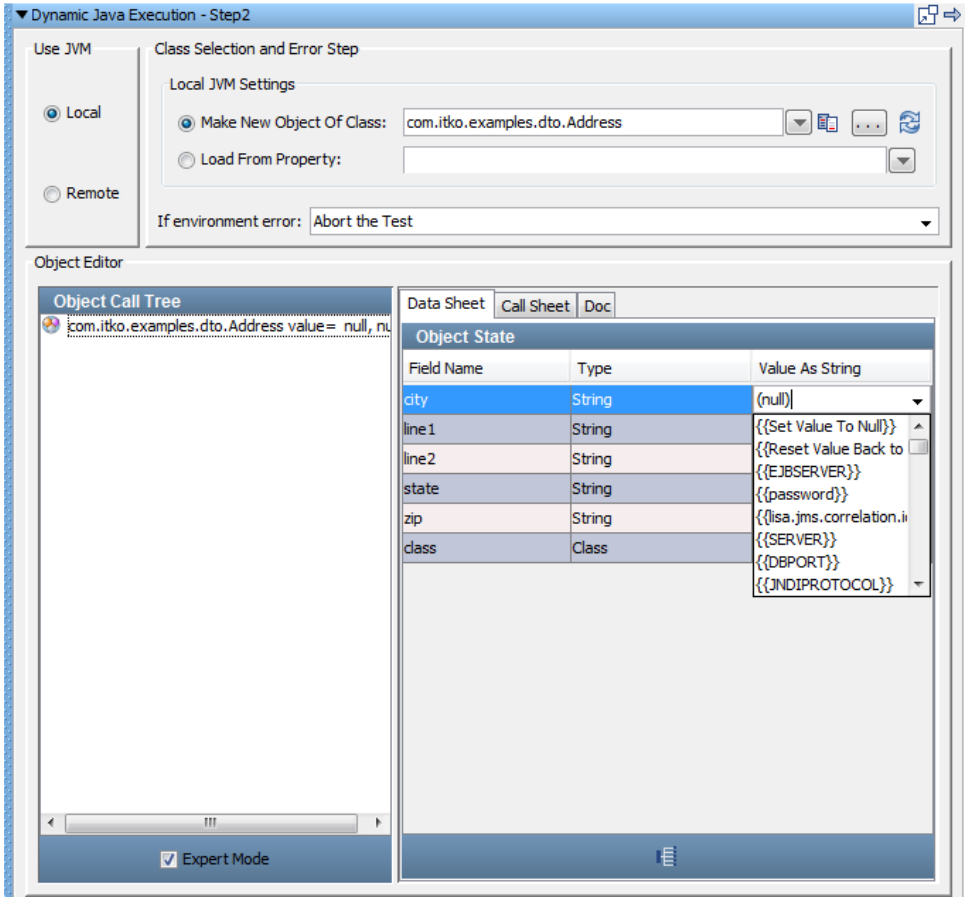
Permet de mettre fin au test une fois que toutes les lignes ont été lues.

Scénarios d'utilisation des objets simples

Les exemples suivants sont basés sur les classes Java standard pour les objets simples. Les classes utilisées proviennent du serveur de démonstration inclus avec DevTest et sont simples à reproduire dans votre environnement.

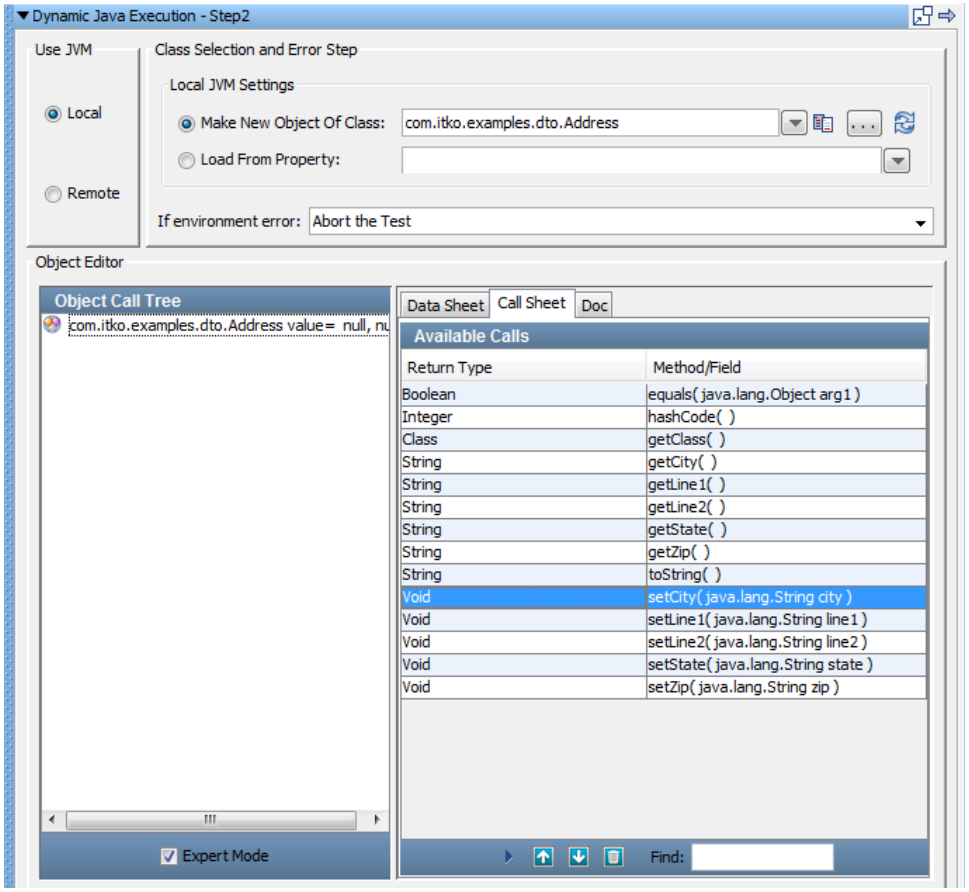
Scénario d'objet DTO simple 1

Un objet de transfert de données com.itko.examples.dto.Address simple a été chargé dans l'éditeur Complex Object Editor (Editeur d'objets complexes) à l'aide d'une étape Dynamic Java Execution (Exécution Java dynamique). La classe Address a des propriétés simples uniquement.



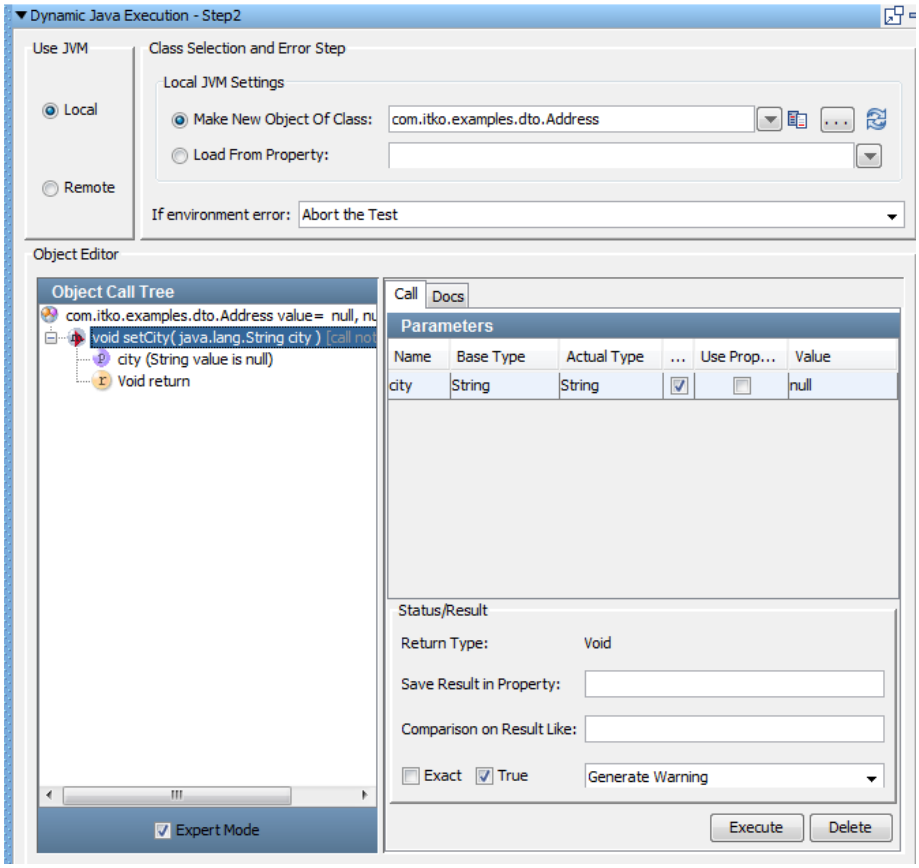
Pour un objet de transfert de données simple, vous pouvez entrer des valeurs de paramètre dans le panneau Data Sheet (Feuille de données) comme valeurs fixes ou propriétés. Dans l'exemple précédent, vous pouvez indiquer que le paramètre city est égal à la propriété currentCity.

Pour entrer des paramètres à l'aide des accesseurs setter d'objet DTO dans le panneau Call Sheet (Feuille d'appel) :



1. Dans l'onglet Call Sheet, sélectionnez un accesseur getter, comme setCity(java.lang.String city) et cliquez sur  Add Method (Ajouter une méthode).

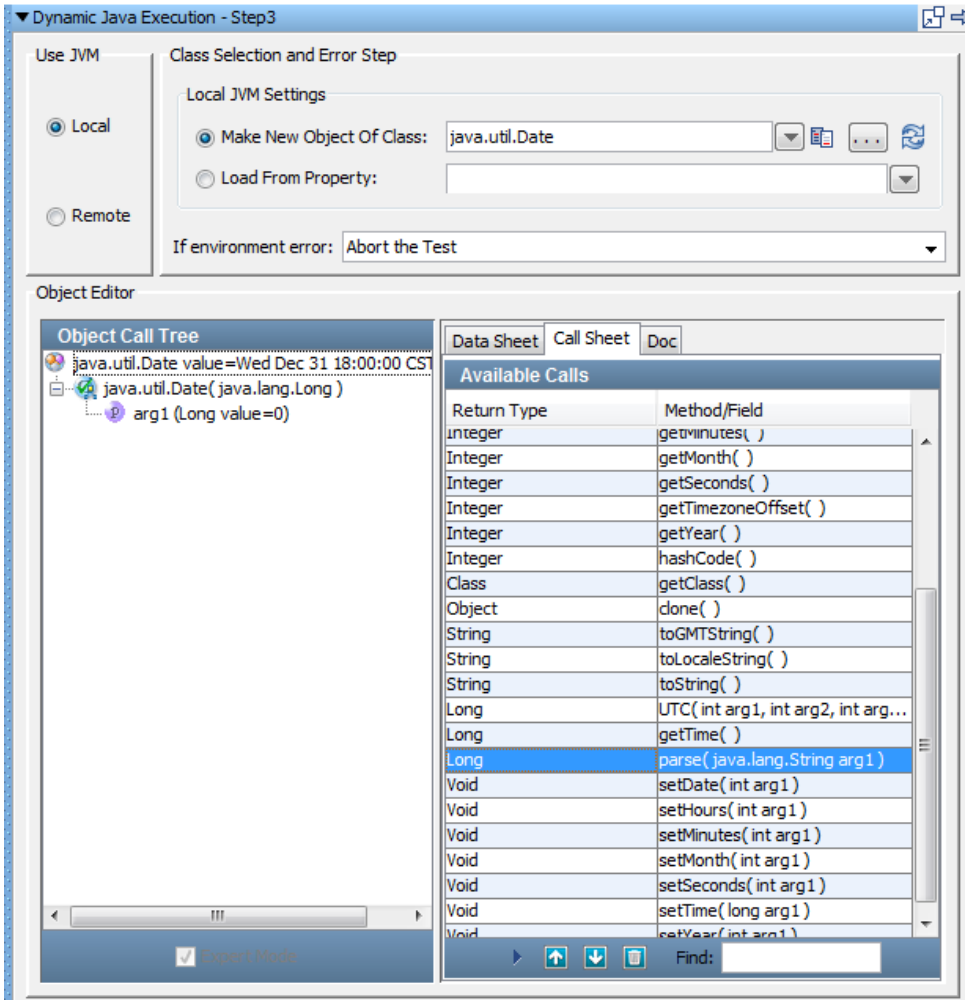
   La méthode s'exécute et l'éditeur Complex Object Editor s'ouvre dans l'onglet Call (Appel).



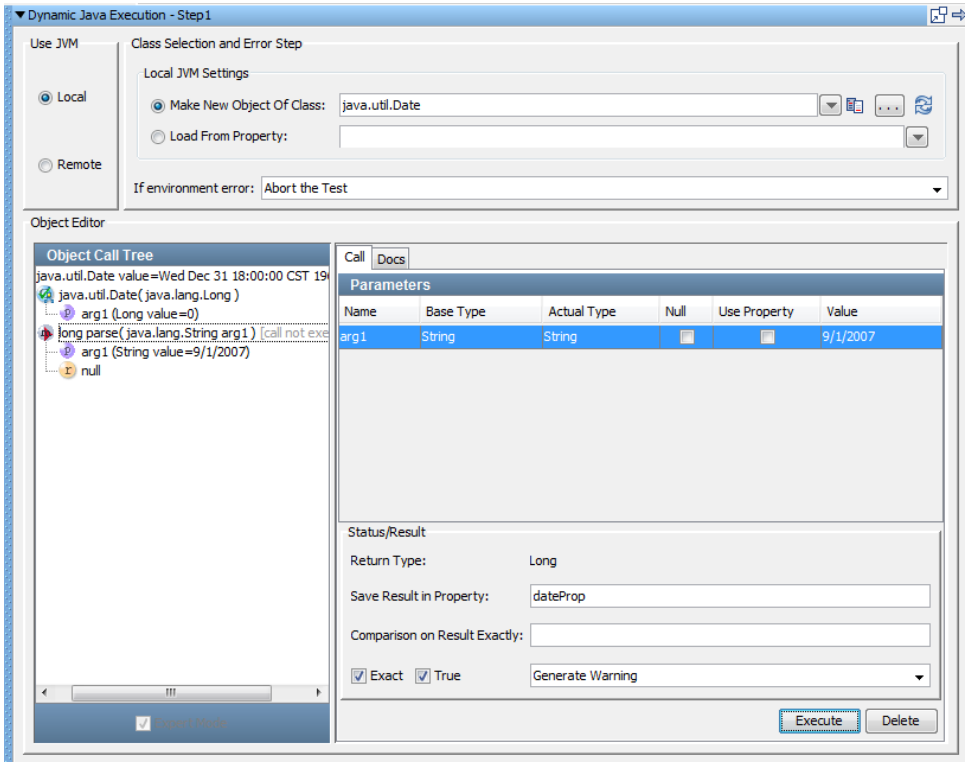
2. Entrez la valeur de paramètre sous la forme d'une valeur fixe ou d'une propriété. Utilisez la syntaxe de propriété DevTest : propname.
3. Cliquez sur Execute (Exécuter) pour appeler la méthode.
4. Répétez cette procédure pour définir d'autres propriétés d'objet DTO.

Scénario d'objet Java simple 2

Un objet Java simple, **java.util.Date**, a été chargé dans l'éditeur Complex Object Editor (Editeur d'objets complexes) à l'aide d'une étape Dynamic Java Execution (Exécution Java dynamique).



Dans ce cas, le mode Expert doit être utilisé, car le type Date n'est pas un objet de transfert de données. En fait, l'éditeur Complex Object Editor impose le mode Expert.



Mais vous pouvez toutefois entrer des valeurs de paramètre et appeler des méthodes. Dans l'exemple précédent, la méthode d'analyse, qui requiert un paramètre d'entrée, est exécutée. Vous avez entré ce paramètre en tant que valeur de chaîne, 9/1/2007.

Remarque : En mode Expert, utilisez les cases à cocher Null ou Use Property (Utiliser une propriété) pour définir le type de paramètre. Si aucune n'est sélectionnée, la valeur fixe est entrée. Il est recommandé de ne pas utiliser la syntaxe de propriété {{ }} dans ce cas. Même si vous entrez une propriété plutôt qu'une valeur de chaîne, saisissez uniquement le nom de propriété.

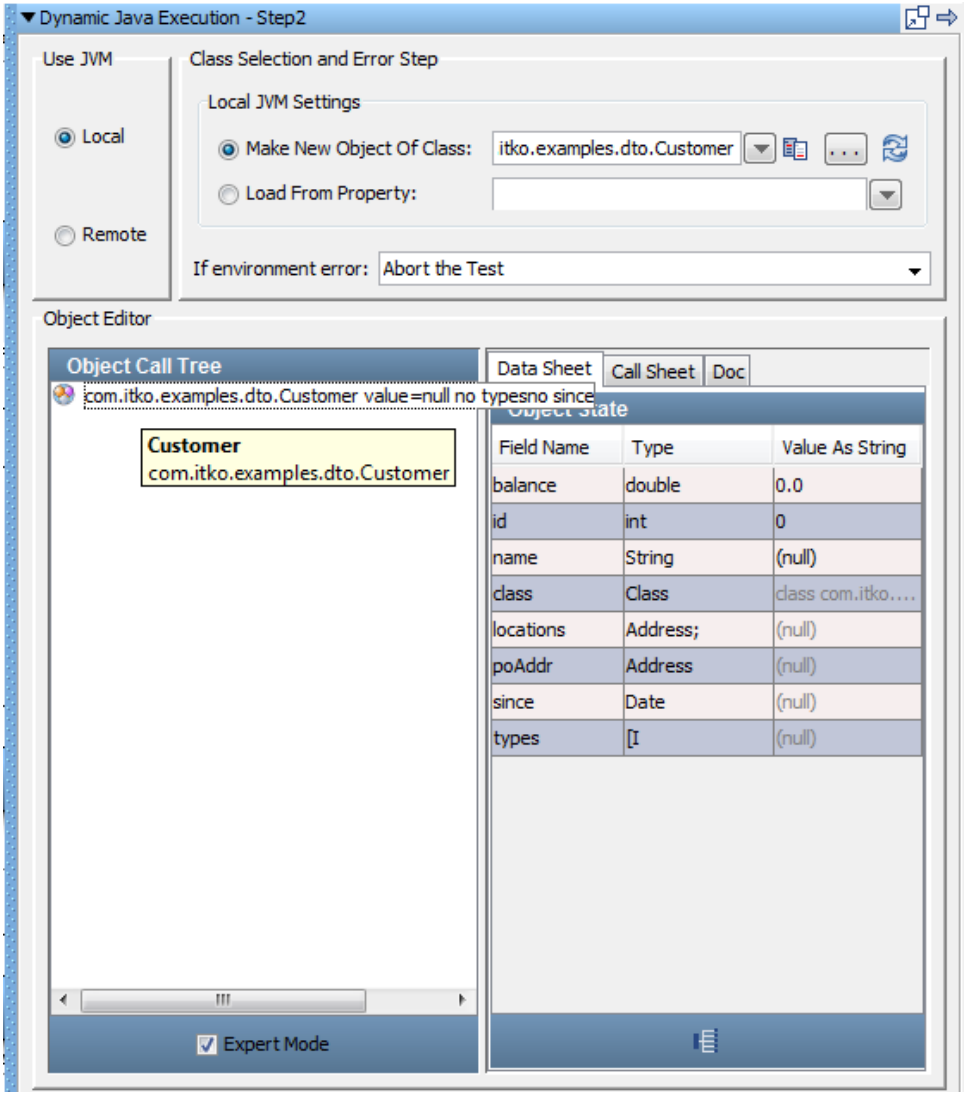
Comme le mode Expert est activé, vous pouvez ajouter des filtres et des assertions intégrés.

Scénarios d'utilisation des objets complexes

Les exemples suivants sont basés sur les classes Java standard pour les objets complexes. Les classes utilisées proviennent du serveur de démonstration inclus avec DevTest et sont simples à reproduire dans votre environnement.

Scénario d'objet DTO complexe 1

Un objet DTO complexe, com.itko.examples.dto.Customer, a été chargé dans l'éditeur Complex Object Editor (Editeur d'objets complexes) à l'aide d'une étape Dynamic Java Execution (Exécution Java dynamique).



Cet objet de transfert de données est complexe, car ses propriétés ne sont pas toutes des valeurs simples, comme des primitives ou des chaînes. Toutefois, en raison de sa structure d'objet DTO, vous pouvez encore utiliser le mode Simple, si vous le souhaitez.

Vous devez spécifier des valeurs pour chaque propriété avant de pouvoir utiliser l'objet Customer.

locations

Tableau d'objets Address

poAddr

Un objet Address

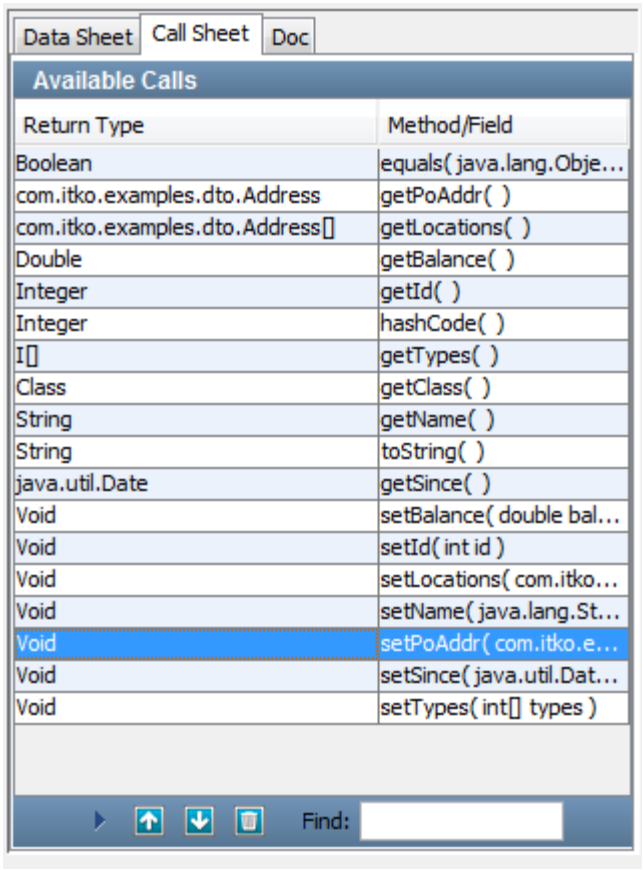
since

Un objet Date Java

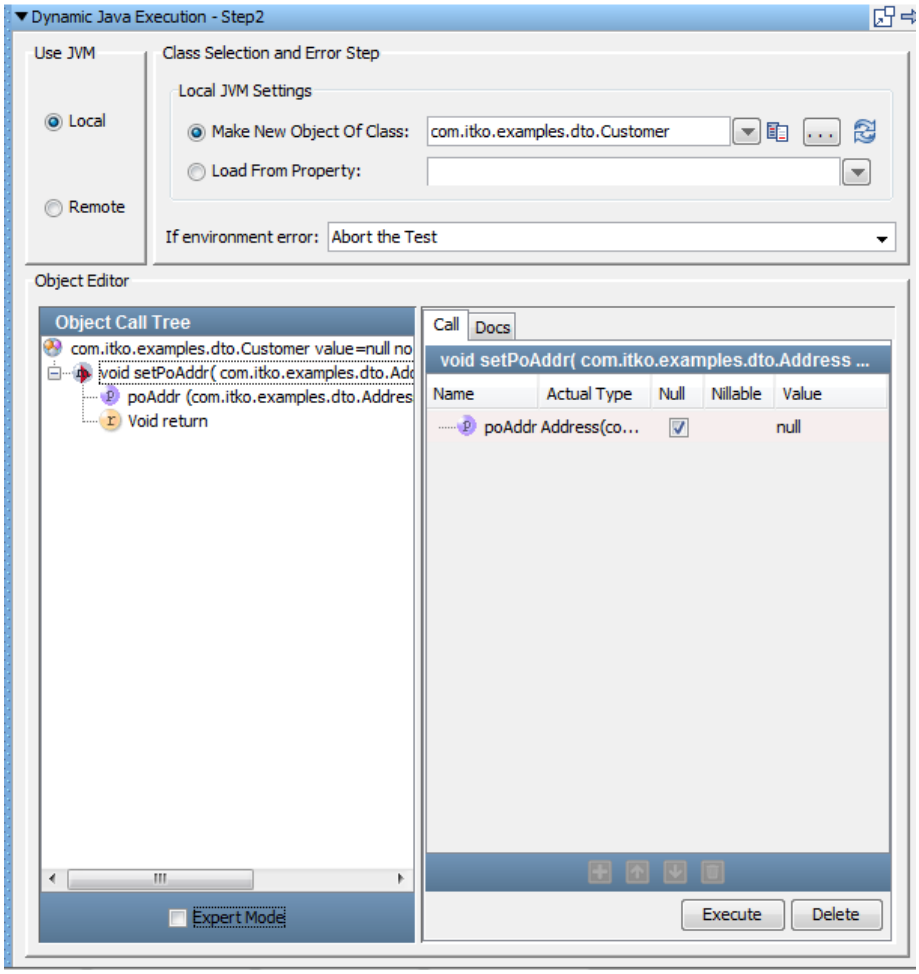
types

Tableau de nombres entiers

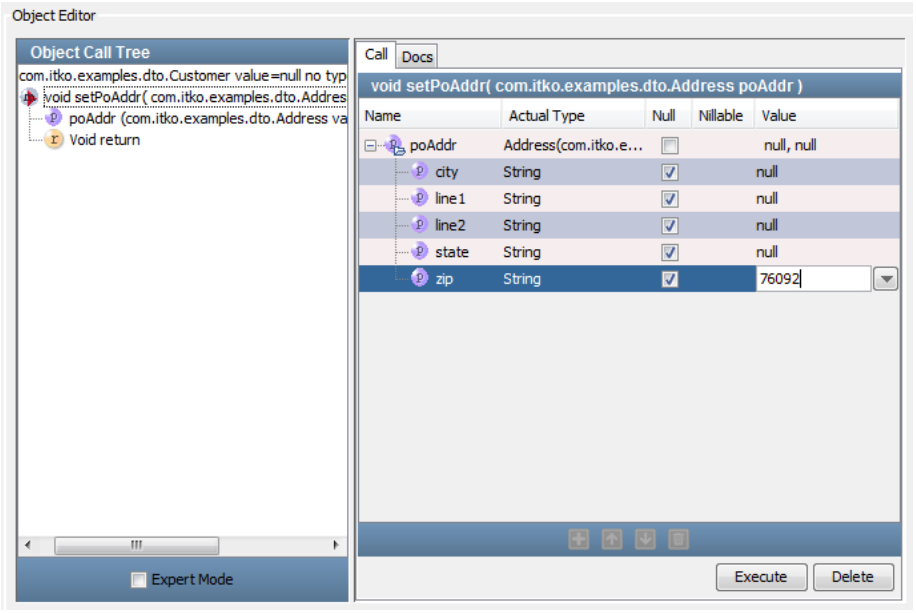
1. En commençant par l'objet poAddr, identifiez la méthode setPoAddr dans la feuille Call Sheet (Feuille d'appel)



2. Sélectionnez la méthode setPoAddr et double-cliquez dessus ou cliquez sur  Add Method (Ajouter une méthode) pour exécuter cette méthode.

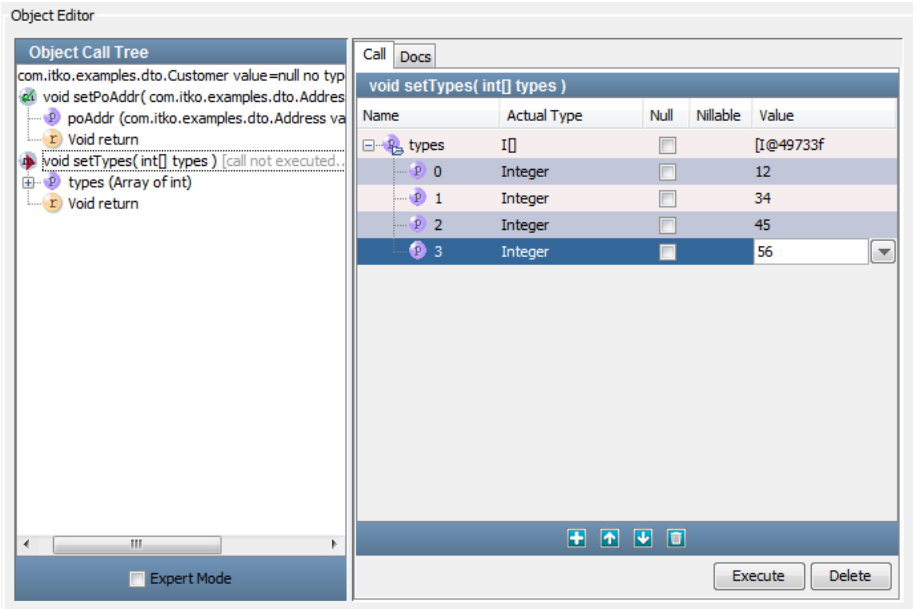


3. L'objet de transfert de données active l'utilisation du mode Simple. Ne cochez pas la case Expert Mode (Mode Expert). La propriété poAddress est identifiée comme étant de type Address.
4. En mode Simple, lorsque vous effacez le paramètre Null, l'objet Address est développé afin d'afficher ses propriétés. Comme dans le scénario d'objet DTO simple illustré précédemment, cet objet Address a des propriétés simples. Vous pouvez les entrer comme des valeurs ou des propriétés dans la colonne Value (Valeur).



5. Cliquez sur Execute (Exécuter) pour appeler la méthode setPoAddr.

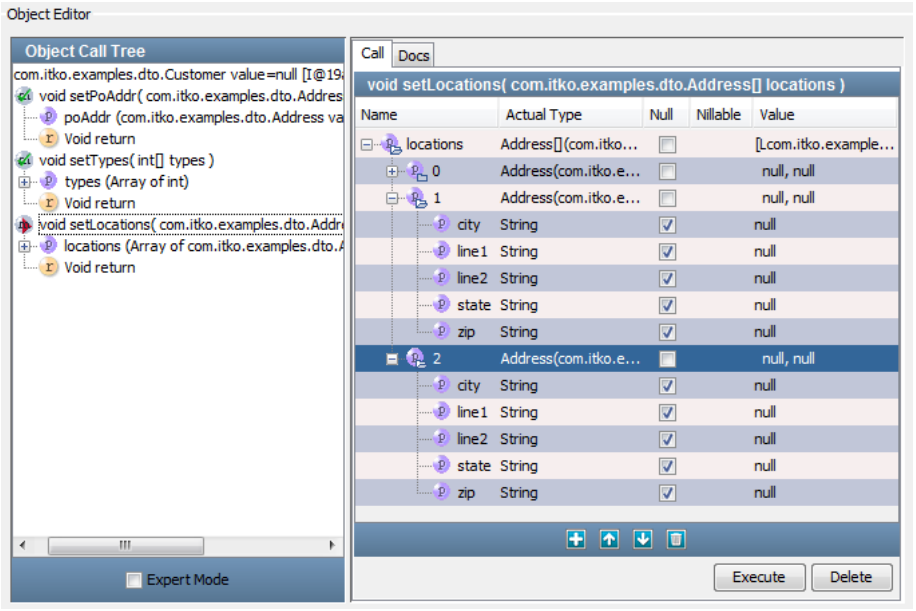
6. Sélectionnez la méthode setTypes dans la feuille Call Sheet et cliquez sur  Invoke Method (Appeler la méthode).



7. **Types** est un tableau de nombres entiers. Pour ajouter autant de nombres entiers que le tableau le requiert, cliquez sur  Add (Ajouter) dans la partie inférieure. Dans l'exemple précédent, vous avez ajouté quatre éléments et entré des valeurs pour chacun.

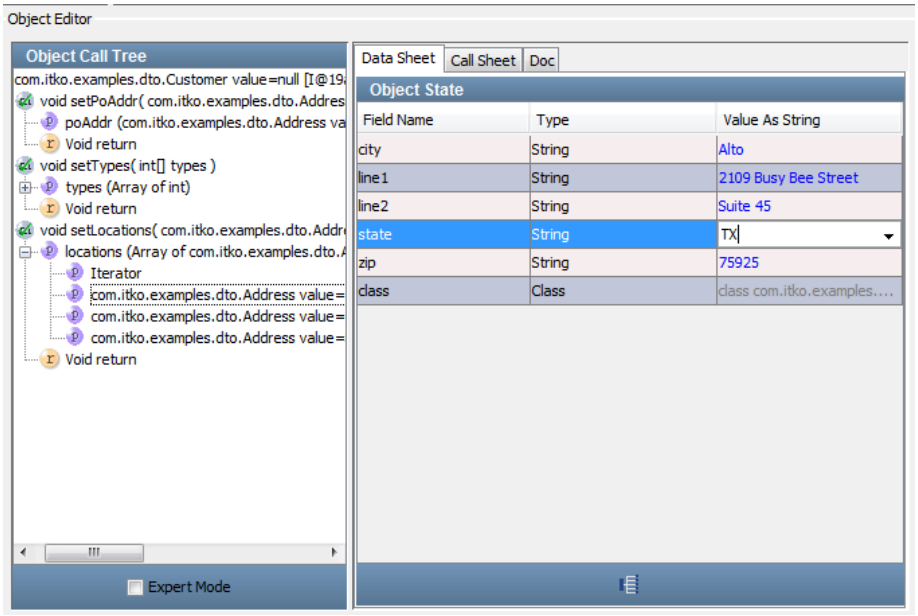
8. Cliquez sur Execute pour appeler la méthode setTypes.

9. Sélectionnez la méthode setLocations dans la feuille Call Sheet et cliquez sur  Invoke Method.



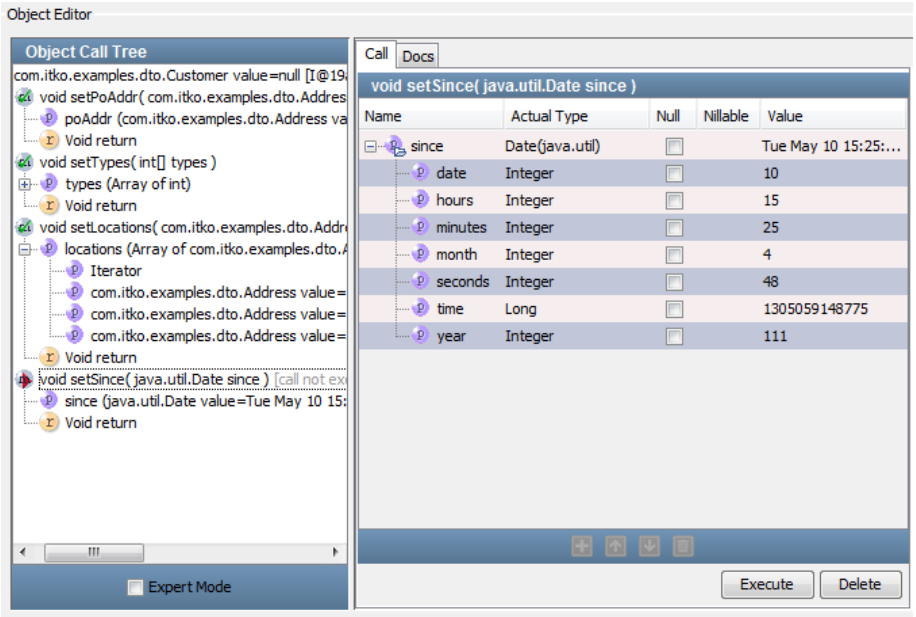
10. **Locations** est un tableau d'objets Address. Cliquez sur  Add pour ajouter autant d'éléments (de type Address) que requis.

11. Dans l'exemple précédent, vous avez ajouté trois objets Address. Deux d'entre eux sont complets et vous êtes prêt pour le développement du troisième objet Address pour entrer des valeurs de propriété. Lorsque vous avez terminé, cliquez sur Execute pour appeler la méthode setLocations. Remarquez que vous pouvez cliquer sur un des éléments Location dans l'arborescence Object Call Tree (Arborescence des appels d'objet) pour afficher et modifier des propriétés dans l'onglet Data Sheet (Feuille de données).



Cela est vrai pour toutes les propriétés qui sont répertoriées dans l'arborescence Object Call Tree.

12. Sélectionnez la méthode getSince dans la feuille Call Sheet et cliquez sur  Invoke Method (Appeler la méthode).



Les paramètres d'entrée pour l'objet de données sont affichés et peuvent être des valeurs données.

13. Cliquez sur Execute (Exécuter).

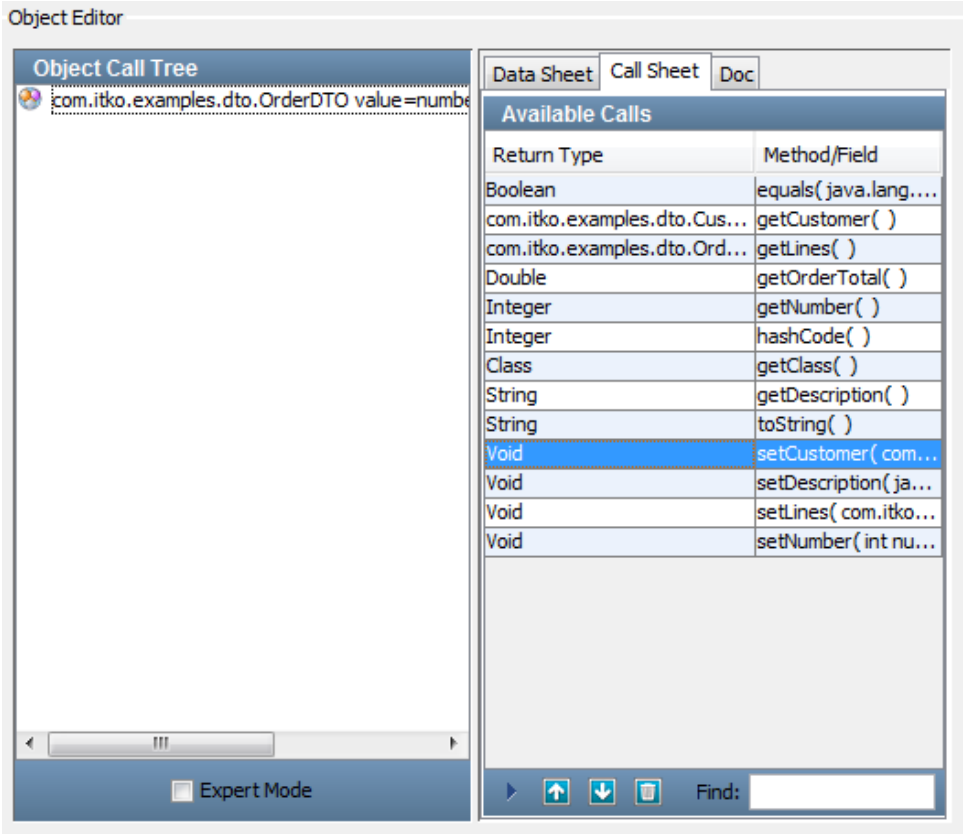
L'objet Customer est complètement spécifié et peut être utilisé dans le scénario de test.

Scénario d'objet DTO complexe 2

Le dernier scénario s'appuie sur les trois derniers scénarios.

L'objet de transfert de données com.itko.examples.dto.OrderDTO comprend un objet Customer, comme une de ses propriétés. Dans ce scénario, vous verrez comme il est facile de créer un objet Customer en mode simple sans devoir appeler des méthodes setter.

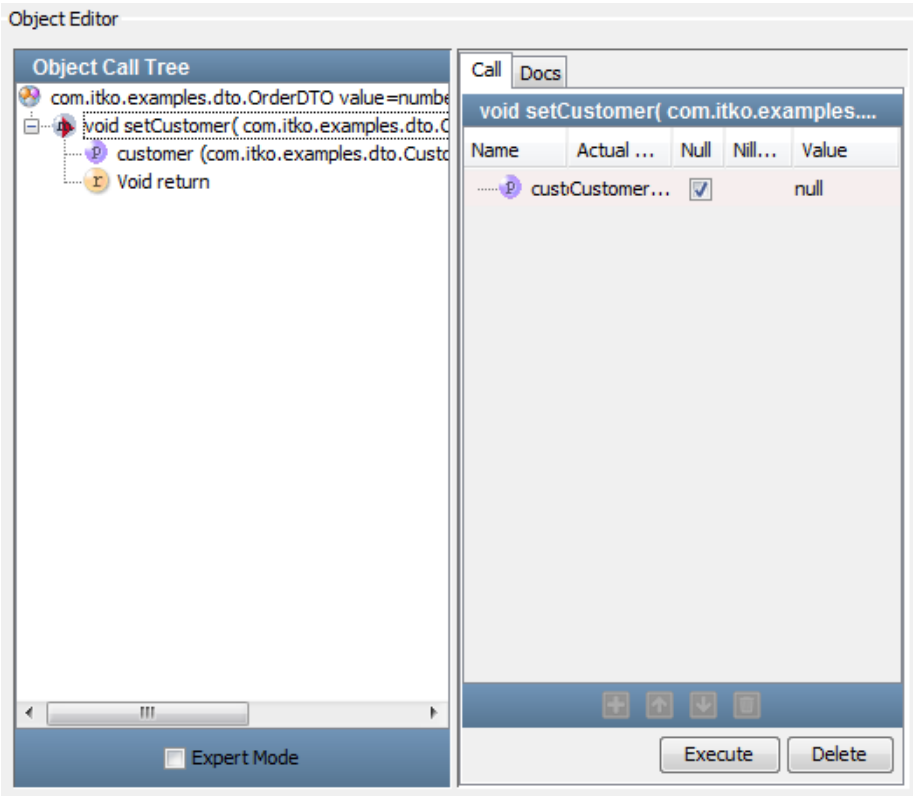
L'objet OrderDTO a été chargé dans l'éditeur Complex Object Editor (Editeur d'objets complexes) à l'aide d'une étape Dynamic Java Execution (Exécution Java dynamique).



Vous pouvez encore utiliser le mode Simple pour l'objet OrderDTO.

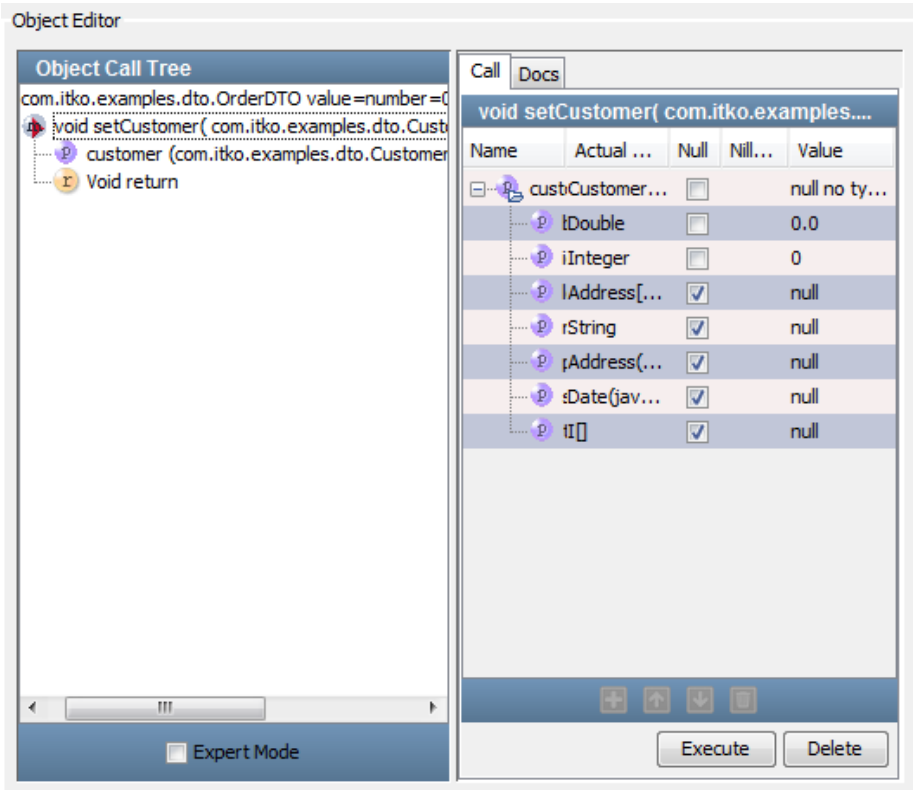
Procédez comme suit:

1. Sélectionnez la méthode setCustomer dans la feuille Call Sheet (Feuille d'appel) et cliquez sur  Invoke Method (Appeler la méthode). Comme prévu, le paramètre d'entrée est un objet Customer.

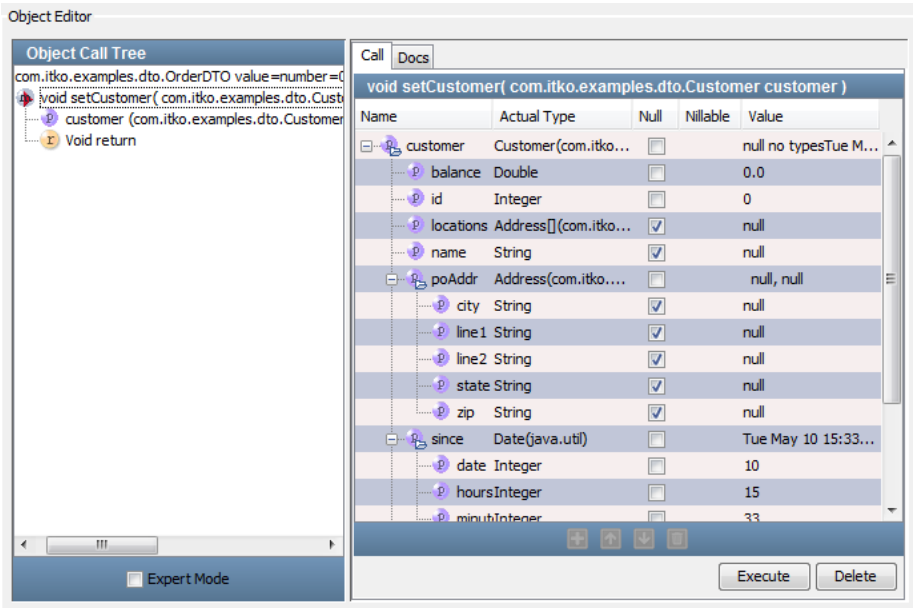


2. Décochez la case dans la colonne Null.

   La ligne Customer se développe et affiche ses propriétés.



Vous pouvez modifier toutes les propriétés dans cette fenêtre. Vous pouvez ajouter les propriétés Integer et String dans la colonne Value (Valeur). Les propriétés restantes se développent pour afficher leurs propriétés lorsque vous décochez la case Null pour la propriété. Si la propriété est un objet unique, elle se développe et affiche ses propriétés. S'il s'agit d'un tableau ou d'une collection, vous pouvez ajouter le nombre approprié d'éléments. Ce diagramme présente un cliché du processus de modification.



La propriété locations a deux éléments et la propriété types en a trois. L'objet poAddr de type Address est développé et affiche ses propriétés de chaîne simples.

Génération d'étapes de test

Une étape de test est un élément du flux de travaux d'un scénario de test qui représente une action de test unique effectuée. Les étapes de test sont réparties dans deux catégories principales :

- La plupart des étapes de test interviennent sur le système testé et évaluent la réponse. Par exemple, des étapes de test communes consistent à tester une méthode EJB (Enterprise JavaBean), un service Web ou un message envoyer via un fournisseur de services de messagerie.
- Une deuxième catégorie d'étapes de test effectue des fonctions d'utilitaire, comme la conversion de données, la manipulation de données (codage, etc.), la journalisation ou l'écriture d'informations dans des fichiers.

Remarque : DevTest ne prend pas en charge l'envoi de flux d'E/S à partir d'étapes de test, de propriétés ou les deux.

Les deux catégories d'étapes sont utilisées pour la génération d'un scénario de test.

Cette section comprend les rubriques suivantes :

Ajout d'une étape de test

Pour ajouter une étape de test :

Effectuez l'une des actions suivantes :

- Cliquez sur Add Step (Ajouter une étape) dans la barre d'outils.
- Dans le menu principal, sélectionnez Commands (Commandes), Create New Step (Créer une étape).

Vous pouvez également ajouter une étape à un emplacement spécifique dans le flux de travaux en cliquant avec le bouton droit de la souris sur l'étape dans le flux de travaux pour ouvrir un menu contextuel. Cliquez sur Add Step After (Ajouter une étape après) et sélectionnez l'étape de test appropriée.

Informations complémentaires :

Exemple d'ajout d'étape de test (page 208)

Exemple d'ajout d'étape de test

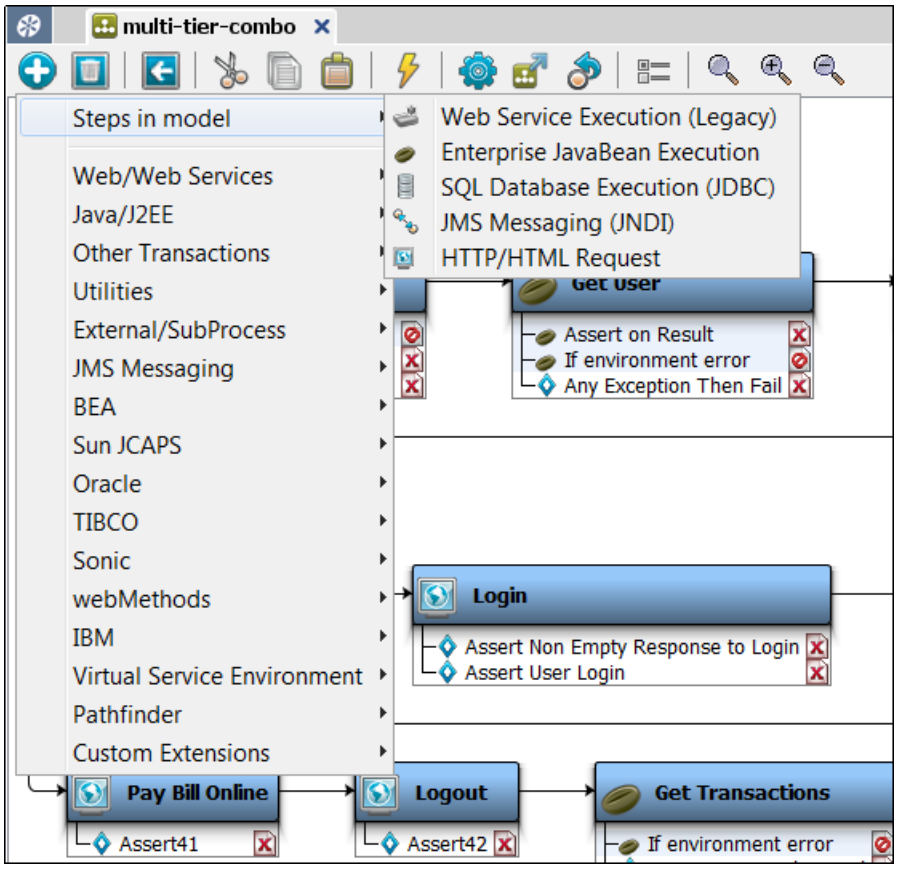
Dans cet exemple, vous ajoutez une étape au scénario de test à plusieurs niveaux du répertoire Examples (Exemples). Une nouvelle étape Dynamic Java Execution (Exécution Java dynamique) sera ajoutée au scénario de test à plusieurs niveaux, après l'étape Get User (Obtenir le nom d'utilisateur).

Procédez comme suit:

1. Cliquez sur Add Step (Ajouter une étape).

   Un panneau qui répertorie les étapes de test communes s'ouvre.

   - Si certaines étapes ont déjà été créées dans le scénario de test, comme pour ce scénario de test, le panneau affiche la liste Steps in Model (Etapes du modèle) dans la partie supérieure. La liste Steps in Model répertorie toutes les étapes figurant dans le scénario de test.
   - Dans le cas d'un nouveau scénario de test, cette liste est vide. Lorsque vous ouvrez le scénario de test à plusieurs niveaux, les étapes qui le constituent sont affichées dans la liste Steps in Model.



2. Sélectionnez la catégorie principale de l'étape à configurer (par exemple, Web/Web Services (Web/Services Web), Java/J2EE, Utilities (Utilitaires), etc.).

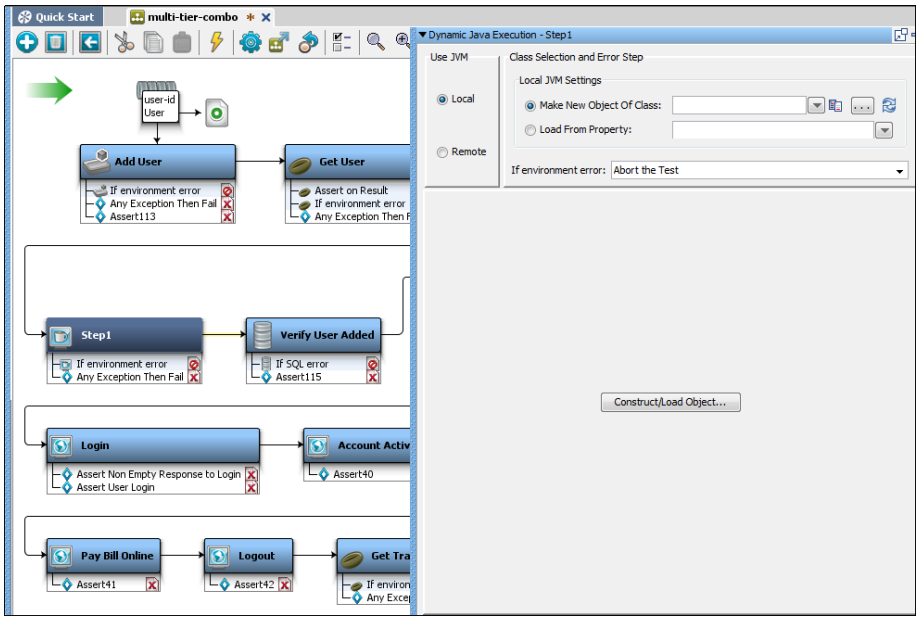
La sous-catégorie s'ouvre.

3. Pour ajouter une étape au scénario de test, cliquez sur l'étape.

   L'éditeur d'étapes pour le scénario de test sélectionné s'ouvre. Chaque type d'étape a son propre éditeur d'étapes.

4. Pour ajouter une étape Dynamic Java Execution (Exécution Java dynamique) après l'étape Get User (Obtenir le nom de l'utilisateur), cliquez avec le bouton droit de la souris sur l'étape Get User pour ouvrir le menu et sélectionnez l'étape Dynamic Java Execution.

   L'étape est ajoutée et l'éditeur de l'étape Dynamic Java Execution s'ouvre.

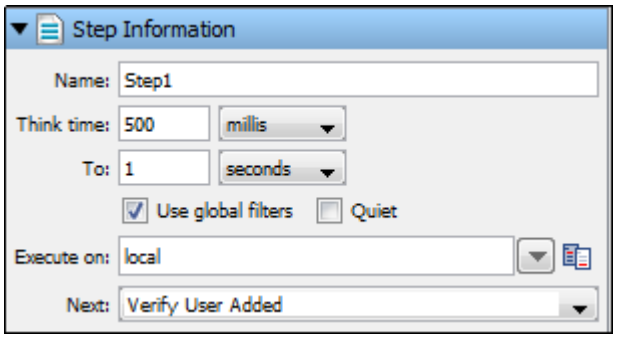


Ajout des informations d'étape

Procédez comme suit:

1. Pour ajouter des informations de base sur une étape, ouvrez et développez l'onglet Step Informations (Informations sur l'étape) dans l'arborescence Elements (Eléments) du panneau droit.

   L'éditeur Step Informations s'ouvre.



2. Entrez les paramètres suivants :

   Name (Nom)

   Nom de l'étape. Vous pouvez renommer l'étape dans cette zone de texte.

   Think Time (Délai de réflexion)

   Durée d'attente du scénario de test avant l'exécution de l'étape. Le délai de réflexion permet à DevTest de simuler le temps que prend un utilisateur pour décider de l'action suivante à effectuer. Pour spécifier le mode de calcul du délai :

   a. Sélectionnez l'unité de temps en cliquant sur la liste déroulante appropriée (millisecondes, secondes, minutes).

   b. Saisissez une valeur de début dans le champ Think Time (Délai de réflexion).

   c. Dans le champ To (A), entrez une valeur de fin et un indicateur de temps.

      DevTest sélectionne un délai de réflexion aléatoire dans la plage spécifiée. Par exemple, pour simuler un délai de réflexion d'utilisateur pour un intervalle aléatoire compris entre 500 millisecondes et 1 seconde, saisissez 500 millis et 1 seconds.

   Use Global Filters (Utiliser des filtres globaux)

   Pour indiquer à l'étape d'utiliser des filtres globaux, cochez cette case. Pour plus d'informations sur les filtres, consultez la rubrique Ajout d'un filtre (page 124) dans le *Manuel de l'utilisateur de CA Application Test.*

   Quiet (Silencieuse)

   Sélectionnez cette case à cocher de sorte que DevTest ignore cette étape pour les événements de temps de réponse et le calcul des performances.

Execute On (Exécuter au niveau de)

Spécifie le simulateur sur lequel l'étape s'exécute. Spécifiez des simulateurs appropriés pour des étapes qui doivent s'exécuter sur un ordinateur spécifique. Par exemple, lors de la lecture d'un fichier journal, l'étape doit être exécutée sur l'ordinateur sur lequel le journal réside.

Next (Suivant)

L'étape suivante à exécuter dans le test. Si l'étape termine l'exécution du scénario de test, vous ne pouvez pas spécifier d'étape suivante. Une assertion qui se déclenche dans cette étape remplace cette valeur.

Configuration des étapes de test

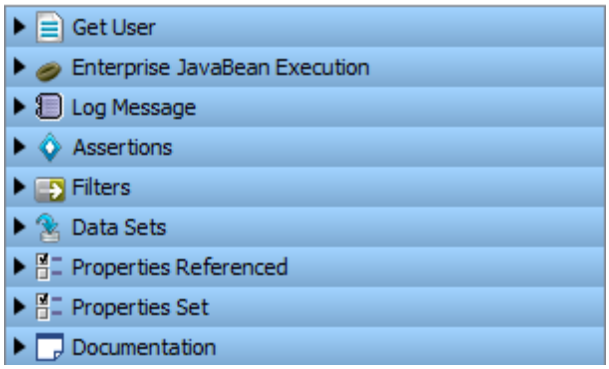
Procédez comme suit:

1. Ajoutez l'étape dans l'éditeur de modèles.

   Une fois que l'étape est ajoutée, un panneau d'étape de test différent s'ouvre dans l'arborescence Element (Elément).

2. Configurez l'étape de test en définissant les paramètres dans les éléments de configuration (assertions, filtres, ensembles de données, etc.).

Vous pouvez afficher les détails de chaque élément de scénario de test ou d'étape en cliquant sur la flèche d'élément  à côté des éléments de configuration pour le développer.



Step Information (Informations sur l'étape)

L'élément Step information (Informations sur l'étape) est le premier élément dans le panneau d'éléments et porte le nom de l'étape comme étiquette. Dans l'exemple précédent, l'élément Step Information est **Get User** (Obtenir le nom d'utilisateur). Vous pouvez changer le nom de l'étape de test après l'avoir développée. Définissez l'étape suivante à exécuter une fois que l'étape actuelle est terminée. Les autres options sont expliquées dans la rubrique Ajouter une étape de test (page 207).

Step Type Information (Informations sur le type d'étape)

Ce champ contient le titre du type d'étape sélectionné (Enterprise JavaBean Execution (Exécution d'un objet EJB) dans l'exemple). Chaque étape est différente et comporte ses propres conditions de configuration. Un éditeur personnalisé est donc disponible pour fournir les informations requises pour l'exécution de l'étape et son test (et probablement obtenir une réponse). Cet éditeur personnalisé s'ouvre lorsque l'élément est développé.

Log Message (Message de journal)

Ce message s'affiche lorsqu'une étape est ajoutée à DevTest et vous permet de définir le message de journal à afficher à l'issue de l'exécution de l'étape.

Assertions

Ce champ permet d'ajouter et de configurer une ou plusieurs assertions. Dans une configuration d'assertion, vous devez également définir l'étape suivante à exécuter lorsque l'assertion se déclenche.

Filters (Filtres)

Ce champ permet d'ajouter et de configurer des filtres. Les filtres sont ajoutés sous l'élément de filtre de chaque étape de test.

Data Sets (Ensembles de données)

Ce champ permet d'ajouter et de configurer un ou plusieurs ensembles de données qui s'appliquent à l'étape de test. Les ensembles de données sont déclenchés avant l'exécution d'une étape de test. Toutes les propriétés que l'ensemble de données définit sont disponibles pour l'étape de test.

Properties Referenced (Propriétés référencées)

Liste de propriétés en lecture seule que l'étape référence (lit).

Properties Set (Ensemble de propriétés)

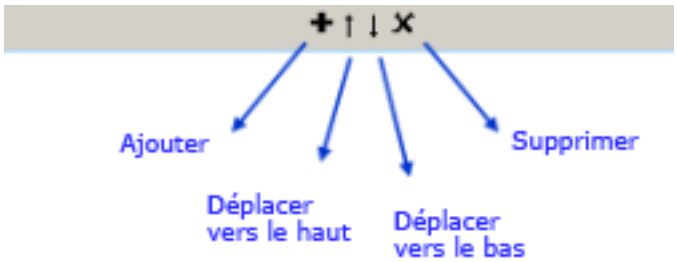
Liste de propriétés en lecture seule que l'étape définit (à laquelle l'étape affecte une valeur).

Documentation

Notes accompagnant l'étape de test.

Barre d'outils d'élément d'étape

Tous les éléments (assertions, filtres et ensembles de données) ont leur propre barre d'outils au bas de l'onglet Element (Elément).

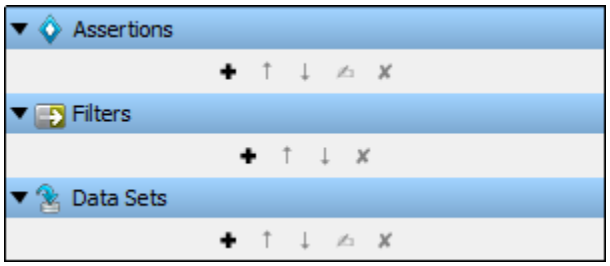


Vous pouvez ajouter un élément à une étape ou le supprimer en cliquant sur les icônes au bas de chaque élément.

- Pour des informations détaillées sur l'ajout de filtres, consultez la rubrique Ajout d'un filtre (page 124).
- Pour des informations détaillées sur l'ajout d'assertions, consultez la rubrique Ajout d'une assertion (page 143).

Ajout de filtres, d'assertions ou d'ensembles de données à une étape

Les filtres, les assertions et les ensembles de données sont ajoutés sous l'élément correspondant de chaque étape dans le panneau droit.



Cliquez sur Add (Ajouter) dans la barre d'outils pour ajouter un filtre, une assertion ou un ensemble de données.

- Pour des informations détaillées sur l'ajout et la configuration des filtres, consultez la rubrique Filtres (page 124).
- Pour des informations détaillées sur l'ajout et la configuration des assertions, consultez la rubrique Assertions (page 142).
- Pour des informations détaillées sur l'ajout et la configuration des ensembles de données, consultez la rubrique Ensembles de données (page 162).

Configuration de l'étape suivante

Affectation de l'étape suivante

Dans un flux de travaux de scénario de test, vous pouvez affecter le statut d'étape suivante à l'étape de test que vous sélectionnez.

Après avoir exécuté l'étape sélectionnée dans le flux de travaux, ce dernier se poursuivra à l'étape suivante définie pour l'exécution.

Vous pouvez configurer l'étape suivante de sorte à renvoyer vers d'autres étapes du flux de travaux ou à rediriger vers les actions suivantes :

- End the test (Arrêter le test)
- Fail the test (Faire échouer le test)
- Abort the test. (Interrompre le test)

Procédez comme suit:

1. Cliquez sur l'étape pour laquelle vous voulez définir une étape suivante.
2. Cliquez avec le bouton droit de la souris sur cette étape, sélectionnez For next step (Pour l'étape suivante) et cliquez sur l'étape suivante cible.

Le flux de travaux dans l'éditeur de modèles est modifié. Les informations du champ Next (Suivant) dans l'éditeur d'étapes sont également modifiées.

Vous pouvez également arrêter le test, le faire échouer ou l'interrompre.

Etape End (Fin)

L'étape End met fin à un flux de travaux. Elle est exécutée lorsqu'un flux de travaux se termine. Le scénario de test entier est réussi si cette étape est atteinte.

Etape Fail (Echec)

L'étape Fail signifie la fin d'un flux de travaux. Elle est exécutée lorsqu'un échec du flux de travaux se produit suite à un événement d'erreur. Le scénario de test entier est considéré comme un échec si cette étape est atteinte. L'étape Fail (Echec) est la valeur par défaut de nombreuses exceptions DevTest internes (par exemple, une exception EB). Toutefois, les assertions peuvent définir l'étape Fail (Echec) comme l'étape suivante pour mettre en échec un scénario de test.

Etape Abort (Interrompre)

L'étape Abort signifie également la fin d'un flux de travaux. Elle est exécutée lorsqu'un flux de travaux est brusquement interrompu. Le scénario de test entier est considéré comme interrompu (sans fin) si cette étape est atteinte.

Relecture jusqu'à une étape

Vous pouvez effectuer une relecture rapide d'un scénario de test jusqu'à l'une de ses étapes.

Pour effectuer une relecture jusqu'à une étape :

Sélectionnez l'étape, puis cliquez avec le bouton droit de la souris pour sélectionner Replay to here (Relire jusqu'à ce point).

Le scénario est relu jusqu'à l'étape sélectionnée.

Définition d'une étape de démarrage

Vous pouvez définir une étape de test comme étape de démarrage dans le flux de travaux.

L'étape de test de démarrage permet de lancer le flux de travaux de scénario de test.

Pour définir une étape de démarrage :

Sélectionnez l'étape, puis cliquez avec le bouton droit de la souris pour sélectionner Set as starter (Définir comme étape de démarrage).

L'étape sélectionnée est définie comme la première étape dans le flux de travaux. Cette option n'est pas disponible pour la première étape du scénario de test.

Génération d'avertissements et d'erreurs

Vous pouvez configurer deux autres types de tests comme étapes suivantes.

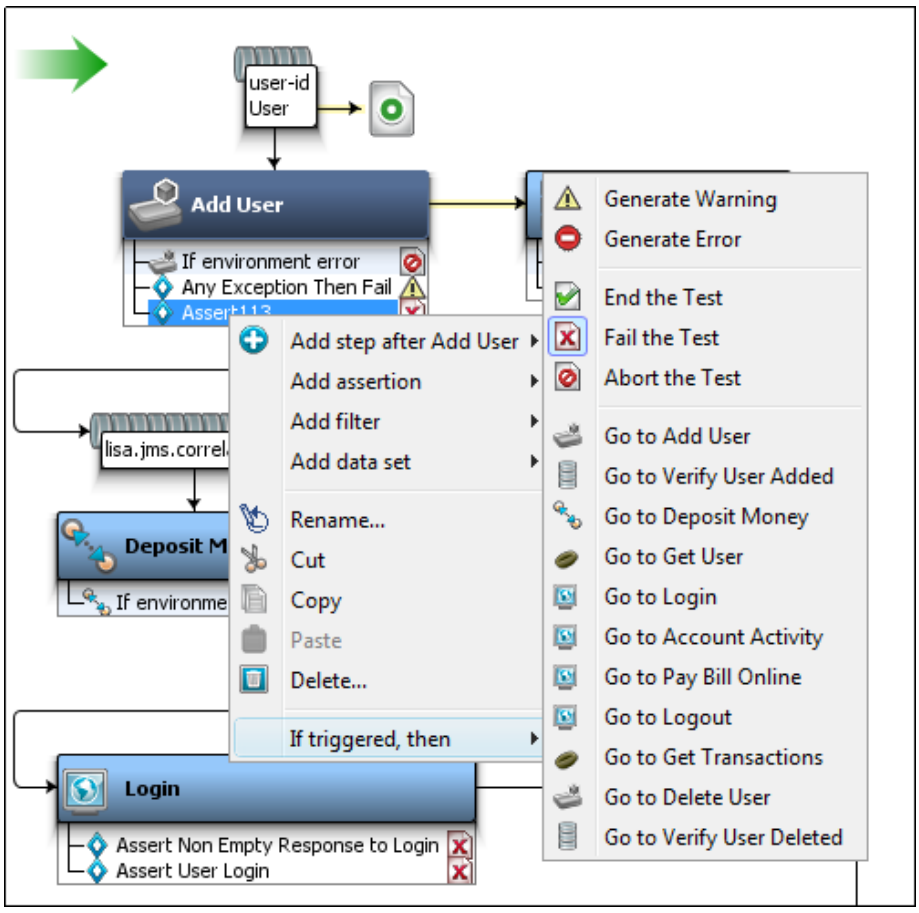
- Generate Warning (Générer un avertissement)
- Generate Error (Générer une erreur)

Par exemple, vous sélectionnez l'assertion dans l'étape Get User (Obtenir le nom d'utilisateur).

Procédez comme suit:

1. Pour ouvrir un menu, sélectionnez une étape et cliquez avec le bouton droit de la souris sur l'assertion.
2. Sélectionnez Generate Warning.

   Le scénario de test accède à cette étape suivante (Generate Warning), uniquement lorsque l'assertion est déclenchée.



Etape Generate Warning (Générer un avertissement)

Lorsque l'étape de test échoue, DevTest utilise le type Ignore de logique d'étape qui ne déclenche aucune alarme ou événement (Etape Generate Warning (Générer un avertissement)). L'étape Generate Warning (Générer un avertissement) ne change pas le flux de travaux de scénario de test.

Etape Generate Error (Générer une erreur)

Les étapes de test peuvent réussir ou échouer. Lorsqu'elles échouent, le test, lui, n'échoue pas.

Pour que le test échoue, l'étape de test doit définir le flux de travaux de scénario de test de sorte à exécuter l'étape Fail. Cette action indique implicitement que l'étape est considérée comme entraînant un échec.

Si une étape échoue de manière explicite, une erreur est générée et un événement NODEFAILED est déclenché, puis l'étape de test se poursuit.

Création de scénarios de test

Un scénario de test regroupe toutes les spécifications qui permettent de mener à bien un test sur un composant professionnel dans le système testé ou, dans certains cas, sur la totalité du système testé.

Un scénario de test est enregistré sous la forme d'un document XML, qui contient toutes les informations nécessaires au test du composant ou du système spécifié. Les scénarios de test sont créés et maintenus dans DevTest Workstation.

La première étape du processus de création de scénario de test consiste à créer un projet (page 50). Dans ce projet, vous pouvez créer un ou plusieurs scénarios de test.

Cette section comprend les rubriques suivantes :

Création d'un scénario de test

Vous pouvez créer un scénario de test en ouvrant un projet existant ou en créant un projet (page 50).

Par exemple, le projet par défaut examples affiche une arborescence contenant les dossiers Config (Configurations), Data (Données), Staging Doc (Document de simulation) Suites, Tests, etc.

Pour créer un scénario de test :

1. Cliquez avec le bouton droit de la souris sur le dossier Tests dans le panneau Project (Projet), puis cliquez sur Create New Test Case (Créer un scénario de test).
2. Dans la boîte de dialogue, accédez au répertoire dans lequel vous prévoyez de stocker le scénario de test et saisissez le nom du nouveau scénario de test.

   Pour plus d'informations, consultez la rubrique Scénarios de test dans l'éditeur de modèles (page 221).

Ouverture d'un scénario de test

Pour ouvrir ou afficher un scénario de test existant :

1. Dans le menu principal, sélectionnez File (Fichier), Open (Ouvrir), Test Case (Scénario de test).

   Les scénarios de test récemment ouverts sont affichés et répertoriés dans l'ordre d'utilisation. Le rapport le plus récent se trouve en haut de la liste.
2. Si le scénario de test cible est répertorié dans la liste, sélectionnez-le.

   Dans le cas contraire, accédez à ce scénario en cliquant sur File System (Système de fichiers), Classpath ou URL.
3. Sélectionnez le fichier à ouvrir, puis cliquez sur Open (Ouvrir).

   Le scénario de test sélectionné est ouvert et affiché dans l'éditeur de modèles. Vous pouvez également ouvrir un scénario de test existant en le double-cliquant à partir du panneau Project (Projet).

   Vous pouvez désormais ajouter de nouveaux éléments (filtres, assertions) ou modifier les éléments existants dans le scénario de test.

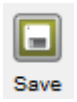
Enregistrement d'un scénario de test

Lorsque vous enregistrez un scénario de test, DevTest enregistre également les résultats de chacune de ses étapes dans un document de réponse disponible dans le même répertoire et portant le suffixe .rsp. Pour plus d'informations, consultez la rubrique Documents de réponse (page 225).

Si un élément, un filtre, une assertion ou un ensemble de données d'une étape de test a un champ obligatoire vide, vous ne pouvez pas enregistrer le scénario de test.

Pour enregistrer un scénario de test :

Effectuez l'une des opérations suivantes :

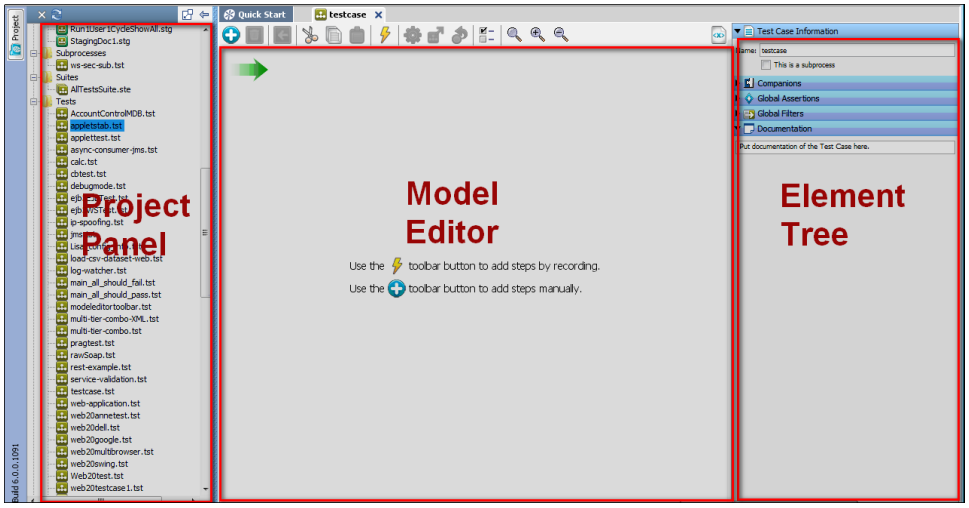
- Cliquez sur Save (Enregistrer) 

dans la barre d'outils.
- Dans le menu principal, sélectionnez File (Fichier), Save (Enregistrer). Le nom d'origine du scénario de test est affiché.

Scénarios de test dans l'éditeur de modèles

Lorsque vous créez ou ouvrez un scénario de test, l'éditeur de modèles s'ouvre.

L'éditeur de modèles fournit une vue graphique du scénario de test, avec toutes les étapes de test et les éléments associés. Pour consulter des exemples, ouvrez les scénarios de test du répertoire **LISA_HOME/examples.**



L'éditeur de modèles comprend trois sections :

- Panneau Project (Projet) **:** permet de créer un projet sous lequel vous créez, affichez ou modifiez des scénarios de test ou d'autres documents.
- Model Editor (Editeur de modèles) **:** permet d'ajouter, de modifier et de supprimer les étapes de test créées pour un scénario de test.
- Panneau Element (Elément) **:** permet d'appliquer des filtres, des assertions, des ensembles de données ou des compagnons à un scénario de test ou à une étape de test.

Ajout d'étapes de test

Vous pouvez ajouter des étapes de plusieurs façons.

Pour ajouter une étape de test :

Effectuez l'une des opérations suivantes :

- Sélectionnez Commands (Commandes), Create New Step (Créer une étape) à partir du menu principal.
- Cliquez sur l'icône Add Step (Ajouter une étape) dans la barre d'outils Test Case (Scénario de test).
- Cliquez avec le bouton droit de la souris sur une étape dans le flux de travaux, puis cliquez sur Add Step After (Ajouter une étape après). Une étape est ajoutée après ou au-dessous de l'étape.

Lorsqu'une étape est insérée dans un flux de travaux existant, les étapes attenantes à l'étape insérée sont mises à jour automatiquement, afin de maintenir la linéarité du flux de travaux.

Si le flux de travaux contient des branches ou des boucles, l'étape suivante n'est pas définie automatiquement.

Configuration de l'étape suivante

Pour configurer l'étape suivante dans l'éditeur de modèles :

1. Sélectionnez l'étape de test cible et cliquez avec le bouton droit de la souris pour ouvrir un menu.
2. Cliquez sur For Next Step (Pour l'étape suivante) et sélectionnez l'étape suivante cible.

Réorganisation des étapes de test d'un scénario de test

Pour réorganiser les étapes dans l'éditeur de modèles :

1. Sélectionnez une étape et cliquez avec le bouton droit de la souris pour la réorganiser dans un flux de travaux.
2. Sélectionnez l'étape à définir comme étape suivante.

Vous pouvez également effectuer un glisser-déposer dans les étapes, dans l'éditeur de modèles. Si le flux de travaux est linéaire, toutes les étapes sont mises à jour automatiquement. Un flux de travaux non linéaire contient des branches et des boucles. Lorsque le flux de travaux n'est pas linéaire, les étapes suivantes ne sont pas mises à jour. Dans les flux de travaux non linéaires, vous pouvez mettre l'étape suivante à jour dans l'onglet Step Information (Informations sur l'étape) de cette étape.

Création de branches et de boucles dans un scénario de test

Création de branches dans un scénario de test

La création d'une branche dans un scénario de test est effectuée à l'aide d'assertions.

Vous pouvez appliquer un certain nombre d'assertions à une étape de test. Toutes les assertions consistent en une condition qui renvoie la valeur true ou false. La première assertion pour laquelle la condition est remplie est déclenchée. Cela permet de modifier le flux de travaux. Dans plusieurs étapes, une condition d'erreur par défaut est automatiquement créée comme assertion d'échec. Lorsque vous créez des assertions, il est recommandé de faire preuve de cohérence et de toujours créer des branches positives ou négatives.

Les assertions sont décrites en détail dans la rubrique Ajout d'une assertion (page 143).

Création de boucles dans un scénario de test

Les boucles sont créées lorsqu'une étape de test suivant une étape de test particulière dans le flux envoie un contrôle à l'étape de test qui a commencé le flux, comme dans une boucle. Ce qui est important est la capacité de sortir des boucles. Une sortie conditionnelle d'une boucle (équivalente à une boucle while) est effectuée en définissant des assertions sur une étape de test appartenant à la boucle.

Une boucle qui s'exécute un certain nombre de fois ou jusqu'à épuisement de certaines données (équivalente à une boucle for ou foreach) est effectuée à l'aide d'ensembles de données. Un ensemble de données est utilisé pour affecter des valeurs à une ou plusieurs propriétés un nombre limité de fois. L'étape suivante exécutée une fois que l'ensemble de données est épuisé est spécifiée avec la définition de l'ensemble de données, ce qui vous permet de l'interrompre.

Par exemple, si un ensemble de données contient 20 lignes d'utilisateurs qui doivent se connecter à un système, vous pouvez créer une boucle permettant d'exécuter l'étape de connexion pour chaque ligne dans l'ensemble de données. De même, vous pouvez utiliser un ensemble de données de compteur numérique pour exécuter une étape spécifique un certain nombre de fois.

Une étape unique peut s'appeler elle-même et effectuer une boucle sur un ensemble de données. Plusieurs étapes peuvent s'exécuter dans une boucle et utiliser les données d'un ensemble de données. Cela se produit lorsque l'étape finale du groupe d'étapes pointe vers la première étape du groupe.

Les ensembles de données sont décrits en détail dans la rubrique Ensembles de données (page 162).

Importation de scénarios de test

Vous pouvez importer d'anciens scénarios de test dans le répertoire de projet actif.

Pour importer des scénarios de test :

1. Cliquez avec le bouton droit de la souris sur le dossier Tests dans le panneau Project (Projet) et sélectionnez Import Files (Importer des fichiers).
2. Sélectionnez les fichiers à importer dans ce dossier.
3. Cliquez sur OK.

   Le processus d'importation démarre et copie tous les fichiers, y compris les fichiers des sous-répertoires, dans le dossier sélectionné. Toutes les configurations des scénarios de test et des modèles de service virtuel sont fusionnées avec les configurations du projet.

Pour importer des versions de fichiers antérieures à LISA 5.0 :

1. Dans le menu principal, sélectionnez File (Fichier), New (Créer), Project (Projet).

   La boîte de dialogue Create New Project (Créer un projet) s'affiche.
2. Sélectionnez un répertoire contenant des scénarios de test de version antérieure.
3. Cochez la case Base the new project on one of the following (Baser le nouveau projet sur l'un des éléments suivants).
4. Sélectionnez l'option Create Project from existing documents directory (Créer un projet à partir du répertoire de documents existant).
5. Cliquez sur Create (Créer).

   Le projet est créé avec tous les fichiers de version antérieure convertis pour la nouvelle version. Si un projet portant le même nom existe, vous êtes invité à spécifier un autre nom pour le nouveau projet.

Les messages de création du projet incluent le message de conversion automatique pour tous les scénarios de test et les modèles de service virtuel transformés pour remplir les conditions de la nouvelle version.

Une fois le processus terminé, vous pouvez afficher les messages suivants :

- Migration du projet
- Détails des modifications de la configuration
- Détails des modifications du scénario de test

Tous les fichiers importés sont sélectionnés.

Documents de réponse (.rsp)

Lorsque vous procédez à un enregistrement à partir d'un site Web ou interagissez avec un serveur, les réponses sont enregistrées dans un document de réponse. Vous pouvez donc consulter les informations ultérieurement.

Les documents de réponse sont créés et maintenus automatiquement, sans votre intervention, et enregistrés dans des fichiers avec l'extension .rsp, portant le même nom que le fichier de scénario de test. Comme pour les fichiers de scénario de test, les documents de réponse sont des fichiers XML.

Un document de réponse contient la réponse HTTP pour chacune des étapes HTTP d'un scénario de test, les informations de réponse des appels de service Web et les résultats JDBC d'une requête de base de données.

Par exemple, si vous avez exécuté les étapes HTTP à l'aide de l'utilitaire Interactive test Run (Exécution d'un test interactif), le document de réponse enregistré contient la totalité de l'arborescence DOM pour le résultat de chaque étape HTTP du test. Vous pouvez utiliser ces informations de réponse pour valider des données et créer des filtres ou des assertions simples.

Pour plus d'informations sur l'utilisation des réponses HTTP pour la création de filtres, consultez la rubrique Filtres (page 124).

Pour plus d'informations sur l'utilisation des réponses HTTP pour la création d'assertions, consultez la rubrique Assertions (page 142).

Certaines étapes ont des résultats qui ne sont pas éligibles pour être stockés dans un document de réponse.

Si vous copiez un fichier de scénario de test vers un autre emplacement, copiez également le document de réponse associé afin de ne pas perdre les réponses enregistrées.

Les documents de réponse sont facultatifs, car ils ne sont pas nécessaires à l'exécution des tests. Ils servent principalement à afficher les résultats, l'arborescence DOM, la table JDBC, etc.

Lorsque vous utilisez la fonction de relecture, les informations sont lues à partir du document de réponse.

Accès aux fichiers d'un autre projet

Vous pouvez utiliser la propriété **LISA_PROJ_ROOT** pour accéder aux fichiers d'un autre projet qui se trouve au même niveau de répertoire que le projet actuel.

Par exemple, vous avez les projets suivants :

- C:\Lisa\Projects\Project1
- C:\Lisa\Projects\Project2

Lorsque vous travaillez sur le projet Project2, vous pouvez accéder à un fichier du projet Project1 en spécifiant le type de chemin suivant :

```
{{LISA_PROJ_ROOT}}/../Project1/Data/AccountNumbers.xls
```

Assurez-vous de fournir un chemin valide dans tous vos environnements.

Barre d'outils de scénario de test

Cette barre d'outils s'ouvre une fois que vous avez ouvert un scénario de test dans l'éditeur de modèles. Toutes les tâches indiquées sont spécifiques à un scénario de test.

Icône	Description
	Créer un étape
	Supprimer une étape de test
	Définir l'étape actuellement sélectionnée comme l'étape de démarrage dans le flux de travaux
	Couper le texte sélectionné
	Copier le texte sélectionné
	Coller le texte sélectionné

	Cliquez sur cette option pour ouvrir un menu et créer des étapes en enregistrant un scénario de test dans le navigateur DevTest. Vous pouvez enregistrer un scénario de test avec les éléments suivants : ■ Enregistreur Web (Proxy HTTP) ■ Enregistreur mobile
	Cliquez dessus pour ouvrir un menu et sélectionnez l'une des actions suivantes : ■ Importation d'un script JSON pour créer une étape de test Selenium ■ Exportation d'une étape de test Selenium comme script JSON
	Démarrer une nouvelle exécution d'un test interactif
	Simuler un test rapide
	Relire le test jusqu'à un point particulier
	Afficher les propriétés du modèle
	Réinitialiser le zoom à l'échelle 1:1
	Effectuer un zoom avant
	Effectuer un zoom arrière
	Afficher la source XML

Génération de sous-processus

Un sous-processus est un scénario de test appelé par un autre scénario de test, au lieu d'un test exécuté comme test autonome.

Vous pouvez utiliser les sous-processus comme des modules dans d'autres scénarios de test, ce qui facilite leur réutilisation. Vous pouvez créer une bibliothèque de sous-processus que vous pouvez partager dans plusieurs scénarios de test.

Un sous-processus équivaut à une fonction ou une sous-routine dans un langage de programmation.

Un scénario de test doit être autonome. La valeur de toutes les propriétés utilisées dans le scénario de test doit provenir de ce scénario de test. Un sous-processus attend l'étape de test qui l'exécute pour fournir certaines valeurs de propriétés (propriétés d'entrée). Lorsque le sous-processus a terminé, il met les valeurs de propriétés à la disposition de l'étape appelante (propriétés de retour).

Les sous-processus peuvent être intégrés et appeler un autre sous-processus.

Remarque : Si un sous-processus qui en appelle un autre est marqué comme Quiet (Silencieux), tous les sous-processus appelés seront silencieux.

Vous créez les étapes du sous-processus de la même façon que pour un scénario de test standard, à l'exception des différences suivantes :

- Marquez le scénario de test en tant que sous-processus dans l'onglet Test Case Information (Informations sur le scénario de test) du scénario de test. Pour plus d'informations, consultez la rubrique Création d'un scénario de test de sous-processus (page 229) dans la rubrique *Utilisation de CA Application Test*.
- N'ajoutez pas d'ensembles de données comme pour un scénario de test. L'ensemble de données doit faire partie du scénario appelant et les valeurs actuelles sont transférées au sous-processus lorsqu'il est appelé. Cependant, lorsqu'un ensemble de données fait partie de la logique même du sous-processus, Dans ce scénario, n'utilisez pas d'ensemble de données global.
- N'utilisez aucun fichier de configuration ou compagnon dans le sous-processus pour initialiser les paramètres qui doivent être transférés à partir de l'étape appelante. A des fins de test, ces valeurs sont ajoutées ailleurs. Lorsque le sous-processus est appelé, l'étape appelante transfère ces valeurs.

Remarque : Le paramètre de délai de réflexion du scénario de test parent (le scénario de test qui appelle le sous-processus) est propagé au sous-processus. Pour exécuter les délais de réflexion de vos sous-processus indépendamment du processus appelant, définissez une propriété testExec portant le nom **lisa.subprocess.setThinkScaleFromParent** sur false, de sorte à décider pour chaque sous-processus. Pour une application globale, définissez **lisa.subprocess.setThinkScaleFromParent=false** dans le fichier local.properties.

Vous pouvez créer des sous-processus de zéro ou convertir un scénario de test existant en un sous-processus.

L'étape de test Execute Subprocess (Exécuter un sous-processus) simplifie l'appel d'un scénario de test de sous-processus.

Cette section comprend les rubriques suivantes :

Création d'un scénario de test de sous-processus

Procédez comme suit:

1. Créez un scénario de test ou ouvrez un scénario existant.
2. Ouvrez l'onglet Test Case Info (Informations sur le scénario de test) du scénario de test.

   Pour ouvrir l'onglet Test Case Info, cliquez sur un espace vide dans l'éditeur de modèles (aucune étape ne doit être sélectionnée). L'onglet Test Case Info s'ouvre dans le panneau droit.
3. Pour définir ce scénario de test en tant que sous-processus, cochez la case This is a subprocess (Il s'agit d'un sous-processus).

   Par défaut, un scénario de test n'est pas conçu pour être un sous-processus.
4. Les onglets Subprocess Input Parameters (Paramètres d'entrée du sous-processus) et Subprocess Ouput Properties (Propriétés de sortie du sous-processus) sont ajoutés.
5. Dans l'onglet Documentation, documentez le sous-processus de manière détaillée.

   Ce texte est affiché dans l'étape de test qui appelle le sous-processus.
6. Lorsque vous avez fini d'ajouter les étapes du sous-processus, configurez les propriétés d'entrée et de sortie pour le sous-processus.

Définition de propriétés d'entrée de sous-processus

Procédez comme suit:

1. Cliquez sur l'onglet Subprocess Input Parameters (Paramètres d'entrée du sous-processus).
2. Définissez les paramètres d'entrée actuels et les paramètres d'entrée potentiels.

   Une liste des paramètres requis par le sous-processus s'affiche dans le champ Subprocess Input Parameters (Paramètres d'entrée du sous-processus). Certains paramètres sont automatiquement ajoutés. Vous pouvez en ajouter si nécessaire.
3. Pour ajouter une propriété, cliquez sur Add (Ajouter)  au bas du panneau.
4. Entrez les valeurs des champs Key (Clé, nom de la propriété), Description et Default Value (Valeur par défaut).

   La valeur par défaut est utilisée lorsque vous exécutez le sous-processus dans l'ITR (utilitaire d'exécution d'un test interactif). Cette valeur vous permet de tester le sous-processus comme s'il s'agissait d'un scénario de test standard. Ces valeurs par défaut sont ignorées lorsqu'une autre étape de test appelle le sous-processus.
5. Pour supprimer des propriétés inutiles, utilisez le bouton Delete  (Supprimer).

   Prospective Input Parameters (Paramètres d'entrée potentiels, qui peuvent être requis ultérieurement.)

   Si DevTest détecte une éventuelle propriété d'entrée dans le sous-processus, elle sera répertoriée dans ce champ. S'il s'agit d'une propriété d'entrée valide, cliquez sur Add (Ajouter) pour ajouter cette propriété à la liste Subprocess Input Parameters (Paramètres d'entrée du sous-processus).

Définition de paramètres de sortie de sous-processus

Subprocess Output Properties (Résultat, propriétés de sortie du sous-processus) **:** liste de toutes les propriétés définies par le sous-processus. Une liste des paramètres requis par le sous-processus s'affiche dans le champ Subprocess Output Properties. Certains paramètres sont ajoutés automatiquement, tandis que d'autres doivent l'être manuellement.

Pour ajouter une propriété, cliquez sur Add (Ajouter)  au bas du panneau.

Si vous souhaitez que certaines propriétés ne soient pas mises à la disposition de l'étape appelante, cliquez sur Delete (Supprimer)  pour les supprimer.

Une fois que les propriétés d'entrée et de retour ont été vérifiées, et que les valeurs par défaut ont été entrées pour tous les paramètres d'entrée, vous pouvez exécuter le sous-processus dans l'ITR.

Conversion d'un scénario de test existant en sous-processus

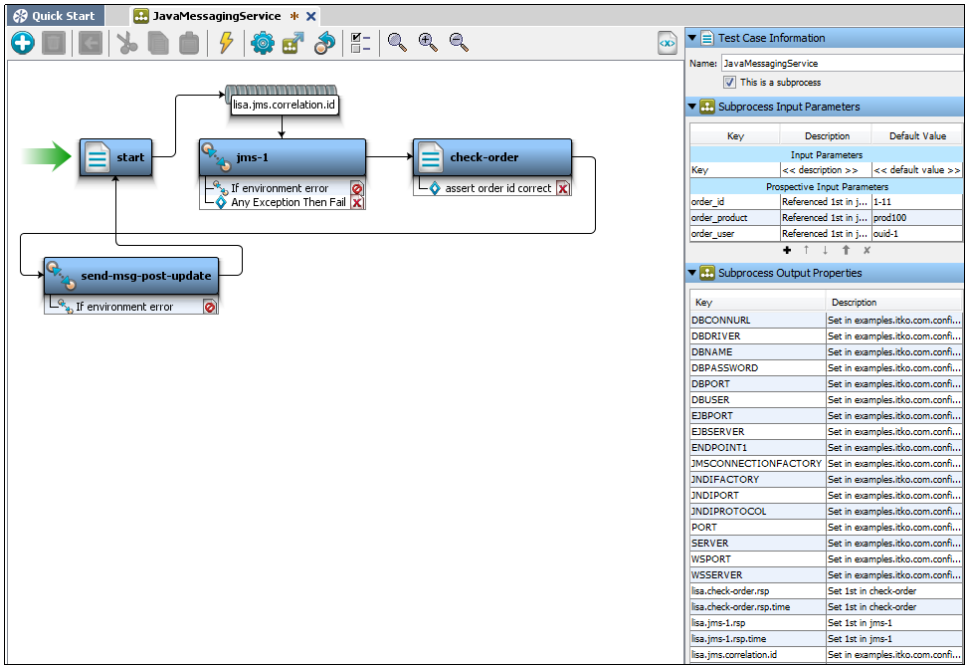
Procédez comme suit:

1. Ouvrez le scénario de test et renommez-le de façon appropriée.
2. Marquez le scénario de test en tant que sous-processus dans l'onglet Test Case Info (Informations sur le scénario de test) du scénario de test.
3. Supprimez tous les ensembles de données qui fournissent les valeurs des propriétés qui doivent être des propriétés d'entrée du nouveau sous-processus.

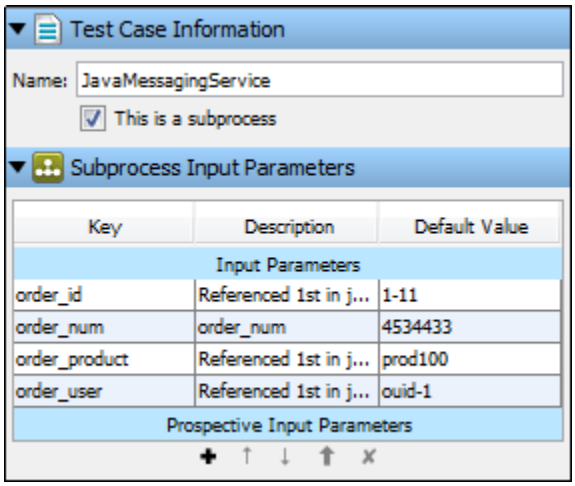
   Cependant, ne supprimez pas les ensembles de données lorsqu'ils font partie de la logique même du sous-processus.
4. Supprimez toutes les propriétés des fichiers de configuration ou un compagnon qui initialisent les paramètres qui doivent être des propriétés d'entrée du nouveau sous-processus.
5. Une fois que les propriétés d'entrée et de sortie ont été vérifiées, et que les valeurs par défaut ont été entrées pour tous les paramètres d'entrée, vous pouvez exécuter le sous-processus dans l'ITR.

Exemple de sous-processus

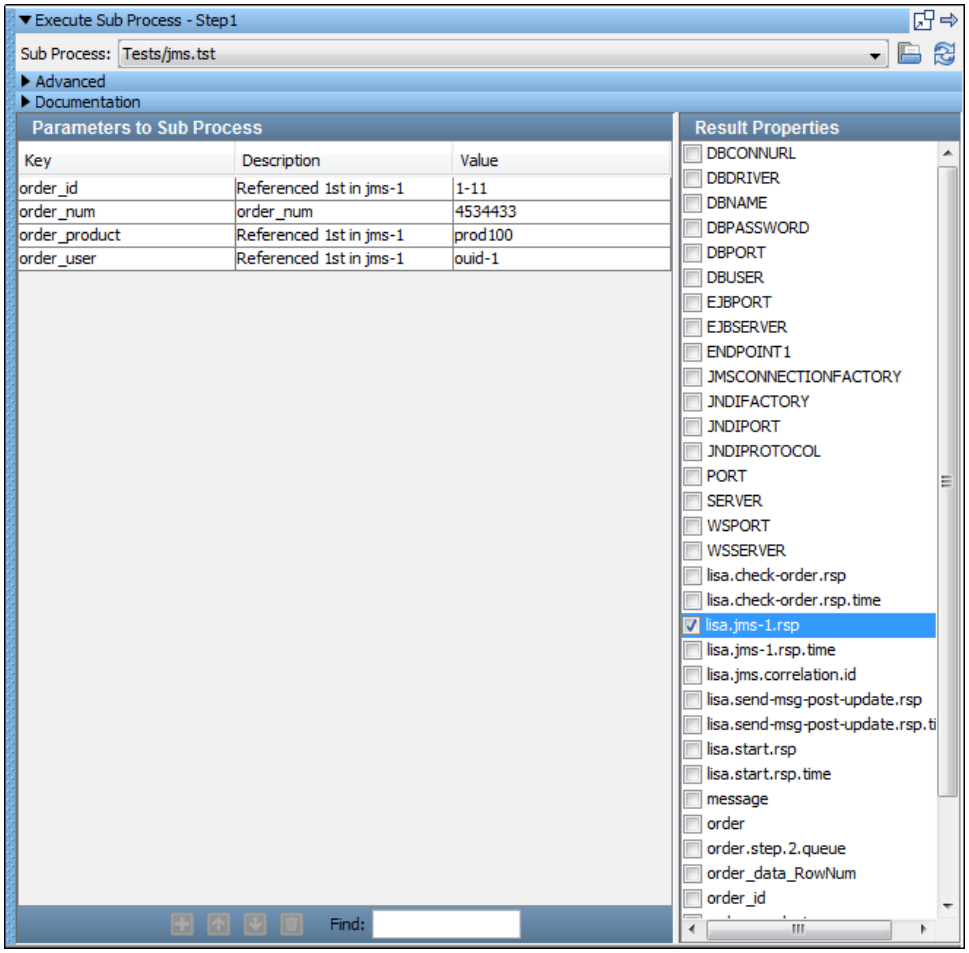
L'exemple suivant illustre un sous-processus dérivé du scénario de test jms.tst disponible dans le répertoire Examples de DevTest et un scénario de test qui appelle ce sous-processus.



Un ensemble de données, order_data, a été supprimé du scénario de test d'origine. L'ensemble de données order_data a fourni les valeurs pour la propriété **order_num** dans le scénario de test d'origine. La propriété **order_num** est désormais une propriété d'entrée.



Le graphique suivant représente un scénario de test comprenant une étape de test Execute Sub Process (Exécuter un sous-processus).



L'étape 1 est de type Execute Sub Process (Exécuter un sous-processus) et appelle le sous-processus **jms** (illustré précédemment).

La propriété d'entrée correspond et l'étape appelante a demandé que la propriété **lisa.jms-1.rsp** soit mise à disposition à la fin de l'exécution du sous-processus.

Chapitre 11: Génération de documents

Un document de simulation contient les informations sur l'exécution d'un scénario de test.

Un document d'audit vous permet de définir des critères de réussite pour un scénario de test dans une suite.

Vous pouvez appliquer des mesures et ajouter des événements, qui sont utilisés pour surveiller le scénario de test après son exécution.

Vous pouvez utiliser un document de suite pour exécuter une collection de scénarios de test.

Ce chapitre traite des sujets suivants :

Génération de documents de simulation

Un document de simulation contient les informations sur l'exécution d'un scénario de test.

La section contient les rubriques suivantes :

Création d'un document de simulation

Pour créer des documents de simulation, utilisez DevTest Workstation.

Si un scénario de test contient un ensemble de données global sur la première étape configurée pour terminer le test lorsque l'ensemble de données est épuisé, toutes les instances de test pour une exécution simulée prennent fin. D'autres paramètres de simulation, tels que la durée de l'état stable, seront remplacés.

Les ensembles de données locaux ne mettent pas fin à l'exécution simulée de cette façon, ni les ensembles de données dans les étapes autres que la première étape.

Procédez comme suit:

1. Dans le menu principal, sélectionnez File (Fichier), New (Créer), Staging Document (Document de simulation).

   La boîte de dialogue New Staging Doc (Nouveau document de simulation) s'affiche.
2. Entrez le nom du nouveau document de simulation.
3. Cliquez sur OK.

   L'éditeur de documents de simulation s'affiche.
4. Dans l'onglet Base (page 238), spécifiez les informations de base sur le document de simulation.

   Ces informations incluent le nom du document de simulation, le modèle de charge et le modèle de distribution.
5. Dans l'onglet Reports (page 248) (Rapports), spécifiez le type de rapport que vous voulez créer lors de l'exécution.
6. Dans l'onglet Metrics (page 250) (Mesures), spécifiez les mesures que vous voulez enregistrer lors de l'exécution.
7. Dans l'onglet Documentation (page 252), spécifiez une description pour le document de simulation.
8. (Facultatif) Dans l'onglet IP Spoofing (page 253) (Usurpation de l'adresse IP), activez et configurez l'usurpation de l'adresse IP.
9. (Facultatif) Dans l'onglet Source View (page 256) (Affichage de la source), vérifiez la version XML du document de simulation.
10. Dans le menu principal, sélectionnez File (Fichier), Save (Enregistrer).

Editeur Staging Document Editor (Editeur de documents de simulation)

L'éditeur Staging Document Editor vous permet de spécifier des critères pour l'exécution des scénarios de test.

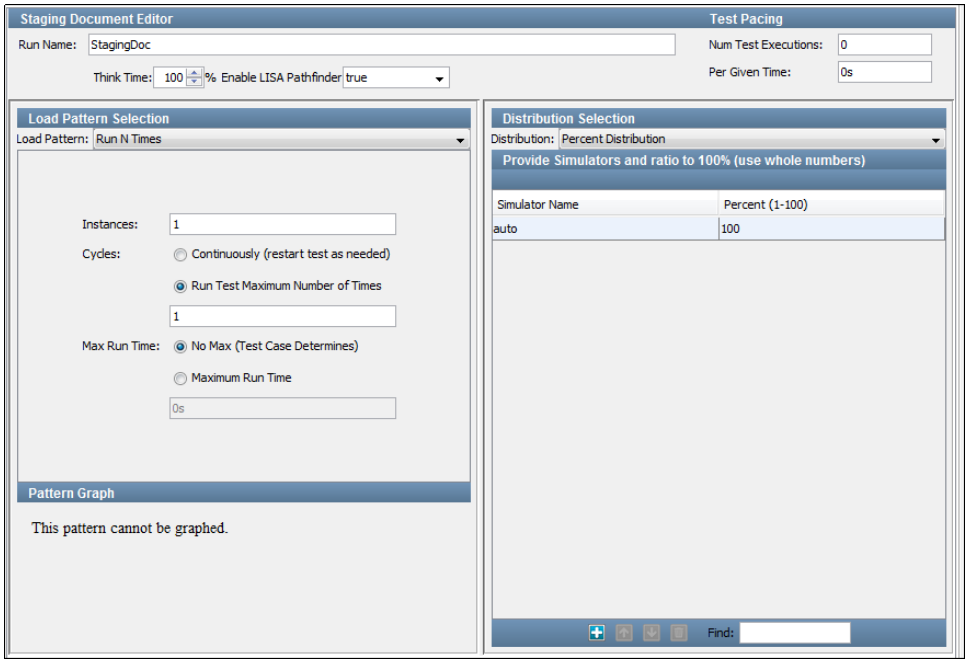
Il contient les onglets suivants :

- Base (page 238): permet de spécifier les paramètres de base.
- Reports (Rapports) (page 248) : permet de sélectionner et d'ajouter des rapports.
- Metrics (Mesures) (page 250): permet de sélectionner des mesures et de spécifier les intervalles d'échantillonnage.
- Documentation (page 252) : permet d'entrer une description pour le document de simulation.
- IP Spoofing (Usurpation de l'adresse IP) (page 253): permet d'entrer les détails d'usurpation de l'adresse IP. L'usurpation de l'adresse IP permet d'utiliser plusieurs adresses IP sur une interface réseau pour des demandes réseau.
- Source View (Affichage de la source) (page 256): permet d'afficher la source XML du document de simulation.

Onglet Base de l'éditeur Staging Document Editor (Editeur de documents de simulation)

L'onglet Base de l'éditeur Staging Document Editor décrit les paramètres de base d'un scénario de test :

- Les ajustements globaux des délais de réflexion
- La durée du test (temps écoulé ou nombre d'exécutions)
- Les informations liées au rythme d'exécution des tests (nombre de tests à terminer dans la période spécifiée)
- Le nombre d'utilisateurs virtuels
- Les modèles de charge pour les utilisateurs virtuels
- Les modèles de distribution pour les utilisateurs virtuels



L'onglet Base (De base) comprend les panneaux suivants :

- Panneau supérieur
- Panneau Load Pattern Selection (Sélection du modèle de charge)
- Panneau Distribution Selection (Sélection de la distribution)

Panneau supérieur

Le panneau supérieur vous permet de définir les paramètres suivants :

Run Name (Nom de l'exécution)

Nom du document de simulation.

Think Time (Délai de réflexion)

Délai de réflexion en pourcentage, pour toutes les étapes de test du scénario de test. Chaque étape de test peut définir un délai de réflexion dans la section Step Information (Informations sur l'étape). Cette section vous permet d'appliquer une modification globale aux délais de réflexion, sous la forme d'un pourcentage de leurs valeurs. Par exemple :

- Pour éliminer les délais de réflexion, définissez le pourcentage sur 0 %.
- Pour réduire de moitié les délais de réflexion, définissez le pourcentage sur 50 %.
- Pour doubler les délais de réflexion, définissez le pourcentage sur 200 %.

Activation de CA CAI

Vous pouvez activer ou désactiver CA Continuous Application Insight.

Num Test Executions (Exécutions de test)

Nombre d'exécutions de test à réaliser dans un délai spécifique

Per Given Time (Délai)

Période (temps horloge) pendant laquelle les tests sont exécutés

Valeurs : h représentent les heures, **m** les minutes et **s** les secondes.

Vous pouvez spécifier que 2 500 tests doivent être exécutés en 8 minutes.

Les délais de réflexion ne sont pas modifiés pour suivre le rythme requis.

DevTest est exécuté sans effectuer de pause entre les tests lorsque le rythme des tests ne peut pas être suivi, car il est trop élevé. DevTest indique dans le journal que le rythme de test est inférieur à celui demandé.

DevTest n'ajoute pas d'autres utilisateurs si le rythme des tests ne peut pas être suivi, car un nombre insuffisant d'utilisateurs virtuels est spécifié. Pour évaluer le nombre d'utilisateurs virtuels requis, reportez-vous à l'utilitaire Optimizer (page 332) (Optimiseur).

Panneau Load Pattern Selection (Sélection du modèle de charge)

Le panneau Load Pattern Selection (Sélection du modèle de charge) permet de réaliser les actions suivantes :

- Définir la durée des tests
- Définir le nombre d'utilisateurs virtuels (instances)
- Définir le modèle de charge pour les utilisateurs virtuels (si vous en avez plusieurs)

Pour plus d'informations, consultez la rubrique Sélection du modèle de charge (page 241).

Panneau Distribution Selection (Sélection de la distribution)

Le panneau Distribution Selection vous permet de distribuer des utilisateurs virtuels (instances) sur les simulateurs en cours d'exécution. Pour plus d'informations, consultez la rubrique Sélection de la distribution (page 245).

Panneau Load Pattern Selection (Sélection du modèle de charge)

L'onglet Base de l'éditeur Staging Document Editor (Editeur de documents de simulation) inclut un panneau Load Pattern Selection (Sélection du modèle de charge).

Les options de modèle de charge sont les suivantes :

- Immediately Ramp (Rampe)
- Manual Load Pattern (Modèle de charge manuelle)
- Run N Times (Exécutions multiples)
- Stair Steps (Marches d'escalier)
- Weighted Average Pattern (Modèle de moyenne pondérée)

Remarque : Le rythme est pris en charge uniquement avec les modèles de charge suivants :

- Run N Times (Exécutions multiples)
- Immediate Ramp (Rampe immédiate)
- Stair Step (Marche d'escalier)

Définir un rythme dépend de la possibilité pour le modèle de charge d'estimer le nombre d'instances en état stable. Le nombre d'instances en état stable en cours d'exécution affecte le rythme d'exécution des étapes. Ces modèles de charge sont les seuls à pouvoir évaluer ce rythme.

Immediately Ramp (Rampe)

Le modèle de charge Immediately Ramp (Rampe) s'applique lorsque vous exécutez uniquement un nombre limité d'utilisateurs virtuels (instances) et que vous voulez spécifier la durée d'exécution du test. Vous n'êtes pas concerné par aucun modèle de charge. Ce modèle démarre tous les utilisateurs virtuels simultanément.

Pour configurer ce modèle, entrez les paramètres suivants :

Instances

Nombre d'utilisateurs virtuels

Max Run Time (Temps maximum d'exécution)

Choisissez entre No Max (Test Case Determines) (Aucune valeur maximum (déterminé par le scénario de test)) et Maximum Run Time (Temps maximum d'exécution). Si vous sélectionnez Maximum Run Time, vous devez spécifier une valeur. Ce paramètre écrase toutes les valeurs entrées pour les cycles. Vous pouvez spécifier **h** pour les heures, **m** pour les minutes, **s** (ou aucune lettre) pour les secondes (valeur par défaut).

Le graphique dans la partie inférieure fournit une vue graphique du modèle.

Manual Load Pattern (Modèle de charge manuelle)

Le modèle de charge Manual Load Pattern vous offre un meilleur contrôle sur le chargement et le déchargement des utilisateurs virtuels (instances). Ce modèle est similaire au modèle Stair Steps (Marches d'escalier), mais il vous permet de spécifier le nombre d'utilisateurs virtuels à ajouter et l'intervalle pour chaque étape du modèle.

Pour configurer ce modèle, vous définissez chaque étape en tant que ligne dans une table. Dans chaque ligne, vous définissez l'intervalle et le nombre d'utilisateurs virtuels à ajouter ou supprimer (colonne Instances Change (Modification d'instances)) pour cette étape. Le temps écoulé (colonne Total Time (Durée totale)) et le nombre total d'utilisateurs virtuels (colonne Running Instances (Instances en cours d'exécution)) sont automatiquement calculés.

Dans la colonne Time Interval (Intervalle) entrez un intervalle suivi par un **h** pour les heures, un **m** pour les minutes, un **s** (ou aucune lettre) pour les secondes (valeur par défaut).

Pour ajouter, supprimer ou modifier l'ordre actuel des étapes, utilisez les icônes standard de la barre d'outils au bas de la table.

Pour afficher ces icônes, vous devrez peut être faire défiler la page. De même, vous pouvez sélectionner une ligne et utiliser les raccourcis suivants :

- Ajouter une ligne : Ctrl+Maj+A
- Supprimer une ligne : Ctrl+Maj+D
- Déplacer la ligne vers le haut : Ctrl+Maj+flèche vers le haut
- Déplacer la ligne vers le bas : Ctrl+Maj+flèche vers le bas
- Vue étendue : Ctrl+Maj+L

Le graphique dans la partie inférieure fournit une vue graphique du modèle.

Run N Times (Exécutions multiples)

Le modèle Run N Times (Exécutions multiples) s'applique lorsque vous exécutez un seul utilisateur virtuel ou un nombre limité d'utilisateurs virtuels (instances) et que vous voulez spécifier le nombre d'exécutions du test. Vous n'êtes pas concerné par aucun modèle de charge. Ce modèle démarre tous les utilisateurs virtuels simultanément.

Pour configurer ce modèle, entrez les paramètres suivants :

Instances

Nombre d'utilisateurs virtuels

Cycles

Choisissez entre une exécution continue jusqu'à ce que le temps spécifié dans le paramètre Maximum Run Time (Temps maximum d'exécution) ait été atteint ou une exécution selon un nombre maximum de fois.

Max Run Time (Temps maximum d'exécution)

Choisissez entre No Max (Test Case Determines) (Aucune valeur maximum (déterminé par le scénario de test)) et Maximum Run Time (Temps maximum d'exécution). Si vous sélectionnez Maximum Run Time, vous devez spécifier une valeur. Ce paramètre écrase toutes les valeurs entrées pour les cycles. Vous pouvez spécifier **h** pour les heures, **m** pour les minutes, **s** (ou aucune lettre) pour les secondes (valeur par défaut).

Stair Steps (Marches d'escalier)

Le modèle Stair Steps (Marches d'escalier) introduit des utilisateurs virtuels (instances) dans le système à des étapes définies, non de manière simultanée. Spécifiez le nombre total d'utilisateurs d'état stable, la durée de la rampe ascendante et de la rampe descendante, ainsi que le nombre d'étapes pour la rampe.

Pour configurer ce modèle, entrez les paramètres suivants :

Steady State Instances (Instances d'état stable)

Nombre maximum d'utilisateurs virtuels à exécuter dans un état stable.

Number of Steps (Nombre d'étapes)

Nombre d'étapes à utiliser pour atteindre le nombre maximum d'utilisateurs virtuels. Le nombre d'utilisateurs virtuels à introduire dans chaque étape correspond au résultat de la valeur de Steady State instances (Instances d'état stable) divisée par la valeur de Number of Steps (Nombre d'étapes).

Ramp Up Time (Durée de la rampe ascendante)

Période pendant laquelle les utilisateurs virtuels sont ajoutés pour atteindre le nombre maximum d'utilisateurs virtuels. L'intervalle entre les étapes correspond à la valeur de Ramp Up Time divisée par la valeur de Number of Steps (Nombre d'étapes).

Steady State Time (Durée de l'état stable)

Période de l'exécution du test explicite.

Ramp Down Time (Durée de la rampe descendante)

Période pendant laquelle les utilisateurs virtuels sont supprimés. La durée spécifiée dans le champ Ramp Down Time est approximative, car l'exécution des tests se poursuivra après une demande d'arrêt.

Vous pouvez spécifier une heure suivie de **h** pour les heures, **m** pour les minutes, **s** (ou aucune lettre) pour les secondes (valeur par défaut).

Le graphique dans la partie inférieure fournit une vue graphique du modèle.

Weighted Average Pattern (Modèle de moyenne pondérée)

Le modèle Weighted Average Pattern permet d'ajouter et de supprimer des utilisateurs virtuels (instances) selon un calcul statistique. DevTest calcule le nombre d'étapes et la période entre celles-ci, toutes les secondes, en appliquant une moyenne pondérée mobile. Cette distribution se rapprochera d'une distribution de courbe en cloche. La plupart des utilisateurs virtuels sont ajoutés dans deux écarts types du point intermédiaire du temps de rampe de charge ou de décharge.

Pour configurer ce modèle, entrez les paramètres suivants :

Steady State Instances (Instances d'état stable)

Nombre maximum d'utilisateurs virtuels à exécuter dans un état stable.

Ramp Up Time (Durée de la rampe ascendante)

Période pendant laquelle les utilisateurs virtuels sont ajoutés pour atteindre le nombre maximum d'utilisateurs virtuels.

Steady State Time (Durée de l'état stable)

Période de l'exécution du test explicite.

Ramp Down Time (Durée de la rampe descendante)

Période pendant laquelle les utilisateurs virtuels sont supprimés. La durée spécifiée dans le champ Ramp Down Time est approximative, car l'exécution des tests se poursuivra après une demande d'arrêt.

Vous pouvez spécifier une heure suivie de **h** pour les heures, **m** pour les minutes, **s** (ou aucune lettre) pour les secondes (valeur par défaut).

Le graphique dans la partie inférieure fournit une vue graphique du modèle.

Panneau Distribution Selection (Sélection de la distribution)

L'onglet Base de l'éditeur Staging Document Editor (Editeur de documents de simulation) inclut un panneau Distribution Selection (Sélection de la distribution).

Si vous utilisez la station de travail DevTest Workstation, le panneau Distribution Selection (Sélection de distribution) n'est pas requis, car vos utilisateurs virtuels sont exécutés au niveau local (à l'aide d'un simulateur intégré à la station de travail DevTest Workstation).

Si vous utilisez le DevTest Server avec plusieurs serveurs de simulation actifs, ce panneau Distribution Selection (Sélection de distribution) permet de spécifier le mode de distribution des utilisateurs virtuels sur ces simulateurs.

Les options de distribution sont les suivantes :

- Balanced Based on Instance Capacity (Equilibrage d'après la capacité d'instances)
- Dynamic Simulator Scaling with DCM (Mise à l'échelle du simulateur dynamique avec DCM)
- Percent Distribution (Distribution en pourcentage)
- Round Robin Distribution (Distribution en tourniquet)

Balanced Based on Instance Capacity (Equilibrage d'après la capacité d'instances)

Avec la distribution Balanced Based on Instance Capacity (Equilibrage d'après la capacité d'instances), DevTest contrôle l'allocation des utilisateurs virtuels (instances), selon une évaluation de la charge actuelle (pourcentage de charge sur chaque simulateur).

Chaque simulateur est initialisé avec un nombre défini d'utilisateurs virtuels que vous pouvez allouer. La valeur par défaut est 255. DevTest permet de suivre de façon dynamique le pourcentage de charge et d'ajouter des utilisateurs virtuels à des simulateurs, afin de maintenir un équilibre dans le pourcentage de charge. Par conséquent, le simulateur avec le pourcentage de charge le moins élevé est un candidat pour le prochain utilisateur virtuel introduit dans le système.

Cette distribution est utile lorsque le système exécute déjà des tests à partir de plusieurs autres testeurs et vous voulez optimiser la distribution de la charge.

Aucun paramètre n'est requis.

Dynamic Simulator Scaling with DCM (Mise à l'échelle du simulateur dynamique avec DCM)

Avec la distribution Dynamic Simulator Scaling with DCM (Mise à l'échelle du simulateur dynamique avec DCM), DevTest détermine automatiquement la nécessité d'une capacité supplémentaire pour répondre aux besoins d'un test en cours d'exécution et étend automatiquement le laboratoire.

Ce modèle de distribution inclut les paramètres suivants :

Checkpoint time (Intervalle d'évaluation du point de contrôle)

Intervalle d'évaluation par DevTest de la nécessité de simulateurs supplémentaires. Vous pouvez saisir la valeur en secondes (par exemple, 300 s) ou en minutes (par exemple, 5 m).

DCM Dynamic Lab Name (Nom du laboratoire dynamique DCM)

Si vous simulez un coordinateur existant et que le test détermine qu'une capacité supplémentaire est requise pour l'exécution du test, ce paramètre indique le laboratoire à activer pour exécuter d'autres simulateurs. Incluez le préfixe VLM et le nom de laboratoire complet.

Maximum Expansion (Extension maximum)

Nombre maximum de simulateurs à créer au moment de l'extension. Si une stratégie limite le nombre de simulateurs par utilisateur, utilisez ce paramètre pour l'appliquer. La valeur par défaut 0 indique un nombre illimité.

Pendant l'évaluation de point de contrôle, DevTest examine les performances des simulateurs en cours d'exécution et détermine le nombre d'utilisateurs virtuels à ajouter. Si d'autres simulateurs sont requis, DevTest démarre le processus d'extension du laboratoire. Lorsque les simulateurs sont en ligne, DevTest dirige une partie du trafic vers ceux-ci.

Percent Distribution (Distribution en pourcentage)

Le paramètre Percent Distribution (Distribution en pourcentage) permet de distribuer des utilisateurs virtuels (instances) sur les simulateurs, en fonction des pourcentages spécifiés.

Les noms de simulateurs en cours d'exécution (en plus des simulateurs locaux) sont disponibles dans une liste déroulante dans la colonne Simulator Name (Nom de simulateur).

Sélectionnez un simulateur et spécifiez un pourcentage d'utilisateurs virtuels pour celui-ci. Répétez cette action pour tous les simulateurs que vous voulez inclure jusqu'à atteindre 100 %. Utilisez des nombres entiers pour les pourcentages.

Remarque : L'option Auto qui s'affiche dans la liste déroulante n'est pas recommandée. Elle entraîne des allocations d'utilisateurs virtuels qui interfèrent avec vos choix de distribution explicites.

Round Robin Distribution (Distribution en tourniquet)

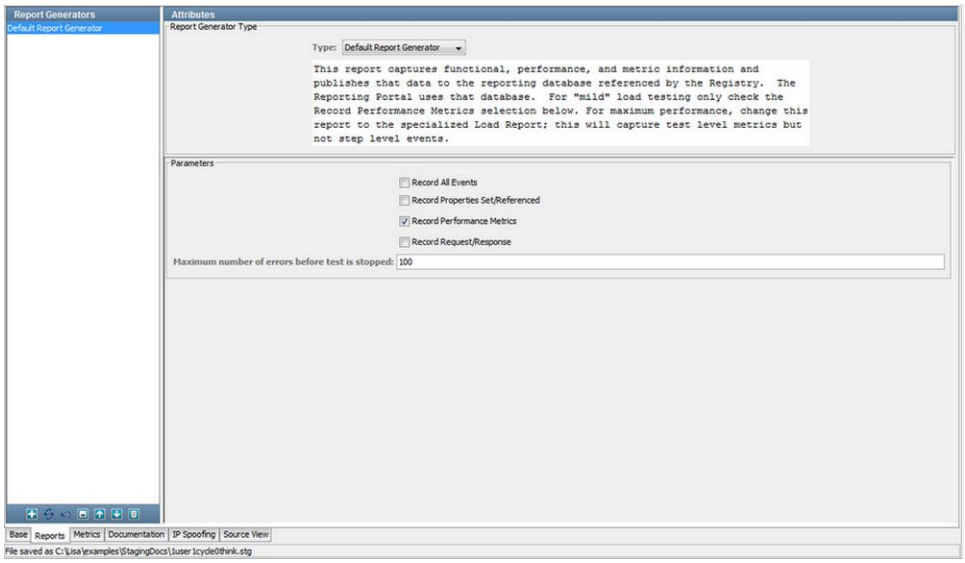
Le paramètre Round Robin Distribution (Distribution en tourniquet) indique à DevTest de contrôler l'allocation des utilisateurs virtuels (instances), selon un modèle simple de distribution en tourniquet.

DevTest sélectionne un simulateur de manière aléatoire et ajoute un utilisateur virtuel, puis sélectionne un autre simulateur et ajoute un utilisateur, ainsi de suite autant de fois que nécessaire. Une fois que tous les simulateurs ont été utilisés, le processus reprend avec le premier simulateur. Cette distribution est utile lorsque vous simulez un test de charge unique volumineux sur votre système.

Aucun paramètre n'est requis.

Onglet Reports (Rapports) de l'éditeur Staging Document Editor (Editeur de documents de simulation)

L'onglet Reports de l'éditeur Staging Document Editor vous permet de spécifier le générateur de rapports à appeler pour tous les scénarios de test ou les suites de tests.



Les types de générateurs de rapports suivants sont disponibles :

Default Report Generator (Générateur de rapports par défaut)

Ce générateur de rapports capture des informations fonctionnelles, de performances et de mesures, et publie ces données dans la base de données de rapports référencée par le registre. Le portail Reporting Portal (Portail de rapports) utilise cette base de données.

Load Test Report Generator (Générateur de rapports de test de charge)

Ce générateur de rapports est conçu pour les tests de charge comprenant des milliers d'utilisateurs virtuels. Ce rapport capture des mesures de charge, mais ignore les mesures de niveau d'étape. Si les mesures de niveau d'étape étaient capturées, il y aurait trop de données et la base de données de rapports ralentirait le test.

XML Report Generator (Générateur de rapports XML)

Ce générateur de rapports crée un fichier XML contenant toutes les données que vous pouvez capturer. Vous pouvez limiter les données capturées à l'aide des options de rapport. Pour afficher ce rapport, importez le fichier dans le portail Reporting Portal (Portail de rapports).

Les détails des données disponibles dans chaque générateur de rapports, l'affichage des rapports, le contenu des rapports et les options de sortie sont traités dans la rubrique Rapports (page 403).

Les mesures à capturer pour les rapports sont traitées en détails dans la rubrique Génération de mesures (page 265).

Vous pouvez sélectionner des rapports dans les modules suivants et les afficher dans la visionneuse Report Viewer (Visionneuse de rapports) :

- Simulation de test rapide (page 306)
- Génération de documents de simulation (page 235)
- Exécution d'une suite de tests (page 325)

Une fois que vous les avez demandés, vous pouvez les afficher et les gérer ultérieurement.

Lorsque vous sélectionnez un rapport dans la liste déroulante, un récapitulatif du rapport s'affiche dans la zone au-dessous.

Les paramètres varient selon le rapport sélectionné. Définissez les paramètres si nécessaire.

L'onglet Reports est divisé selon les sections suivantes :

Panneau droit

Le panneau droit vous permet de sélectionner le rapport à ajouter.

Attributes (Attributs)

La zone Attributes contient le menu déroulant Report Generator Type (Type de générateur de rapports) et les paramètres requis pour le rapport.

- Report Generator Type **:** menu déroulant qui répertorie les types de rapport disponibles.
- Parameters (Paramètres) **:** paramètres requis pour définir le générateur de rapports sélectionné.

  Le champ Maximum number of errors before test is stopped (Nombre maximum d'erreurs au-delà duquel le test est arrêté) définit une limite pour le nombre d'erreurs admises avant que le test soit interrompu. La valeur zéro indique que toutes les erreurs sont autorisées.

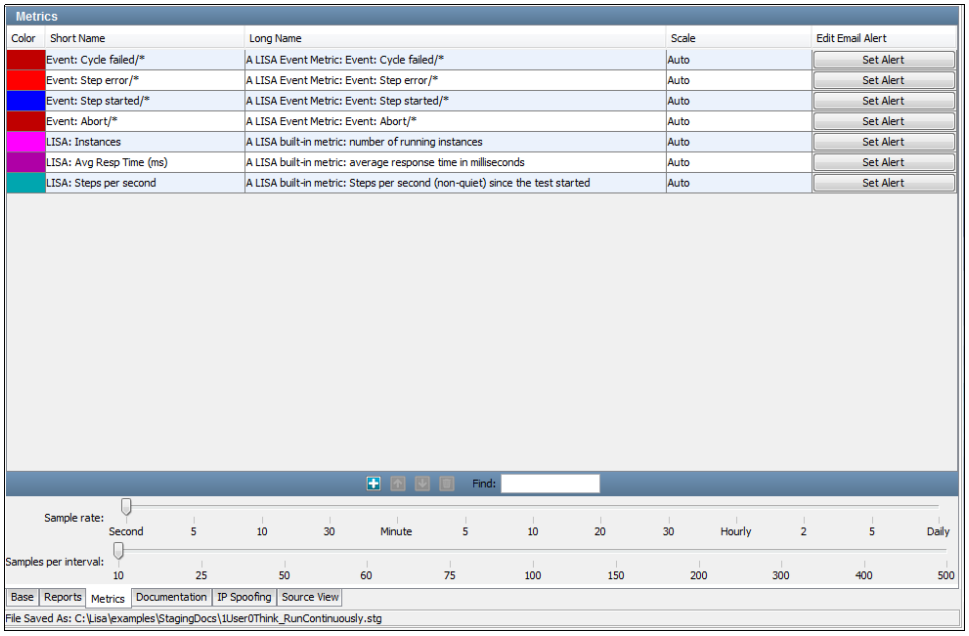
Panneau gauche

Le panneau gauche affiche la liste des rapports ajoutés. Vous pouvez ajouter, enregistrer, déplacer et supprimer des rapports à l'aide de la barre d'outils au bas du panneau.

Onglet Metrics (Mesures) de l'éditeur Staging Document Editor (Editeur de documents de simulation)

L'onglet Metrics (Mesures) de l'éditeur Staging Document Editor vous permet de sélectionner les mesures de test à enregistrer.

Vous pouvez également définir les spécifications de l'échantillonnage pour la collecte de mesures et définir une alerte par courriel pour une mesure sélectionnée.



L'onglet Metrics est divisé selon les sections suivantes : Le panneau supérieur répertorie les mesures par défaut, différenciées par un code couleur. Le panneau inférieur contient les paramètres de l'échantillonnage.

Vous pouvez changer la couleur utilisée pour représenter une mesure dans le document de simulation. Cliquez sur la couleur et sélectionnez une nouvelle couleur, puis enregistrez le document de simulation. Changez la couleur avant de simuler le test.

Vous pouvez ajouter ou supprimer des mesures et des alertes par courriel définies dans le panneau supérieur.

Les types de mesures suivantes sont disponibles :

- Mesures DevTest Whole Test Metrics (Mesures DevTest de test complet)
- Mesures DevTest Test Event Metrics (Mesures d'événement de test DevTest)
- Mesures SNMP Metrics (Mesures SNMP)
- JMX Attribute Reader (Lecteur d'attributs JMX, mesures JMX)
- TIBCO Hawk

- Windows Perfmon Metrics (Mesures Windows PerfMon)
- UNIX Metrics through SSH (Mesures UNIX via SSH)

Pour plus d'informations sur chaque mesure, consultez la section Mesures.

Ajout d'une mesure

Procédez comme suit:

1. Dans la barre d'outils, cliquez sur l'icône Add (Ajouter).

   Une boîte de dialogue vous permettant d'ajouter des mesures s'ouvre avec une liste de mesures que vous pouvez ajouter.

2. Sélectionnez le type de mesure cible et cliquez sur OK.

   La boîte de dialogue Metric Selection (Sélection de mesure) s'affiche.

3. Sélectionnez la sous-catégorie cible pour la mesure et cliquez sur OK.

   Les sous-catégories sont différentes pour chaque mesure. La mesure nouvellement ajoutée s'affiche dans la liste de mesures existantes dans une couleur différente.

Pour obtenir une description de toutes les mesures de toutes les catégories et des informations sur leur configuration pour les inclure dans des rapports, consultez la rubrique Génération de mesures (page 265).

Définition d'un message d'alerte

Pour définir une alerte par courriel pour une mesure :

1. Cliquez sur le bouton Set Alert (Définir une alerte) correspondant à la mesure.

   La boîte de dialogue Edit Alert (Modifier l'alerte) s'ouvre.

2. Vous pouvez définir les limites acceptables pour la mesure (valeur basse et élevée) et les détails à envoyer dans un courriel. Fournissez le nom du serveur de messagerie et une liste d'adresses électroniques.

   Les alertes par courriel qui sont ajoutées à des documents de simulation stockent le mot de passe SMTP en tant que valeur chiffrée. En outre, si vous ouvrez un document de simulation existant et le réenregistrez, le mot de passe SMTP est enregistré en tant que valeur chiffrée.

   Pour ajouter et supprimer des adresses électroniques, utilisez les boutons Add (Ajouter) et Delete (Supprimer) au bas de la fenêtre.

3. Lorsque vous avez terminé, cliquez sur Close (Fermer).

   Remarquez que le bouton Set Alert s'est converti en bouton Edit Alert dans l'onglet Metrics.

Remarque : Pour que l'alerte par courriel fonctionne, vous devez également définir les chemins relatifs au serveur SMTP dans le fichier lisa.properties.

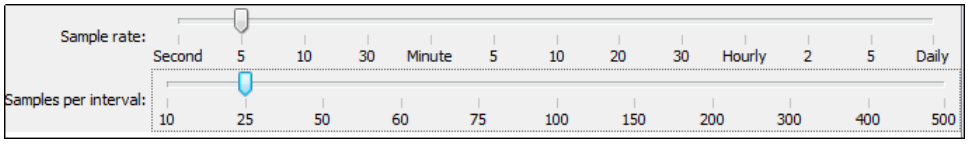
Paramètres d'échantillonnage

Le panneau inférieur comprend deux barres de curseur qui vous permettent de définir des paramètres d'échantillonnage :

- Sample rate (Taux d'échantillonnage) **:** ce paramètre spécifie la fréquence à laquelle effectuer un échantillonnage ou enregistrer la valeur d'une mesure. Ce paramètre est la valeur réciproque du taux d'échantillonnage spécifié sous la forme d'une période.
- Samples per interval (Echantillons par intervalle) **:** ce paramètre spécifie le nombre d'échantillons utilisés pour créer un intervalle pour calculer des valeurs récapitulatives pour la mesure.

Effectuer un échantillonnage toutes les minutes (Sample rate=1 minute) et la moyenne tous les 60 échantillons (Samples per interval=60) renvoie une mesure toutes les minutes et une valeur récapitulative (moyenne) toutes les heures.

Par exemple, la valeur par défaut est un taux d'échantillonnage de 1 seconde et 10 échantillons par intervalle (intervalle de 10 secondes).



Dans l'exemple précédent, 5 secondes par échantillon et 25 échantillons par intervalle étaient utilisés. Ces valeurs renvoient une mesure toutes les 5 secondes et une valeur récapitulative toutes les 125 secondes.

Les mesures sont stockées dans des fichiers XML ou des tables de base de données à inclure dans les rapports (voir la rubrique Rapports (page 248)).

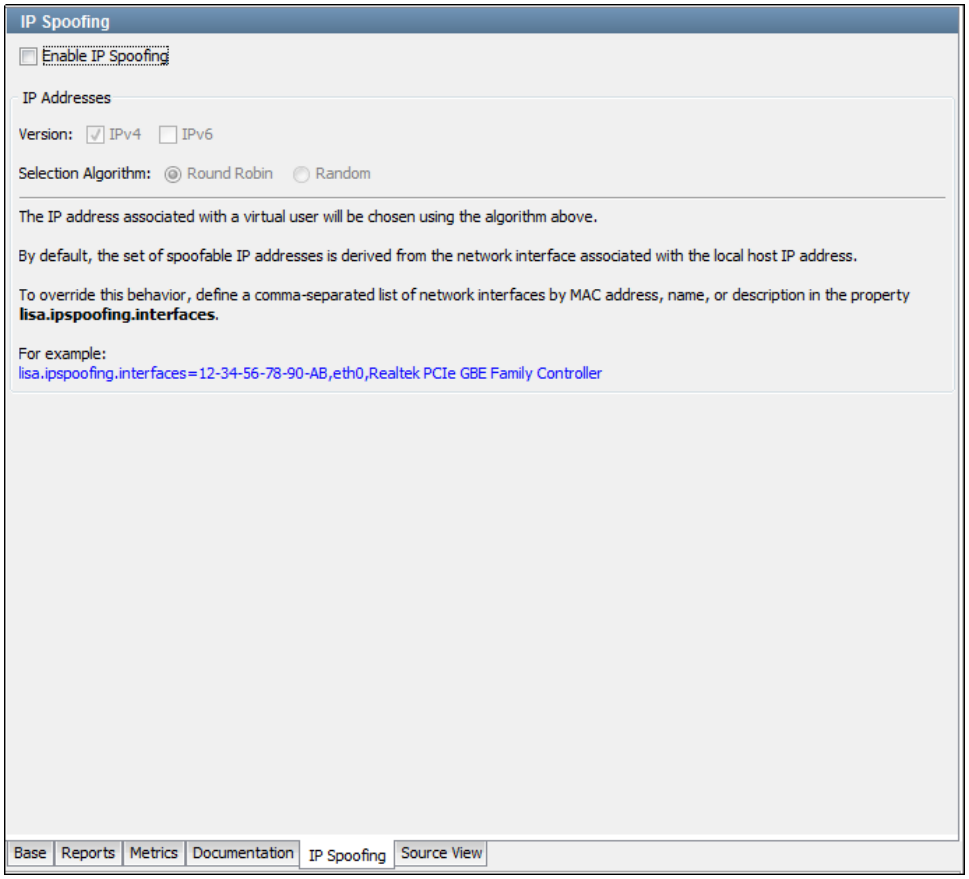
Onglet Documentation de l'éditeur Staging Document Editor (Editeur de documents de simulation)

L'onglet Documentation de l'éditeur Staging Document Editor vous permet d'entrer une description du document de simulation.

Onglet IP Spoofing (Usurpation de l'adresse IP) de l'éditeur Staging Document Editor (Editeur de documents de simulation)

L'onglet IP Spoofing de l'éditeur Staging Document Editor permet d'utiliser plusieurs adresses IP sur une interface réseau pour des demandes réseau.

Dans un scénario de test des performances, l'activation de l'usurpation de l'adresse IP pour un système testé permet de prétendre que les demandes sont générées à partir de plusieurs utilisateurs virtuels différents. Pour certains systèmes comme les applications Web, ce comportement est très proche de leur comportement réel habituel.



Enable IP Spoofing (Activer l'usurpation d'adresse IP)

Si cette option est sélectionnée, les adresses IP sont usurpées pour toutes les étapes de test prises en charge dans un scénario de test.

Version

- IPv4 **:** si cette version est sélectionnée, les adresses IPv4 sont usurpées.
- IPv6 **:** si cette version est sélectionnée, les adresses IPv6 sont usurpées.

Vous devez sélectionner la version IPv4 ou IPv6.

Selection Algorithm (Algorithme de sélection)

- Round Robin (Tourniquet) **:** les adresses IP usurpées sont sélectionnées selon un format de tourniquet.
- Random (Aléatoire) **:** les adresses IP usurpées sont sélectionnées selon un format aléatoire.

Prise en charge de l'usurpation d'adresse IP

L'usurpation de l'adresse IP est prise en charge uniquement pour les transactions HTTP.

Vous pouvez configurer les étapes suivantes pour utiliser l'usurpation de l'adresse IP :

- HTTP/HTML Request (Demande HTTP/HTML)
- REST Step (Etape REST)
- Web Service Execution (XML) (Exécution de service Web (XML))
- Raw SOAP Request (Demande SOAP brute)

Scénarios d'usurpation d'adresse IP

Lorsque l'usurpation de l'adresse IP est configurée dans un document de simulation, les résultats attendus sont les suivants :

Un scénario de test unique avec un utilisateur unique et une boucle :

En fonction de l'ordre séquentiel ou aléatoire de la liste d'adresses IP, chaque boucle dans un cycle du test obtient une adresse IP différente.

Un scénario de test unique avec un utilisateur unique sans boucle, de sorte que le test démarre et se termine dans chaque cycle :

Utilisez l'adressage aléatoire pour que l'adresse IP soit différente. Si vous ne sélectionnez pas l'option **Random** (Aléatoire), chaque utilisateur commence toujours par la première adresse IP.

Un scénario de test unique avec plusieurs utilisateurs et une boucle se comporte comme dans le premier cas :

L'exécution ou le cycle de test maintient la liste des adresses IP à utiliser.

L'option random **(aléatoire) doit être sélectionnée pour un scénario de test unique comptant plusieurs utilisateurs sans boucle :**

Sélectionner l'option **random** (aléatoire) permet d'assurer qu'une adresse IP aléatoire sera utilisée lors du démarrage du cycle et non la première adresse de la liste.

Configuration des adresses IP dans Windows

Cette section décrit la méthode d'ajout d'adresses IP à une interface réseau pour l'usurpation de l'adresse IP.

Remarque : Avant d'ajouter des adresses IP, connectez-vous en tant qu'administrateur.

Sur Windows, vous pouvez obtenir les informations relatives à l'adresse IP à l'aide de l'utilitaire de ligne de commande **ipconfig**.

Pour ajouter une adresse IP unique, 192.168.0.201, utilisez l'utilitaire de ligne de commande **netsh**.

```
netsh in ip add address "Local Area Connection" 192.168.0.201 255.255.255.0
```

Pour ajouter plusieurs adresses IP, utilisez **netsh** dans une boucle.

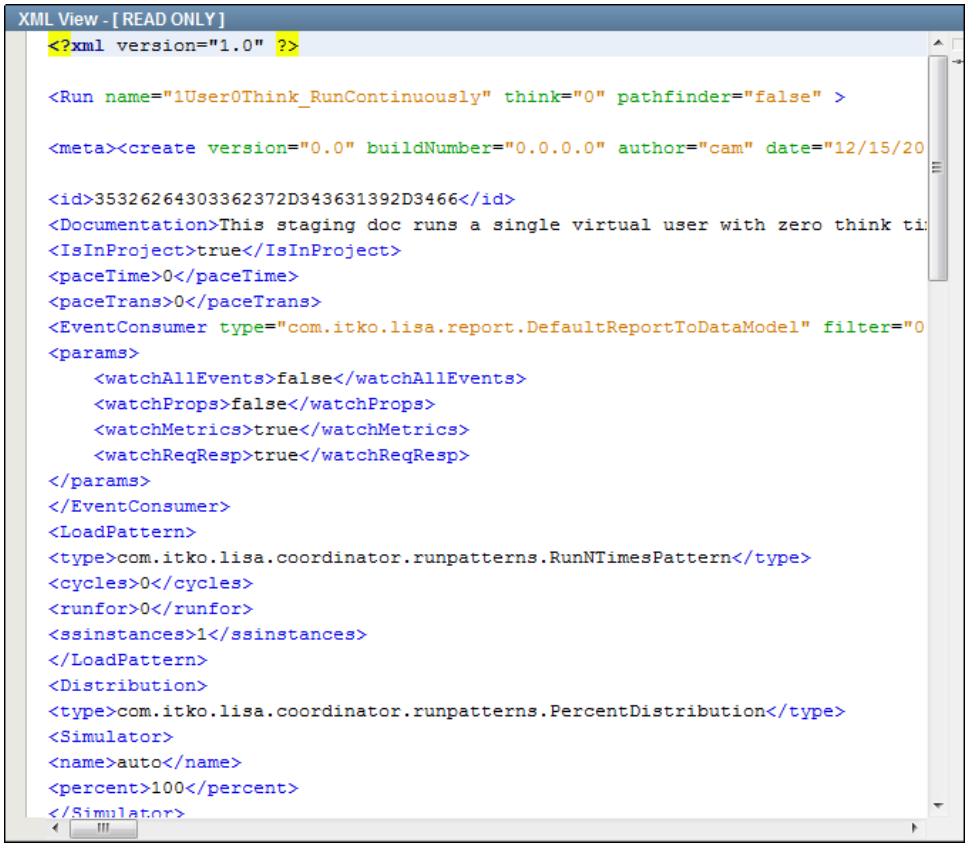
Par exemple, la commande suivante ajoute 9 adresses IP de plus comprises entre 192.168.0.202 et 192.168.0.210 :

```
for /L %i in (202, 1, 210) do netsh in ip add address "Local Area Connection"
192.168.0.%i 255.255.255.0
```

Si ces commandes sont exécutées, vous pouvez vérifier les nouvelles adresses IP à l'aide de l'utilitaire de ligne de commande **ipconfig / all**.

Onglet Source View (Affichage de la source) de l'éditeur Staging Document Editor (Editeur de documents de simulation)

L'onglet Source View de l'éditeur Staging Document Editor affiche la source XML du document de simulation.



Exemples de document de simulation

Des exemples de documents de simulation sont fournis dans le dossier StagingDocs du projet Examples (page 54) (Exemples). Tous utilisent le modèle de charge Run N Times (Exécutions multiples) et le modèle Percent Distribution (Distribution en pourcentage).

Vous pouvez utiliser ces documents comme référence pour créer un document de simulation. Pour cela, renommez-le pour l'enregistrer sous un nom différent.

- **1User0Think_RunContinuously.stg :** ce document de simulation exécute un utilisateur virtuel unique avec un délai de réflexion de zéro. Il exécute également le test en continu, ce qui ne signifie pas nécessairement pour toujours.
- **1user1cycle0think.stg :** ce document de simulation exécute un utilisateur virtuel unique une fois avec un délai de réflexion de zéro.
- **ip-spoofing.stg :** ce document de simulation vous permet de tester la prise en charge de l'usurpation de l'adresse IP.
- **jboss-jmx-metrics-localhost.stg :** ce document de simulation exécute trois utilisateurs virtuels parallèles une fois pendant 440 secondes.
- **Run1User1Cycle.stg :** ce document de simulation exécute un utilisateur virtuel unique une fois avec un délai de réflexion de 100 pour cent.
- **Run1User1CyclePF.stg :** ce document de simulation exécute un utilisateur virtuel unique une fois avec un délai de réflexion de 100 pour cent. CAI est activé.
- **Run1User1CycleShowAll.stg :** ce document de simulation exécute un utilisateur virtuel unique une fois avec un délai de réflexion de 100 pour cent. CAI est activé.

Génération de documents d'audit

Un document d'audit vous permet de définir des critères de réussite pour un scénario de test dans une suite.

Les documents d'audit vous permettent de suivre :

- Les événements qui doivent ou ne doivent pas se produire pendant un test.
- La durée du test (excessive ou trop courte)

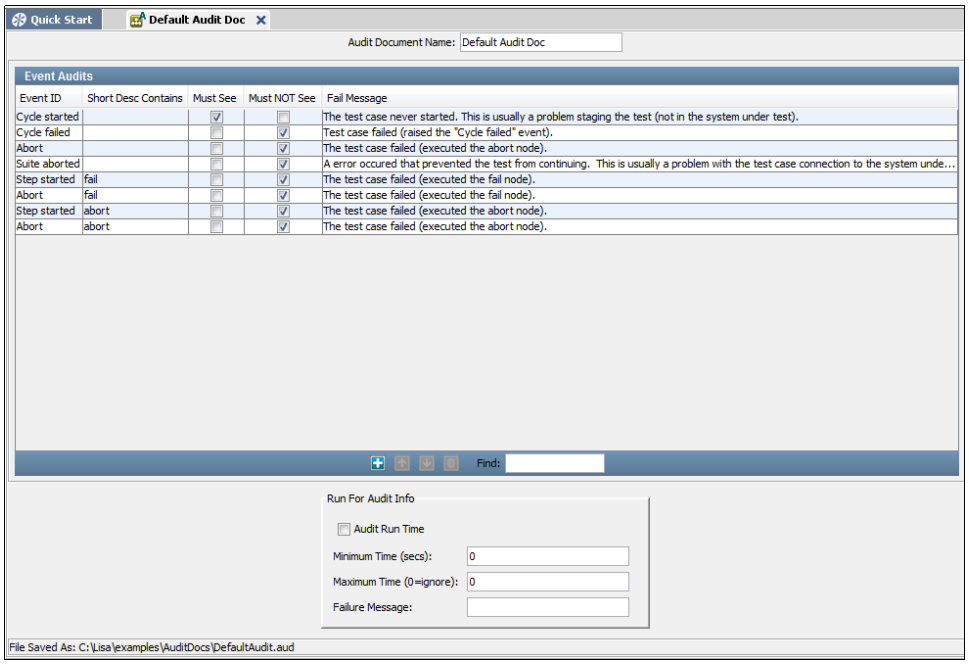
Vous pouvez spécifier des documents d'audit dans l'onglet Base (page 267) de l'éditeur de documents de suite. Assurez-vous de cocher la case Record All Events (Enregistrer tous les événements) dans l'onglet Reports (page 272) (Rapports).

Les documents d'audit se trouvent dans le dossier AuditDocs d'un projet. L'extension de fichier est **.aud**.

Lorsque vous créez un projet, le dossier AuditDocs contient un document d'audit par défaut nommé **DefaultAudit.aud**.

Remarque : Lorsque vous utilisez des documents d'audit, les noms d'étape de test ne doivent contenir aucune description courte du document d'audit utilisé dans le test. Par exemple, si le terme **Abort** (Interrompre) est une description courte du document d'audit, vous ne pouvez pas utiliser **Abort** (Interrompre) dans un nom d'étape de test.

Le graphique suivant présente l'éditeur de documents d'audit. L'éditeur contient un panneau Event Audits (Audits d'événement) et un panneau Run for Audit Info (Informations concernant l'exécution de l'audit).



Pour créer un document d'audit :

1. Dans le menu principal, sélectionnez File (Fichier), New (Créer), Test Audit (Document d'audit).
2. Entrez le nom du document d'audit, puis cliquez sur OK.

   L'éditeur de documents d'audit s'ouvre.
3. Pour auditer des événements, configurez les paramètres dans le panneau Event Audits (page 259) (Audits d'événement).
4. Pour auditer la durée d'exécution, configurez les paramètres du panneau Run for Audit Info (page 260) (Informations concernant l'exécution de l'audit).
5. Dans le menu principal, sélectionnez File (Fichier), Save (Enregistrer).

Panneau Event Audits (Audits d'événement)

Le panneau Event Audits vous permet de spécifier les événements qui doivent se produire ou ceux qui ne doivent pas se produire au cours d'un test.

Pour ajouter un événement :

1. Cliquez sur Add (Ajouter)  .
2. Dans la nouvelle ligne, sélectionnez le nom de l'événement dans la liste déroulante, dans la colonne Event ID (ID d'événement).

   Chaque ligne a les paramètres suivants :

   Short Desc Contains (Contenu de la description courte)

   Si vous voulez filtrer un événement en fonction de la valeur de sa description courte, entrez le ou les mots clés de la description courte dans cette colonne.

   Must See (Affichage obligatoire)

   Cochez cette case si l'événement doit se produire pendant le test.

   Must NOT See (Masquage obligatoire)

   Cochez cette case si l'événement ne doit pas se produire pendant le test.

   Fail Message (Message d'échec)

   Message à enregistrer en cas d'échec de l'audit.
3. Ajoutez des lignes supplémentaires pour chaque événement à inclure.

   Si vous essayez d'ajouter des audits d'événement qui entrent logiquement en conflit l'un avec l'autre, l'éditeur affiche un message d'avertissement.

   Vous pouvez réorganiser les lignes à l'aide des icônes Up (Haut)  et Down (Bas)  .

   Vous pouvez supprimer des lignes à l'aide de l'icône Delete (Supprimer)  .

Panneau Run for Audit Info (Informations concernant l'exécution de l'audit)

Le panneau Run for Audit Info vous permet d'auditer le temps d'exécution.

Ce panneau comprend les paramètres suivants :

Audit Run Time (Audit du temps d'exécution)

Pour activer l'audit du temps d'exécution, cochez cette case.

Minimum Time (Temps minimum)

Durée d'exécution minimum en secondes du test.

Maximum Time (Temps maximum)

Durée d'exécution maximum en secondes du test pendant laquelle l'audit est encore considéré comme réussi. S'il n'y a aucune limite de durée maximum, entrez la valeur 0.

Failure Message (Message d'échec)

Message à enregistrer en cas d'échec de l'audit.

Introduction aux événements

Un événement consiste en l'émission d'un message à partir de DevTest informant les parties intéressées qu'une action s'est produite.

Cette section comprend les rubriques suivantes :

- Présentation des événements (page 261)
- Ajout et affichage des événements (page 263)

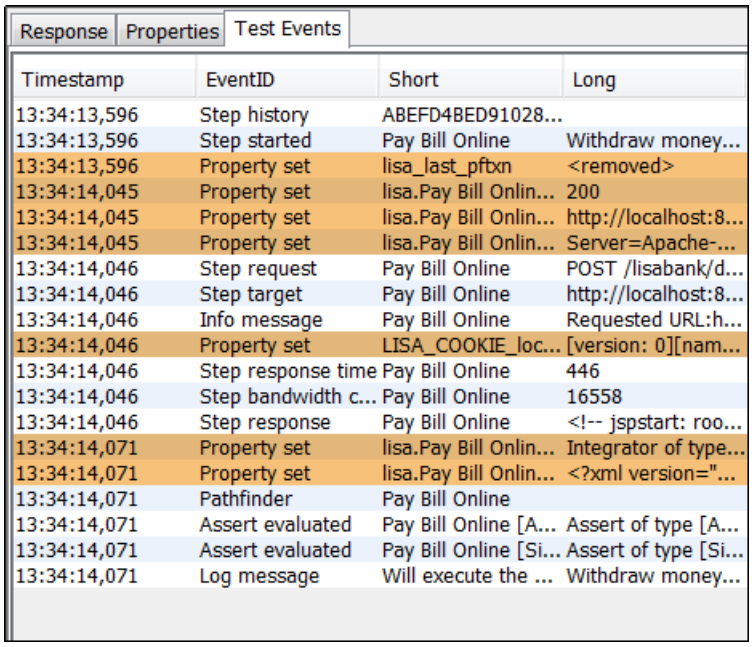
Présentation des événements

Un événement est l'émission d'un message informant les partis concernés qu'une action s'est produite. Des événements sont créés pour toutes les actions principales qui se produisent dans un scénario de test.

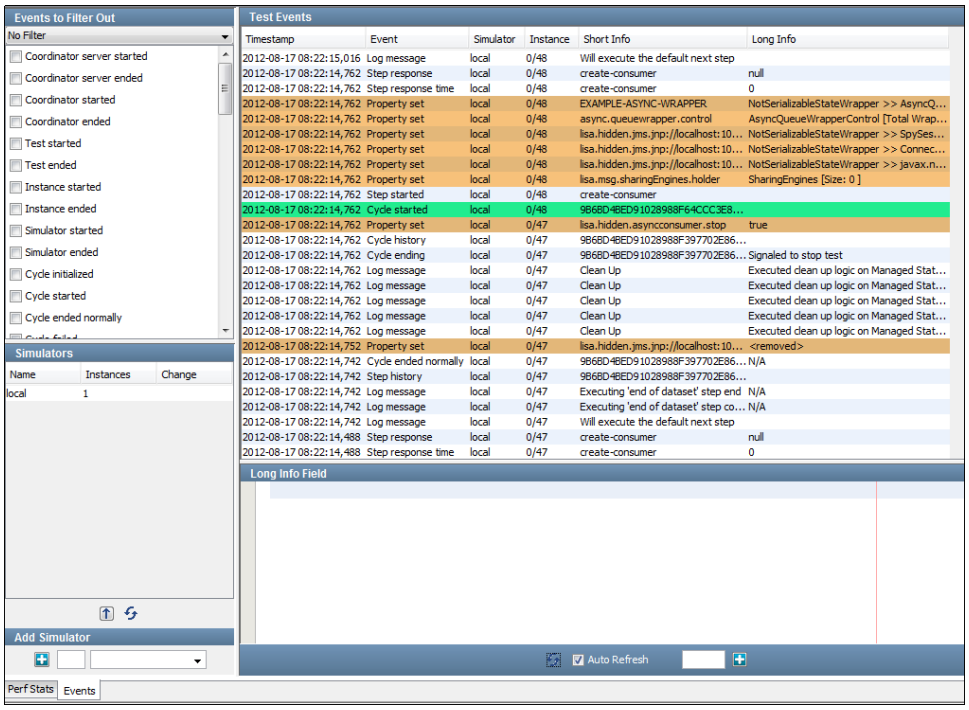
Un événement est créé à chaque exécution d'une étape, définition d'une propriété, renvoi d'un temps de réponse ou d'un résultat, échec ou réussite d'un test, etc. Vous pouvez afficher tous les événements ou les filtrer de sorte à afficher uniquement ceux qui vous intéressent.

Les événements sont importants lorsque vous surveillez des tests ou analysez des résultats de test.

Vous pouvez consulter les événements pendant l'exécution d'un test interactif, en cliquant sur l'onglet Test Events (Evénements de test) dans l'ITR. Le graphique suivant présente l'onglet Test Events (Evénements de test).



Vous pouvez observer et filtrer les événements dans un test rapide.



Vous pouvez également effectuer ces opérations au cours de la surveillance d'un test simulé ou d'une suite de tests.

Vous pouvez sélectionner les événements à inclure dans les rapports et sélectionner les événements à utiliser comme mesures à surveiller et inclure dans les rapports.

Pour plus d'informations sur les rapports, consultez la rubrique Rapports (page 403). Pour plus d'informations sur les mesures, consultez la rubrique Génération de mesures (page 265).

Remarque : Une étape de test interne à DevTest peut également être appelée node (noeud), ce qui explique l'intégration du terme node dans l'ID de certains événements.

Un événement se caractérise par un ID d'événement, une valeur courte et une valeur longue. Les ID d'événement sont des mots clés qui indiquent le type de l'événement. Les valeurs courtes et les valeurs longues contiennent des informations sur l'événement. Leur contenu varie selon le type d'événement.

Ajout et affichage des événements

Vous pouvez ajouter et afficher des événements :

- Via l'utilitaire d'exécution d'un test interactif
- Via un test rapide ou un document de simulation

Ajout et affichage des événements via l'ITR

Vous pouvez activer certains événements dans l'onglet Settings (Paramètres) de l'utilitaire d'exécution d'un test interactif.

Cochez les cases Show CALL_RESULT (Afficher les valeurs CALL_RESULT) et Show NODERESPONSE (Afficher les valeurs NODERESPONSE) pour afficher ces événements.

Vous pouvez afficher des événements de test dans l'ITR en cliquant sur l'onglet Test Events (Evénements de test).

Ajout et affichage des événements via un test rapide ou un document de simulation

Lorsque vous démarrez un scénario de test via l'option Start Test (Lancer un test) ou Start Suite (Lancer une suite), le test est simulé et les graphiques pertinents s'affichent dans l'onglet Perf Stats (Statistiques de performances).

Pour afficher les événements de test, cliquez sur l'onglet Events (Evénements).

Vous pouvez sélectionner l'option Events to Filter Out (Evénements à filtrer) dans le panneau gauche ou une option parmi les ensembles de filtres prédéfinis.

Pour actualiser la liste des événements de test, cochez la case Auto Refresh (Actualisation automatique) dans l'onglet Events (Evénements). Au cours de l'exécution du test, les événements sélectionnés s'affichent dans l'onglet Events (Evénements).

Affichage des événements dans les suites de tests

Vous pouvez également afficher les événements lors de la simulation d'une suite de tests.

Vous pouvez sélectionner les événements à inclure dans les rapports et sélectionner les événements à utiliser comme mesures à surveiller et inclure dans les rapports.

Pour plus d'informations sur les rapports, consultez la rubrique Rapports (page 403). Pour plus d'informations sur les mesures, consultez la rubrique Génération de mesures (page 265).

Un événement comprend un ID d'événement, une valeur courte et une valeur longue. Les ID d'événement sont des mots clés qui indiquent le type de l'événement. Les valeurs courtes et les valeurs longues contiennent des informations sur l'événement. Leur contenu varie selon le type d'événement.

Remarque : Une étape de test interne à DevTest peut également être appelée node (noeud), ce qui explique l'intégration du terme node dans l'ID de certains événements.

Génération de mesures

La collecte de mesures, une méthode de calcul de mesures propre au produit, est un mécanisme de génération de rapports extensible, qui vous permet de générer des rapports vers différentes sorties.

Les mesures vous permettent d'appliquer des méthodes et des mesures quantitatives aux aspects fonctionnels et de performances de vos tests, et au système testé.

La mesure de logiciel est une mesure de certaines propriétés d'une partie d'un logiciel, d'un système matériel ou de leurs caractéristiques. Les méthodes quantitatives utilisant des mesures se sont révélées très puissantes dans certains domaines informatiques, et notamment pour les phases de test.

Les deux groupes généraux de mesures sont les suivants :

- **Gauge (Jauge) :** une jauge fournit une lecture instantanée d'une valeur, telle que le temps de réponse ou l'utilisation de l'UC.
- **Counter (Compteur) :** un compteur permet de compter une propriété en continu. Par exemple , le nombre d'échecs de tests. Si une mesure est un compteur, il s'agit essentiellement d'un nombre toujours croissant pendant l'exécution du test (nombre d'erreurs). Lorsqu'une mesure est marquée comme un compteur, elle est représentée différemment dans un graphique dans la console de génération de rapports. La valeur représentée graphiquement est la différence entre l'échantillon actuel et l'échantillon précédent.

Pour la plupart des mesures, le type de mesure (jauge ou compteur) est déjà connu. Lorsqu'une mesure utilise les deux types, vous pouvez spécifier le type que vous préférez.

Vous pouvez ajouter des mesures aux documents de simulation et aux documents de suite de tests suivants. Les moniteurs de tests et les tests rapides peuvent générer des mesures. Des informations supplémentaires sur la spécification et la génération de mesures pour chaque document ou test sont disponibles dans les sections traitant de chaque éditeur.

- Tests rapides : (page 306) utilisés pour surveiller le test
- Documents de simulation : (page 250) utilisés dans les rapports
- Documents de suite de tests : (page 274) utilisés dans les rapports
- Moniteurs de tests : (page 307) utilisés pour surveiller des tests

Génération de suites de tests

Vous pouvez exécuter ou simuler une combinaison de scénarios de test dans DevTest. Pour exécuter une collection de scénarios de test ensembles, vous utilisez un formulaire de suite de tests.

Cette section comprend les rubriques suivantes :

Création d'une suite de tests

Pour créer des suites de tests, utilisez DevTest Workstation.

Procédez comme suit:

1. Dans le menu principal, sélectionnez File (Fichier), New (Créer), Suite.

   L'éditeur de suites s'ouvre.

2. Dans l'onglet Base (page 267), spécifiez les informations de base sur la suite.

   Ces informations incluent le nom de la suite et ses entrées.

3. Dans l'onglet Reports (page 272) (Rapports), spécifiez le type de rapport que vous voulez créer lors de l'exécution.

4. Dans l'onglet Metrics (page 274) (Mesures), spécifiez les mesures que vous voulez enregistrer lors de l'exécution.

5. Dans l'onglet Documentation (page 276), spécifiez une description pour la suite.

6. Dans le menu principal, sélectionnez File (Fichier), Save (Enregistrer).

Editeur de suites de tests

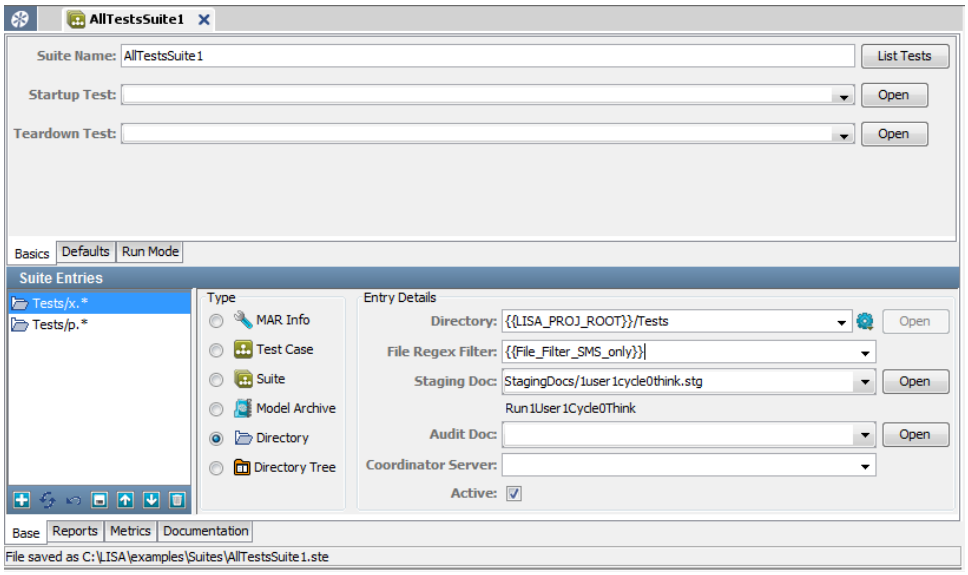
Lorsque vous créez une suite de tests ou lorsque vous ouvrez une suite de tests existante, l'éditeur de suites de tests s'ouvre.

Pour plus d'informations sur les onglets spécifiques de cette page, consultez la rubrique suivante :

- Onglet Base de l'éditeur de suites de tests (page 267)
- Onglet Reports (Rapports) de l'éditeur de suites de tests (page 272)
- Onglet Metrics (Mesures) de l'éditeur de suites de tests (page 274)
- Onglet Documentation de l'éditeur de suites de tests (page 276)

Onglet Base de l'éditeur de suites de tests

L'onglet Base de l'éditeur de suites de tests contient des informations de base sur une suite de tests.



Panneau supérieur

Le panneau supérieur de l'onglet Base contient des informations de base sur la totalité de la suite de tests.

Pour configurer une suite, vous devez également entrer des paramètres dans tous les sous-onglets.

Onglet Basics (Paramètres de base)

Cet onglet comprend les paramètres suivants :

Suite Name (Nom de la suite)

Nom de la suite

Startup Test (Test de démarrage)/Teardown Test (Test de désinstallation)

Vous pouvez spécifier un test de démarrage qui s'exécute avant le démarrage de la suite et un test de désinstallation qui s'exécute à la fin de la suite. Le test de démarrage et le test de désinstallation ne sont pas inclus dans les statistiques de test. Les événements figurant dans ces tests s'affichent dans les rapports. Si un échec du test de démarrage se produit, l'exécution de la suite de tests s'arrête. Vous pouvez utiliser un fichier d'informations MAR comme test de démarrage ou de désinstallation. L'actif principal du fichier d'informations MAR doit être un scénario de test.

List Tests (Répertorier les tests)

Affiche une boîte de dialogue contenant une liste des tests actuellement inclus dans la suite. Enregistrez la suite pour actualiser la liste.

Onglet Defaults (Valeurs par défaut)

Cet onglet comprend les paramètres suivants :

Base Directory (Répertoire de base)

Nom du répertoire qui est considéré comme le répertoire de base pour un test qui ne contient pas un chemin complet. Si un test est introuvable ailleurs, DevTest recherchera dans ce répertoire. Si vous ne spécifiez aucun répertoire de base, une valeur par défaut est créée lorsque le document de suite est enregistré.

Default Staging Doc (Document de simulation par défaut)

Si aucun document de simulation n'est spécifié pour un test, le document de simulation par défaut est utilisé.

Default Audit Doc (Document d'audit par défaut)

Si aucun document d'audit n'est spécifié pour un test, le document d'audit par défaut est utilisé.

Default Coordinator Server (Serveur de coordination par défaut)

Si aucun serveur de simulation n'est spécifié pour un test, le serveur de simulation par défaut est utilisé. Le menu déroulant répertorie les serveurs de coordination disponibles. Ce paramètre est requis uniquement si vous utilisez DevTest Server.

Onglet Run Mode (Mode d'exécution)

Cet onglet comprend les paramètres suivants :

Serial (En série)

Les tests sont exécutés l'un après un autre dans l'ordre spécifié dans le document de suite. Ce paramètre est utilisé pour les tests fonctionnels et de régression.

Parallel (Parallèle)

Les tests sont exécutés simultanément. Ce paramètre est utilisé pour les tests de charge et de performances. Vous devez disposer d'un nombre d'utilisateurs virtuels suffisant pour pouvoir exécuter tous les tests simultanément.

Allow Duplicate Test Runs (Autoriser la duplication des exécutions de test)

Permet de sélectionner si le même test doit être exécuté plusieurs fois. Si le même test figure dans une suite incluse, un répertoire ou une arborescence de répertoires, des tests dupliqués peuvent se produire dans une suite.

Le mode d'exécution parent pour la suite parente est automatiquement appliqué aux suites enfants.

Panneau inférieur

Le panneau inférieur de l'onglet Base affiche les types de documents que vous pouvez ajouter dans la suite et les détails des documents associés.

Ce panneau vous permet de créer une suite en ajoutant des tests, des suites existantes, des répertoires et des arborescences de répertoires qui contiennent des tests.

Une liste des entrées de la suite est affichée dans le panneau gauche, une fois que tous les tests ont été ajoutés et enregistrés dans le document de suite.

Les entrées de suite sont ajoutées une à une en les sélectionnant à partir des types suivants. Selon la sélection, la zone Entry Details (Détails de l'entrée) varie.

Type

Sélectionnez le type d'entrée de suite :

- MAR Info (Fichier d'informations MAR) **:** nom d'un fichier d'informations MAR.
- Test Case (Scénario de test) **:** nom d'un scénario de test.
- Suite **:** nom d'un document de suite. Tous les tests sont extraits de cette suite et exécutés comme des tests individuels.
- Model Archive (Archive de modèle) **:** nom d'un fichier MAR.
- Directory (Répertoire) **:** nom d'un répertoire qui contient les tests à inclure dans la suite. Les mêmes document de simulation, document d'audit et serveur de coordination sont utilisés pour tous les tests dans ce répertoire. Si les nouveaux scénarios de test sont ajoutés au répertoire, ils sont automatiquement inclus dans cette suite.
- Directory Tree (Arborescence de répertoires) **:** nom d'un répertoire qui contient les tests à inclure dans la suite. DevTest vérifie le répertoire nommé et de façon récursive tous les sous-répertoires de l'arborescence. Les mêmes document de simulation, document d'audit et serveur de coordination sont utilisés pour tous les tests dans cette arborescence de répertoires. Si les nouveaux scénarios de test sont ajoutés à l'arborescence de répertoires, ils sont automatiquement inclus dans cette suite.

Vous pouvez filtrer les scénarios de test par nom pour les répertoires et les arborescences de répertoires. Pour filtrer les résultats, entrez une expression régulière dans le champ File Regex Filter (Filtre de fichier par expression régulière). Vous pouvez entrer plusieurs filtres pour une suite en cliquant sur Add (Ajouter)  dans la barre d'outils et en entrant de nouvelles informations de filtre. Vous pouvez également définir des filtres dans des propriétés et utiliser les propriétés dans le champ File Regex Filter. Pour afficher les tests qui seront inclus dans votre suite, utilisez le bouton List Tests (Liste des tests). Pour afficher les derniers résultats, enregistrez la suite.

La case à cocher Active (Actif) vous permet de contrôler si l'entrée sélectionnée s'exécute. Par exemple, vous pouvez décocher la case Active pour un des scénarios de test d'une suite.

Ajout d'un type de document dans une suite

Les champs des détails d'entrée varient selon le type de document sélectionné.

Par exemple, vous voulez ajouter un document d'informations MAR. Les détails d'entrée pour ce document ne sont plus les mêmes que ceux affichés dans l'arborescence des répertoires.

MAR Info (Informations sur la réponse)

Nom du fichier d'informations MAR. Entrez le nom, sélectionnez-le dans le menu déroulant ou accédez au fichier ou au répertoire. Lorsqu'il est sélectionné, cliquez sur Open (Ouvrir) pour ouvrir le fichier correspondant.

Staging doc (Document de simulation)

Nom du document de simulation pour cette entrée. Entrez le nom, sélectionnez-le dans le menu déroulant ou accédez au document. Si vous laissez ce champ vide, le document de simulation par défaut est utilisé. Lorsqu'il est sélectionné, cliquez sur Open (Ouvrir) pour l'ouvrir. Lorsque vous avez sélectionné un document de simulation à utiliser dans la suite, le nom du document de simulation s'affiche en dessous du champ Staging Doc (Document de simulation). Le nom est identique au nom spécifié dans le champ Run Name (Nom de l'exécution) de l'onglet d'éditeur, lorsque vous modifiez le document.

Audit Doc (Document d'audit)

Nom du document d'audit pour cette entrée. Entrez le nom, sélectionnez-le dans le menu déroulant ou accédez au document. Si vous laissez ce champ vide, le document d'audit par défaut est utilisé. Lorsqu'il est sélectionné, cliquez sur Open (Ouvrir) pour l'ouvrir.

Serveur de coordination

Nom du serveur de coordination à utiliser avec cette entrée. Sélectionnez un serveur dans la liste des serveurs de coordination disponibles répertoriés dans le menu déroulant. Ce paramètre est requis uniquement si vous utilisez DevTest Server.

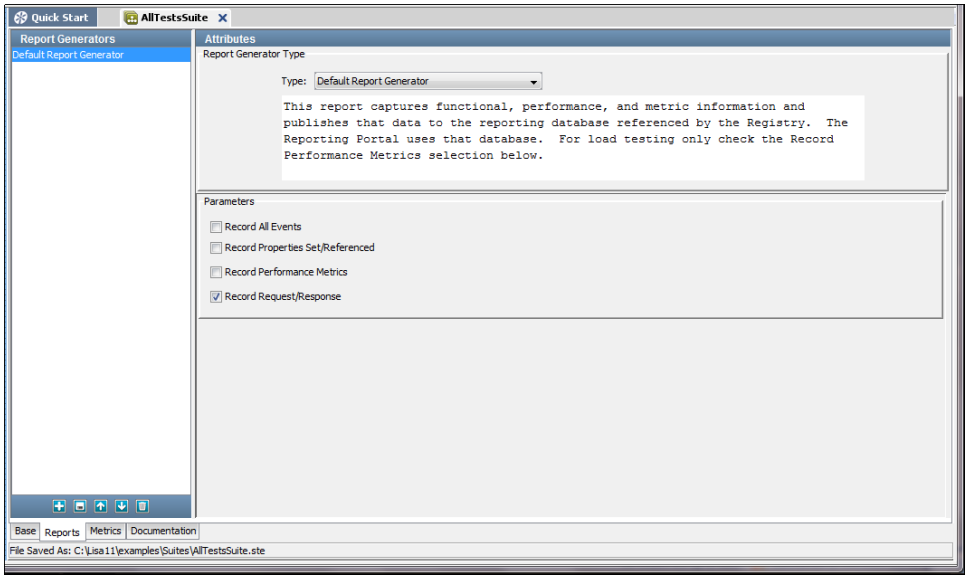
Cliquez sur Add (Ajouter)  dans la barre d'outils pour ajouter cette entrée à la liste affichée dans le panneau de gauche. Vous pouvez supprimer des entrées en cliquant sur Delete (Supprimer)  . Vous pouvez réorganiser les entrées à l'aide des icônes Move Up (Déplacer vers le haut)  et Move Down (Déplacer vers le bas)  .

Après avoir indiqué toutes les entrées de test, enregistrez le document de suite de tests.

Onglet Reports (Rapports) de l'éditeur de suites de tests

L'onglet Reports de l'éditeur de suites de tests vous permet de spécifier les rapports de test à générer et les événements que vous voulez capturer au niveau de la suite de tests.

Les détails des rapports disponibles dans chaque générateur de rapports, l'affichage des rapports, le contenu des rapports et les options de sortie sont traités dans la rubrique Rapports (page 403). Les mesures à capturer pour les rapports sont traitées en détails dans la rubrique Mesures (page 265).



L'onglet Reports (Rapports) comporte les panneaux suivants :

- Report Generators (Générateurs de rapports)
- Attributes (Attributs)

Report Generators (Générateurs de rapports)

Le panneau Report Generators contient une liste des rapports qui ont été sélectionnés.

Pour ajouter un rapport, cliquez sur Add (Ajouter) au bas du panneau de liste et sélectionnez le type de rapport dans le menu déroulant Type au milieu du panneau Reports.

Pour supprimer un rapport, sélectionnez-le dans la liste, puis cliquez sur Delete (Supprimer).

Attributes (Attributs)

Le panneau Attributes répertorie les types de rapport disponibles.

Les types de rapports suivants sont disponibles :

- Default Report Generator (Générateur de rapports par défaut) (page 404)
- XML Report Generator (Générateur de rapports XML) (page 405)

Parameters (Paramètres)

Record All Events (Enregistrer tous les événements)

Si cette option est sélectionnée, tous les événements sont enregistrés.

Record Properties Set/Referenced (Enregistrer les propriétés définies/référencées)

Si cette option est sélectionnée, l'ensemble des propriétés ou les propriétés référencées sont enregistrées.

Record Performance Metrics (Enregistrer les mesures de performances)

Si cette option est sélectionnée, les mesures de performances sont enregistrées. Utilisez cette valeur pour le test de charge.

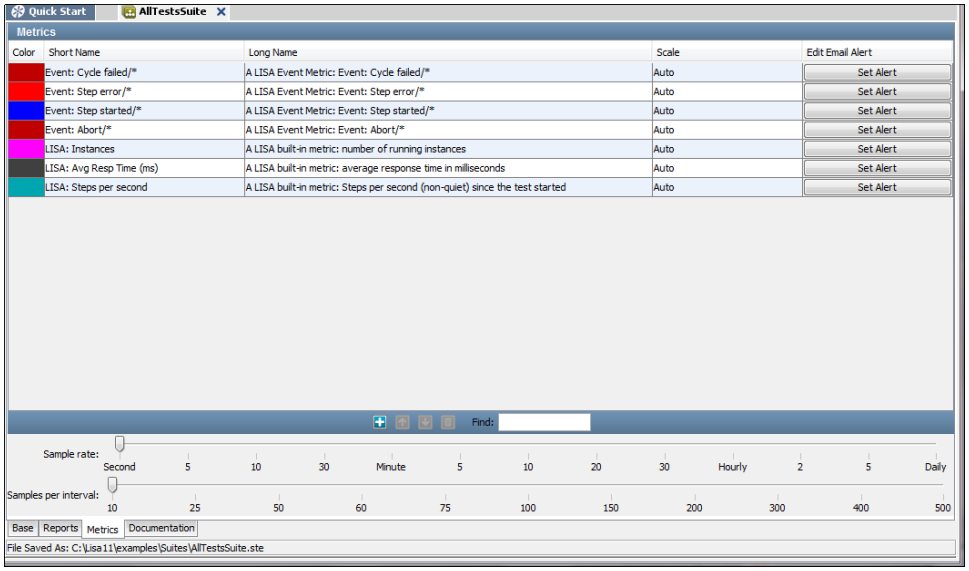
Record Request/Response (Enregistrer la demande/réponse)

Si cette option est sélectionnée, la demande et la réponse sont enregistrées.

Onglet Metrics (Mesures) de l'éditeur de suites de tests

L'onglet Metrics (Mesures) de l'éditeur de suites de tests permet de sélectionner les mesures de test à enregistrer. Cet onglet vous permet également de définir les spécifications d'échantillonnage pour la collection des mesures.

Vous pouvez également définir une alerte par courriel pour une mesure que vous avez sélectionnée.



L'onglet Metrics est divisé selon les sections suivantes :

Le panneau supérieur affiche les mesures par défaut qui sont ajoutées à la suite. Vous pouvez ajouter d'autres mesures en cliquant sur Add (Ajouter) dans la barre d'outils.

Dans le panneau inférieur, deux barres de curseur vous permettent de définir les paramètres d'échantillonnage :

- Sample rate (Taux d'échantillonnage) **:** cette valeur spécifie la fréquence à laquelle effectuer un échantillonnage ou enregistrer la valeur d'une mesure. Ce paramètre est la valeur réciproque du taux d'échantillonnage spécifié sous la forme d'une période.
- Samples per interval (Echantillons par intervalle) **:** cette valeur spécifie le nombre d'échantillons utilisés pour créer un intervalle pour calculer des valeurs récapitulatives pour la mesure.

Effectuer un échantillonnage toutes les minutes (Sample rate=1 minute) et la moyenne tous les 60 échantillons (Samples per interval=60) renvoie une mesure toutes les minutes et une valeur récapitulative (moyenne) toutes les heures.

Par exemple, la valeur par défaut est un taux d'échantillonnage de 1 seconde et 10 échantillons par intervalle (intervalle de 10 secondes).

Les mesures sont stockées dans des fichiers XML ou des tables de base de données à inclure dans les rapports (voir la rubrique Rapports).

Ajout de mesures

Les catégories de mesures suivantes sont disponibles :

- Mesures DevTest Whole Test Metrics (Mesures DevTest de test complet)
- Mesures DevTest Test Event Metrics (Mesures d'événement de test DevTest)
- Mesures JMX Metrics (Mesures SNMP)
- Mesures SNMP Metrics (Mesures SNMP)
- Mesures TIBCO Hawk Metrics (Mesures TIBCO Hawk)
- Windows Perfmon Metrics (Mesures Windows PerfMon)
- Mesures UNIX Metrics Via SSH (Mesures UNIX via SSH)

Pour ajouter des mesures :

1. Cliquez sur Add (Ajouter)  dans la barre d'outils.

   La boîte de dialogue Add Metric (Ajouter une mesure) s'affiche.
2. Sélectionnez le type de mesure et cliquez sur OK.
3. Pour configurer des mesures à partir des autres catégories, répétez la procédure.

Pour obtenir une description de toutes les mesures de toutes les catégories et des informations sur leur configuration pour les inclure dans des rapports, consultez la rubrique Génération de mesures (page 265).

Définition d'alertes par courriel

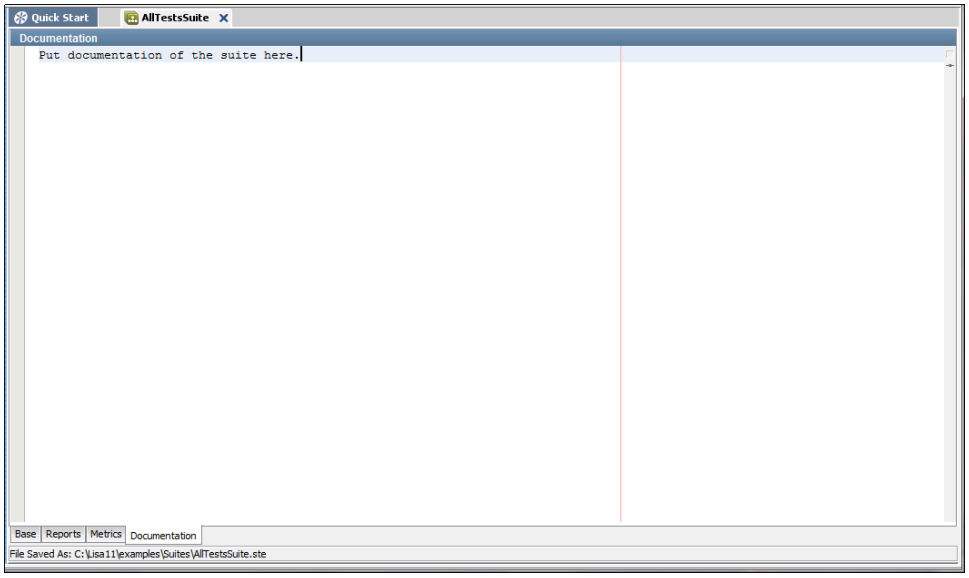
Procédez comme suit:

1. Cliquez sur le bouton Set Alert (Définir une alerte) correspondant à la mesure.

   La boîte de dialogue Edit Alert (Modifier l'alerte) s'ouvre.
2. Vous pouvez définir les limites acceptables pour la mesure (valeur basse et élevée) et les détails à envoyer dans un courriel. Fournissez le nom du serveur de messagerie et une liste d'adresses électroniques.
3. Vous pouvez ajouter et supprimer des adresses électroniques à l'aide des icônes Add (Ajouter) et Delete (Supprimer) au bas de la fenêtre.
4. Une fois que vous avez terminé, cliquez sur Close (Fermer).

Onglet Documentation de l'éditeur de suites de tests

L'onglet Documentation de l'éditeur de suites de tests vous permet d'entrer la documentation associée.



Chapitre 12: Utilisation des archives de modèle (MAR)

L'artefact de déploiement principal est un type de fichier nommé Model Archive (Archive de modèle, MAR).

Vous pouvez configurer les fichiers MAR à partir de DevTest Workstation ou à l'aide de l'utilitaire de ligne de commande Make Mar.

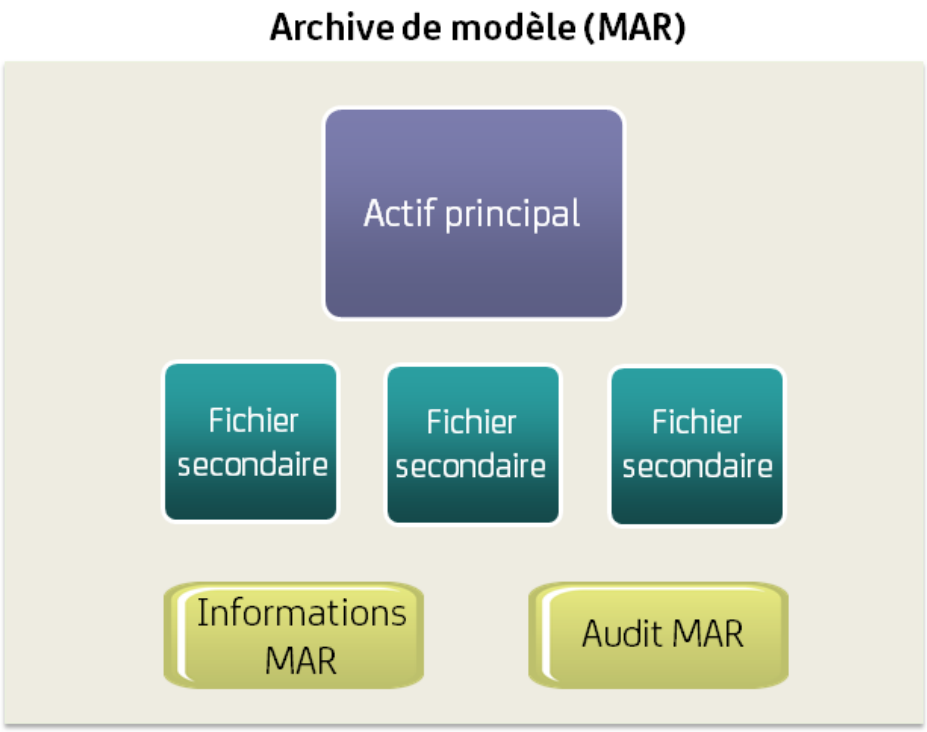
Ce chapitre traite des sujets suivants :

Présentation des archives de modèle (MAR)

L'artefact de déploiement principal est un type de fichier nommé Model Archive (Archive de modèle, MAR). L'extension de fichier est **.mar**.

Les fichiers MAR contiennent les éléments suivants :

- Un actif principal
- Tous les fichiers secondaires nécessaires à l'exécution de l'actif principal
- Un fichier d'informations
- Un fichier d'audit



Les fichiers MAR sont créés à partir de fichiers dans un projet. Une fois qu'un fichier MAR est créé, il ne dépend plus du projet.

Contenu d'un fichier MAR

Les fichiers MAR contiennent un des actifs principaux suivants :

- Scénario de test
- Suite
- Service virtuel
- Moniteur de scénario de test
- Moniteur de suite

Les fichiers MAR contiennent également tous les fichiers secondaires requis par l'actif principal.

Par exemple, si l'actif principal est un modèle de service virtuel, le fichier MAR contient également une image de service.

En outre, les fichiers MAR contiennent les fichiers suivants :

- Fichier d'informations MAR (page 279)
- Fichier d'audit MAR (page 280)

Vous pouvez indiquer qu'un fichier MAR doit être optimisé. Lorsque le fichier MAR est créé, uniquement les fichiers de projet requis seront ajoutés. Toutefois, vous pouvez également configurer un fichier MAR optimisé pour inclure un ou plusieurs fichiers de projet non requis.

Si un fichier MAR n'est pas optimisé, tous les fichiers de projet seront ajoutés.

Remarque : Une archive est généralement conservée dans la mémoire. Par conséquent, l'utilisation d'archives optimisées est fortement recommandée.

Fichier d'informations MAR

Les fichiers d'informations MAR contiennent les informations nécessaires à la création d'un fichier MAR. L'extension de fichier est **.mari**.

Les informations qui sont spécifiées dans un fichier d'informations MAR dépendent du type d'actif principal.

Actif principal	Informations spécifiées
Scénario de test	Scénario de test, fichier de configuration et document de simulation
Suite	Suite et fichier de configuration
Service virtuel	Modèle de service virtuel, fichier de configuration, capacité simultanée, pourcentage de délai de réflexion et indicateur de redémarrage automatique

Moniteur de scénario de test	Toutes les informations spécifiées pour un scénario de test, ainsi que le nom de service, le courriel de notification, la priorité et la planification d'exécution.
Moniteur de suite	Toutes les informations spécifiées pour une suite, ainsi que le nom de service, le courriel de notification, la priorité et la planification d'exécution.

Les fichiers d'informations MAR se trouvent dans le dossier Tests, Suites ou VirtualServices d'un projet. Lorsque vous utilisez les options de simulation ou de déploiement, un fichier d'informations MAR est créé, mais n'est pas enregistré.

Le projet **Examples** (Exemples) contient des fichiers d'informations MAR que vous pouvez consulter.

Fichier d'audit MAR

Les fichiers d'audit MAR contiennent des métadonnées sur la création d'un fichier MAR. L'extension de fichier est **.maraudit**.

Les métadonnées suivantes sont incluses dans un fichier d'audit MAR :

- Date et heure de création du fichier MAR
- Nom de l'ordinateur sur lequel le fichier MAR a été créé.
- Répertoire racine du projet à partir duquel le fichier MAR a été créé.
- Nom du créateur du fichier MAR

Création explicite et implicite de fichier MAR

Deux approches s'offrent à vous pour créer des fichiers MAR :

- Création explicite (page 281)
- Création implicite (page 281)

Pour utiliser les tableaux de bord Web ou les utilitaires de ligne de commande pour déployer des actifs, vous devez créer un fichier MAR de manière explicite. Une exception est l'utilitaire de ligne de commande Test Runner (page 341) (Exécuteur de tests) que vous pouvez utiliser pour exécuter des scénarios de test et des suites qui ne sont pas mises en package dans un fichier MAR.

Pour simuler un scénario de test ou une suite, déployer un service virtuel, déployer un scénario de test comme moniteur ou déployer une suite comme moniteur à partir de DevTest Workstation, utilisez l'approche implicite.

Création explicite de fichier MAR

Dans l'approche explicite, vous effectuez des étapes pour créer un fichier d'informations MAR, que vous utilisez ensuite pour créer le fichier MAR.

Procédez comme suit:

1. Identifiez l'actif principal que vous voulez exécuter.
2. Créez un fichier d'informations MAR.
3. Générez le fichier MAR.

Création implicite de fichier MAR

Dans l'approche implicite, un fichier d'informations MAR et un fichier MAR sont automatiquement créés.

Procédez comme suit:

1. Identifiez l'actif principal que vous voulez exécuter.
2. A partir de DevTest Workstation ou de l'utilitaire Test Runner (page 341) (Exécuteur de tests), effectuez l'une des actions suivantes :
   - Simuler un scénario de test
   - Simuler une suite
   - Déployer un service virtuel
   - Déployer un scénario de test en tant que moniteur
   - Déployer une suite en tant que moniteur

Création de fichiers d'informations MAR

Vous pouvez créer un fichier d'informations MAR à partir d'un scénario de test, d'une suite ou d'un service virtuel existant.

Pour créer un fichier d'informations MAR à partir d'un scénario de test existant :

1. Dans le panneau Project (Projet) de DevTest Workstation, cliquez avec le bouton droit de la souris sur le scénario de test et sélectionnez Create MAR Info File (Créer un fichier d'informations MAR).

   La boîte de dialogue Create MAR Info File s'ouvre.

2. Dans le champ Name (Nom), entrez le nom du fichier d'informations MAR.
3. Dans la liste de champs Configuration, sélectionnez le fichier de configuration pour ce scénario de test.
4. Dans la liste de champs Staging Doc (Document de simulation), sélectionnez le document de simulation pour ce scénario de test.
5. Dans la liste de champs Coordinator Server (Serveur de coordination), sélectionnez le serveur de coordination pour ce scénario de test (le cas échéant).
6. Cliquez sur OK (Redéployer/déployer).

   Le fichier .mari est créé dans le même dossier dans lequel le scénario de test réside.

Pour créer un fichier d'informations MAR à partir d'une suite existante :

1. Dans le panneau Project (Projet) de DevTest Workstation, cliquez avec le bouton droit de la souris sur la suite et sélectionnez Create MAR Info File (Créer un fichier d'informations MAR).

   La boîte de dialogue Create MAR Info File s'ouvre.

2. Dans le champ Name (Nom), entrez le nom du fichier d'informations MAR.
3. Dans la liste de champs Configuration, sélectionnez le fichier de configuration pour la suite.
4. Cliquez sur OK (Redéployer/déployer).

   Le fichier .mari est créé dans le même dossier dans lequel la suite réside.

Pour créer un fichier d'informations MAR à partir d'un service virtuel existant :

1. Dans le panneau Project (Projet) de DevTest Workstation, cliquez avec le bouton droit de la souris sur le service virtuel et sélectionnez Create MAR Info File (Créer un fichier d'informations MAR).

La boîte de dialogue Create MAR Info File s'ouvre.

2. Dans le champ Name (Nom), entrez le nom du fichier d'informations MAR.
3. Dans la liste de champs Configuration, sélectionnez le fichier de configuration pour le service virtuel.
4. Si ce service virtuel fera partie d'un groupe de services virtuels (page 569), entrez un nom de groupe dans le champ Group Tag (Balise de groupe). Une balise group doit commencer par un caractère alphanumérique et peut contenir des caractères alphanumériques et quatre autres caractères : le point (.), le trait d'union (-), le trait de soulignement (_) et le signe dollar ($).
5. Pour indiquer la capacité de charge, entrez un nombre dans le champ Concurrent capacity (Capacité simultanée).

   La valeur par défaut est 1, mais vous pouvez l'augmenter pour fournir une capacité supplémentaire. La capacité correspond au nombre d'utilisateurs virtuels (instances) qui peuvent être exécutés simultanément avec le modèle de service virtuel. Dans cet exemple, la capacité indique le nombre de threads existants pour les demandes de service de ce modèle de service.
6. Entrez le pourcentage du délai de réflexion (en fonction du délai de réflexion enregistré) dans le champ Think Time Scale (Pourcentage de délai de réflexion).

   Pour doubler le délai de réflexion, utilisez 200. Pour réduire de moitié le délai de réflexion, utilisez 50. La valeur par défaut est 100.
7. Cochez ou décochez la case Start the service on deployment (Lancer le service au déploiement).

   Cette case à cocher indique si le service démarre immédiatement lors du déploiement. Par défaut, le service démarre immédiatement.
8. Cochez ou décochez la case If service ends, automatically restart it (Redémarrer automatiquement le service).

   Cochez cette case permet de maintenir l'exécution du service, même lorsqu'une session d'émulation se termine.
9. Cliquez sur OK (Redéployer/déployer).

   Le fichier .mari est créé dans le même dossier dans lequel le service virtuel réside.

Création de fichiers d'informations MAR de moniteur

Vous pouvez créer un fichier d'informations MAR de moniteur à partir d'un scénario de test ou d'une suite existante.

Pour plus d'informations sur les moniteurs, consultez la rubrique Service de validation en continu (page 385).

Procédez comme suit:

1. Dans le panneau Project (Projet) de DevTest Workstation, cliquez avec le bouton droit de la souris sur le scénario de test ou la suite, et sélectionnez Create Monitor MAR Info File (Créer un fichier d'informations MAR de moniteur).

   La boîte de dialogue Create Monitor MAR Info File s'ouvre.
2. Dans le champ Name (Nom), entrez le nom du fichier d'informations MAR.
3. Spécifiez les informations de moniteur dans l'onglet Monitor Info (page 284) (Informations sur le moniteur).
4. Spécifiez les informations de planification dans l'onglet Schedule (page 285) (Planification).
5. (Scénario de test) Spécifiez les informations de scénario de test dans l'onglet Test Case Info (page 286) (Informations sur le scénario de test).
6. (Suite) Spécifiez les informations de suite dans l'onglet Suite Info (page 286) (Informations sur la suite).
7. Cliquez sur OK (Redéployer/déployer).

   Le fichier .mari est créé dans le même dossier dans lequel le scénario de test ou la suite réside.

Onglet Monitor Info (Informations sur le moniteur)

Entrez les paramètres suivants.

Service Name (Nom du service)

Correspond au nom du service. Ce nom s'affiche dans le tableau de bord CVS Dashboard (Tableau de bord du service CVS).

Notify Email (Adresse électronique de notification)

Adresse électronique à notifier.

Priority (Priorité)

Définissez la priorité sur High (Elevée), Medium High (Moyenne à élevée), Medium (Moyenne), Medium Low (Moyenne à faible) ou Low (Faible).

Onglet Schedule (Planification)

Entrez les paramètres suivants.

Start (Démarrer)

Date et heure à laquelle le moniteur démarre. Pour l'heure, utilisez un format 24 heures.

Stop (Arrêter)

Date et heure à laquelle le moniteur s'arrête. Pour l'heure, utilisez un format 24 heures. La date et heure d'arrêt doit être postérieure à la date et heure de démarrage.

Run (Exécuter)

Indique la fréquence d'exécution du moniteur. En fonction de votre sélection, d'autres options peuvent s'afficher.

- One time **(Unique)** : le moniteur s'exécute une fois, à la date et heure de démarrage.
- Every **(Toutes les)** : entrez la fréquence en minutes. Le moniteur s'exécute toutes les NN minutes. Toutefois, si l'exécution précédente ne s'est pas terminée, le service de validation en continu ignore l'exécution et effectue une nouvelle tentative à la prochaine exécution planifiée. Le service de validation en continu n'arrêtera pas un moniteur après son démarrage.
- Daily **(Quotidienne) :** le moniteur s'exécute une fois par jour, à l'heure spécifiée par l'heure de début.
- Weekly **(Hebdomadaire) :** sélectionnez un ou plusieurs jours de la semaine. Le moniteur s'exécute une fois par jour sélectionné, à l'heure spécifiée par l'heure de début.
- Monthly **(Mensuelle)** : spécifiez un ou plusieurs jours du mois. Si vous entrez plusieurs jours, séparez-les par des espaces. Par exemple, vous pouvez entrer 1 15 30. Si vous spécifiez un jour qui n'est pas récurrent chaque mois (comme le 31), le jour est ignoré pour le mois. Le moniteur s'exécute une fois par jour, à l'heure spécifiée par l'heure de début.
- CRON **:** entrez une spécification cron standard. Cron est une syntaxe UNIX standard permettant de spécifier des intervalles. Cette approche offre la meilleure flexibilité lorsque vous spécifiez la planification.

Onglet Test Case Info (Informations sur le scénario de test)

Entrez les paramètres suivants.

Configuration

Correspond au nom de la configuration. Si ce champ est vide, la configuration active dans le scénario de test est utilisée.

Staging doc (Document de simulation)

Nom du document de simulation.

Coordinator server (Serveur de coordination)

Nom du serveur de coordination.

Onglet Suite Info (Informations sur la suite)

Entrez les paramètres suivants.

Configuration

Correspond au nom de la configuration.

Modification de fichiers d'informations MAR

L'éditeur de fichiers d'informations MAR vous permet de modifier les informations spécifiées pendant la création du fichier d'informations MAR.

L'éditeur inclut une case à cocher d'optimisation. Cette case à cocher est activée par défaut. Lorsque le fichier MAR est généré, uniquement les fichiers de projet requis par l'actif principal sont ajoutés. Toutefois, l'éditeur vous permet de spécifier un fichier MAR optimisé qui inclut un ou plusieurs fichiers de projet non requis.

Si vous décochez la case Optimize (Optimiser), tous les fichiers de projet sont ajoutés au fichier MAR.

Pour modifier les fichiers d'informations MAR :

1. Dans le panneau Project (Projet) de DevTest Workstation, double-cliquez sur le fichier d'informations MAR.

   L'éditeur s'ouvre.

   Les informations que vous pouvez modifier dans la partie supérieure dépendent du type de fichier d'informations MAR.
2. Dans la zone Notes, ajoutez des remarques sur le fichier d'informations MAR.
3. Cochez ou décochez la case Optimize (Optimiser).

   Si vous voulez que le fichier MAR inclue tous les fichiers de projet, décochez la case Optimize. Les zones Files to Include (Fichiers à inclure) et Files to Exclude (Fichiers à exclure) sont désactivées.

   Remarque : Une archive est généralement conservée dans la mémoire. Par conséquent, l'utilisation d'archives optimisées est fortement recommandée. Cette case à cocher est sélectionnée par défaut.
4. Si vous voulez que le fichier MAR inclue tous les fichiers de projet non requis, déplacez les fichiers dans la zone Files to Include.

   La zone Files to Include répertorie les fichiers de projet requis, tandis que la zone Files to Exclude répertorie les fichiers de projet non requis.
5. A partir du menu principal, sélectionnez File (Fichier), Save (Enregistrer).

Génération de fichiers MAR

Après avoir créé et modifié un fichier d'informations MAR, vous pouvez utiliser le fichier pour créer une archive de modèle (MAR) (page 278).

Vous pouvez enregistrer le fichier MAR dans un dossier inclus dans le répertoire de projet ou en dehors.

Procédez comme suit:

1. Dans le panneau Project (Projet) de DevTest Workstation, cliquez avec le bouton droit de la souris sur le fichier d'informations MAR et sélectionnez Build Model Archive (Générer une archive de modèle).

   La boîte de dialogue Build Model Archive s'affiche.

2. Accédez au dossier dans lequel vous voulez enregistrer le fichier MAR.
3. Cliquez sur Save (Enregistrer).

Déploiement vers le service de validation en continu

Vous pouvez déployer un fichier d'informations MAR de moniteur (page 284) sur un service de validation en continu.

Procédez comme suit:

1. Dans le panneau Project (Projet) de DevTest Workstation, cliquez avec le bouton droit de la souris sur le fichier .mari et sélectionnez Deploy to CVS (Déployer vers le service de validation en continu).

   Un message de confirmation s'affiche.

2. Pour afficher le moniteur nouvellement ajouté, accédez au tableau de bord CVS Dashboard (Tableau de bord du service CVS).

Utilitaire Make Mar

Vous pouvez utiliser l'utilitaire de ligne de commande Make Mar pour afficher le contenu du fichiers d'informations MAR (autonome ou dans une archive) ou pour créer des fichiers d'archive de modèle à partir de fichiers d'informations MAR.

Cet utilitaire se trouve dans le répertoire **LISA_HOME/bin.**

Options

Chaque option comporte une version courte et une version longue. La version courte commence par un trait d'union unique, tandis que la version longue commence par deux traits d'union.

-h, --help

Affiche le texte d'aide.

-s file-name, --show=file-name

Affiche le contenu d'un fichier d'informations MAR.

Si le nom de fichier fait référence à un fichier d'informations MAR, le contenu est écrit dans le fichier. S'il fait référence à une archive, le contenu est écrit dans l'entrée d'audit et dans l'entrée d'informations.

-c, --create

Crée une ou plusieurs archives de modèle à partir de fichiers d'informations MAR.

Pour spécifier le nom du fichier d'informations MAR pour créer l'archive, utilisez l'argument --marinfo.

Pour spécifier le nom de l'archive à créer, utilisez l'argument --archive.

Si l'argument --source-dir est utilisé, l'argument --marinfo est utilisé comme modèle de nom à rechercher dans l'arborescence des répertoires complète. Dans ce cas, l'argument --target-dir doit être utilisé pour noter l'emplacement dans lequel placer les archives créées. Cette commande implique également l'attribution automatique de nom pour les archives créées.

-m mar-info-name, --marinfo=mar-info-name

Paramètre qui spécifie le nom du fichier d'informations MAR à lire (si l'argument --source-dir n'est pas utilisé) ou le modèle de nom de fichier d'informations MAR à rechercher (si l'argument --source-dir est utilisé). Dans le dernier cas, s'il n'est pas spécifié, sa valeur est remplacée par la valeur .mari par défaut.

-s directory, --source-dir=directory

Spécifie le répertoire dans lequel l'outil doit rechercher les fichiers d'informations MAR à partir desquels les archives doivent être créées. Les fichiers qui correspondent au modèle spécifié par l'argument --marinfo seront recherchés dans la totalité de l'arborescence des répertoires.

-t output-directory, --target-dir=output-directory

Spécifie le répertoire dans lequel les archives doivent être écrites.

Lorsque cet argument est utilisé, les archives créées sont nommées automatiquement en ajoutant un chiffre à la fin du nom de fichier d'informations MAR afin de garantir qu'aucun fichier ne soit écrasé.

-a archive-file, --archive=archive-file

Spécifie le nom du fichier d'archive à créer.

Lorsque cet argument est utilisé, l'argument --marinfo doit spécifier un fichier d'informations MAR existant unique. Les arguments --source-dir et --target-dir ne sont pas autorisés.

--version

Permet d'imprimer le numéro de version.

Exemples

Les exemples suivantes affichent le contenu d'un fichier d'informations MAR.

```
MakeMar --show=C:\Lisa\examples\MARInfos\AllTestsSuite.mari
```

L'exemple suivant crée une archive de modèle nommée **rawSoap.mar**.

```
MakeMar --create --marinfo=C:\Lisa\examples\MARInfos\rawSoap.mari
--archive=rawSoap.mar
```

L'exemple suivant crée une archive de modèle pour tous les fichiers d'informations MAR dans le répertoire **examples\MARInfos.** Le répertoire de destination **examples\MARs** doit exister avant d'exécuter cette commande.

```
MakeMar --create --source-dir=examples\MARInfos --target-dir=examples\MARs
```

Chapitre 13: Exécution de scénarios de test et de suites

Vous disposez de plusieurs options pour exécuter des scénarios de test.

- Dans DevTest Workstation, vous pouvez utiliser l'utilitaire Interactive Test Run (ITR) (page 293) (Exécution d'un test interactif) pour parcourir et vérifier les étapes d'un scénario de test.
- Dans DevTest Workstation, vous pouvez exécuter un scénario de test rapide avec une configuration minimale, jusqu'à une étape particulière (page 216) du scénario.
- Dans DevTest Workstation, vous pouvez exécuter un scénario de test rapide (page 306) avec une configuration minimale.
- Dans DevTest Workstation, vous pouvez simuler un scénario de test (page 316). Dans ce cas, vous devez spécifier une configuration, un document de simulation et un serveur de coordination. Souvent, les mêmes scénarios de test sont utilisés avec des documents de simulation différents pour effectuer des tests différents. Cela vous permet de ne pas écrire un scénario de test distinct pour les tests fonctionnel, de régression et de charge. A la place, un document de simulation différent est préparé à l'aide du même scénario de test.
- Test Runner (page 341) (Exécuteur de tests) est un utilitaire de ligne de commande qui vous permet d'exécuter des tests dans un processus de traitement par lots.
- LISA Invoke (page 347) est une application Web REST qui vous permet d'exécuter des scénarios de test et des suites avec une URL.
- Dans la console Server Console (Console de serveur), vous pouvez déployer l'archive de modèle (MAR) pour un scénario de test sur un laboratoire (page 382).
- Vous pouvez planifier des tests à exécuter à intervalles réguliers à l'aide du service Continuous Validation Service (CVS) (page 385) (Service de validation en continu).
- Vous pouvez exécuter des tests dans un processus de génération automatique à l'aide de Ant et JUnit.

Ce chapitre traite des sujets suivants :

Utilisation de l'utilitaire Interactive Test Run (Exécution d'un test interactif)

L'utilitaire ITR (Interactive Test Run, exécution d'un test interactif) vous guide dans les étapes d'un scénario de test et vous permet de les vérifier. Avec cet utilitaire, vous pouvez vérifier que les scénarios de test sont en cours d'exécution avant de les simuler. L'ITR exécute le test qui se trouve en mémoire, plutôt que de charger un document de scénario de test. Vous pouvez apporter des modifications au scénario de test en temps réel pour altérer ses propriétés ou le flux de travaux.

Par exemple, vous pouvez exécuter une étape de test en dehors du flux de travaux naturel. Les modifications en temps réel que vous apportez lorsque le test est en cours d'exécution sont intégrées au scénario de test. Le scénario de test continue de s'exécuter sans qu'un redémarrage soit requis. Cependant, lorsque vous effectuez une modification globale, telle qu'une modification de la configuration active, vous devez le redémarrer. Si vous exécutez une étape et vous obtenez un résultat inattendu (par exemple, à cause d'un paramètre incorrect), vous pouvez laisser l'ITR ouvert et modifier l'étape. Vous pouvez ensuite revenir à l'ITR et exécuter la même étape à nouveau.

L'ITR vous permet d'enregistrer l'état de l'exécution du test interactif actuel et de charger un état d'ITR enregistré précédemment. Cette fonctionnalité vous permet de communiquer une erreur, en contexte, à un autre membre de votre équipe. Vous pouvez envoyer le scénario de test et l'état de l'ITR enregistré pour diffuser les actions et les résultats que vous avez observés.

Cette section comprend les rubriques suivantes :

Démarrage d'une exécution de test interactif

Vous pouvez exécuter l'utilitaire Interactive Test Run (Exécution d'un test interactif) à partir d'un scénario de test ouvert dans DevTest Workstation.

Procédez comme suit:

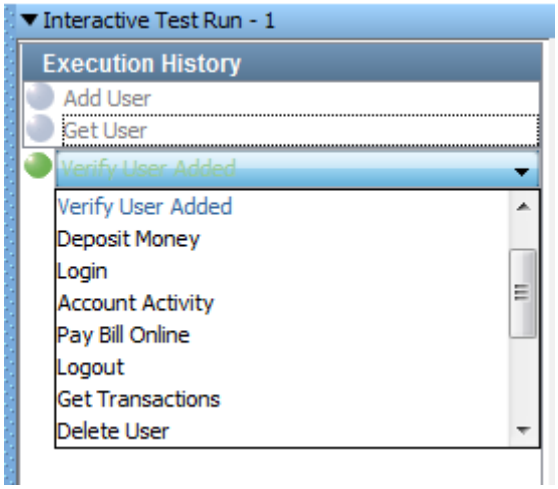
1. Effectuez l'une des opérations suivantes :
   - Dans le menu principal, sélectionnez Actions, Start Interactive Test Run (Démarrer l'exécution d'un test interactif).
   - Dans la barre d'outils, cliquez sur l'icône ITR  . L'icône vous permet de démarrer une nouvelle exécution d'un test interactif ou d'ouvrir une exécution d'un test interactif précédente.

   La fenêtre Interactive Test Run (Exécution d'un test interactif) s'ouvre.
2. (Facultatif) Pour modifier des paramètres, utilisez l'onglet Settings (page 297) (Paramètres).
3. Pour exécuter tout ou partie du scénario de test, utilisez l'onglet Run (page 295) (Exécuter).

Onglet Run (Exécuter) de l'ITR

L'onglet Run (Exécuter) de la fenêtre ITR (Exécution d'un test interactif) indique les étapes exécutées (en gris) et l'étape qui sera exécutée en suivant (en vert).

Dans l'image suivante, les deux premières étapes (Add User (Ajouter un utilisateur) et Get User (Obtenir le nom d'utilisateur)) ont été exécutées. La troisième étape (Verify User Added (Vérifiez l'ajout de l'utilisateur)) sera exécutée en suivant.



Chaque étape a un menu déroulant qui contient toutes les étapes du scénario de test. Vous pouvez utiliser ce menu pour changer l'étape exécutée en suivant. Sélectionnez une étape et elle remplacera l'étape suivante actuelle. Une fois que vous avez changé l'étape, le flux de travaux continuera à partir de la nouvelle étape.

Vous gérez l'ITR à l'aide de la barre d'outils en bas de l'onglet Run (Exécuter).

Permet d'exécuter l'étape suivante dans le scénario de test. Une fois que l'étape a été exécutée, vous pouvez vérifier les résultats (page 298) dans le panneau droit. Cliquez sur l'icône à nouveau pour exécuter l'étape suivante répertoriée ou sélectionner une étape différente à exécuter, comme décrit précédemment.

Permet d'exécuter toutes les étapes du scénario de test. Lors de l'exécution du test, l'icône est remplacée par une icône Stop (Arrêter). Vous pouvez arrêter le test en cliquant sur l'icône Stop. Lorsque la dernière étape a été exécutée, une boîte de dialogue indique que le test est terminé.

Permet de redémarrer le test. Cette fonctionnalité est utile si vous modifiez le scénario de test et voulez l'exécuter depuis le début.

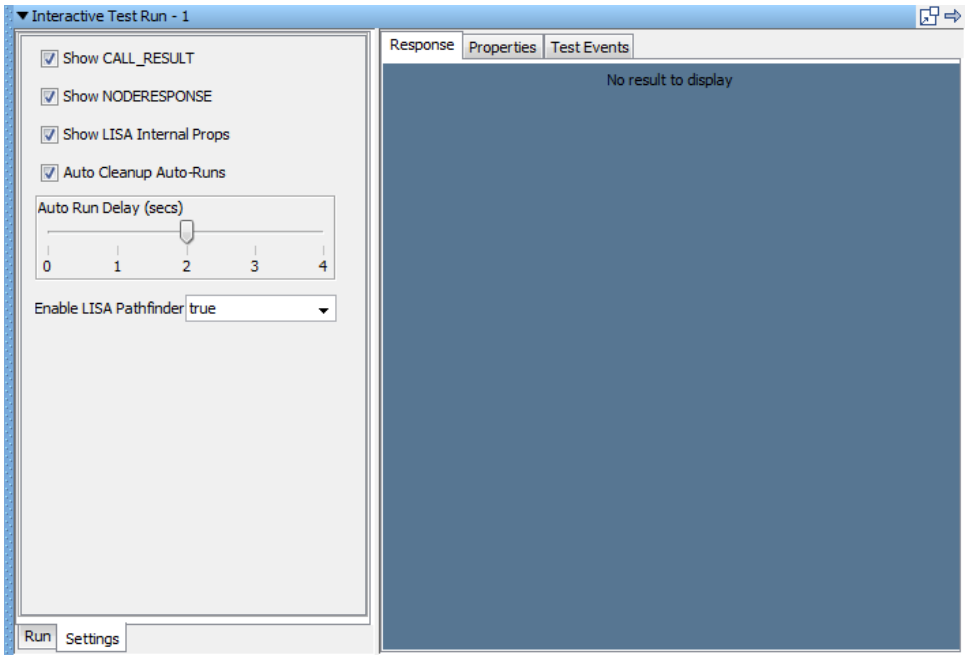
Permet d'enregistrer l'état de l'ITR actuel. Dans la boîte de dialogue, entrez le nom du fichier d'enregistrement. Le suffixe .itr est ajouté au nom. Les enregistrements ultérieurs écrasent ce fichier.

Permet de charger une exécution d'un test interactif enregistré. Accédez au fichier .itr que vous voulez charger. L'état de l'ITR enregistré contient les informations qui sont affichées dans les onglets et capture toutes les données du test effectué. Pour utiliser l'état enregistré, ouvrez le scénario de test qui a été utilisé lorsque les tests d'origine ont été exécutés.

Pour ajouter ou afficher des propriétés, ouvrez la fenêtre des propriétés.

Onglet Settings (Paramètres) de l'ITR

L'onglet Settings (Paramètres) vous permet de contrôler différents aspects de l'ITR.



Vous pouvez filtrer les événements et les propriétés à partir des onglets de résultats. Vous pouvez vouloir exclure certains événements détaillés. De même, vous voulez parfois éviter d'encombrer la liste de propriétés avec celles internes à DevTest.

Show CALL_RESULT (Afficher les valeurs CALL_RESULT)

Permet d'inclure des événements EVENT_CALL_RESULT dans la liste Test Events (Evénements de test).

Show NODERESPONSE (Afficher les valeurs NODERESPONSE)

Permet d'inclure des événements EVENT_NODERESPONSE dans la liste Test Events (Evénements de test).

Show Internal Props (Afficher les propriétés internes)

Permet d'inclure tous les événements internes dans les listes de propriétés des onglets Initial State (Etat initial) et Post Exec State (Etat post-exécution).

Auto Cleanup Auto-Runs (Nettoyer automatiquement les exécutions automatiques)

Permet de nettoyer automatiquement les exécutions automatiques.

Auto Run Delay (secs) (Délai d'exécution automatique, en secondes)

En mode Automatically Execute Test (Exécuter automatiquement le test), l'exécution d'un test interactif est mise en pause entre chaque exécution d'étape. Ce paramètre permet de modifier l'intervalle de pause. L'exécution d'un test interactif ne respecte pas le délai de réflexion. Ce paramètre vous permet donc d'ajouter un délai régulier entre chaque exécution d'étape.

Activation de CA CAI

Permet de contrôler si CAI est activé.

Valeur par défaut : false

Examen des résultats d'une exécution de test interactif

Vous pouvez consulter les résultats de l'exécution d'une étape de test pendant l'exécution, entre deux étapes ou à la fin de l'exécution.

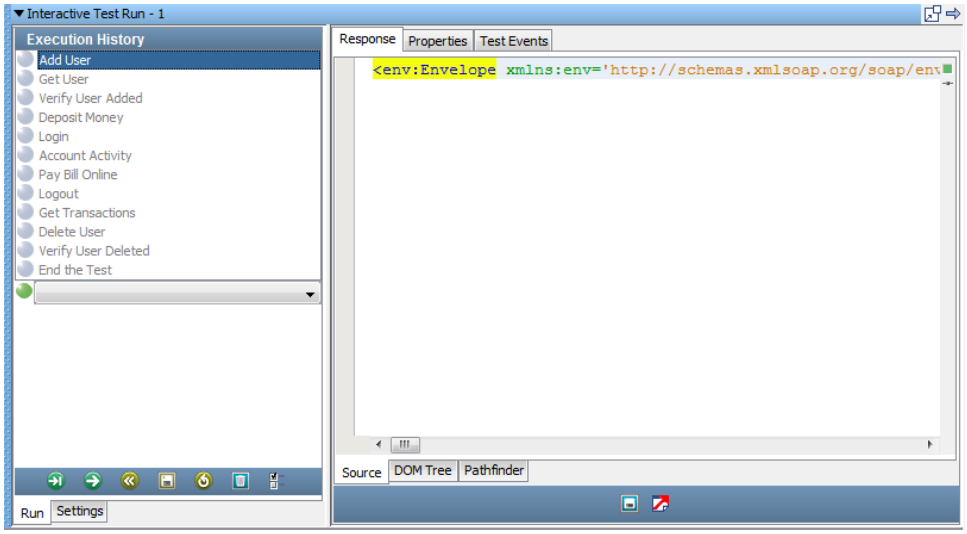
Sélectionnez l'étape dans la liste Execution History (Historique d'exécution). Examinez les informations figurant dans le panneau droit, qui contient les onglets suivants :

- Onglet Response (Réponse) (page 299)
- Onglet Properties (Propriétés) (page 300)
- Onglet Test Events (Evénements de test) (page 301)

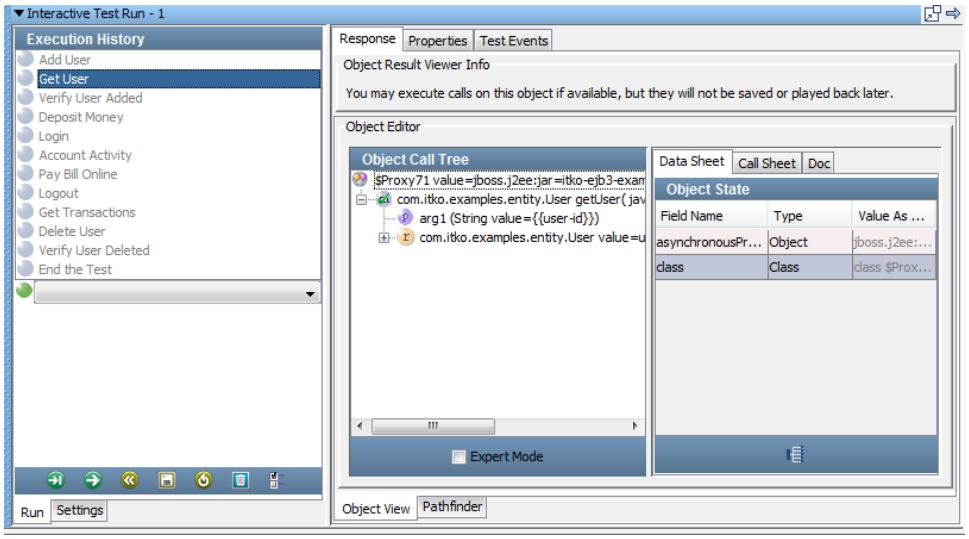
Onglet Response (Réponse) des résultats de l'ITR

L'onglet Response (Réponse) affiche la réponse à l'étape après son exécution.

L'éditeur utilisé pour l'affichage est approprié au type de réponse. Par exemple, l'étape Add User (Ajouter un utilisateur) du scénario de test à plusieurs niveaux invoque la réponse HTML/XML.



L'étape Get User (Obtenir le nom d'utilisateur) du scénario de test à plusieurs niveaux invoque un objet EJB renvoyant un objet qui est affiché dans l'éditeur Complex Object Editor (page 175) (Editeur d'objets complexes).



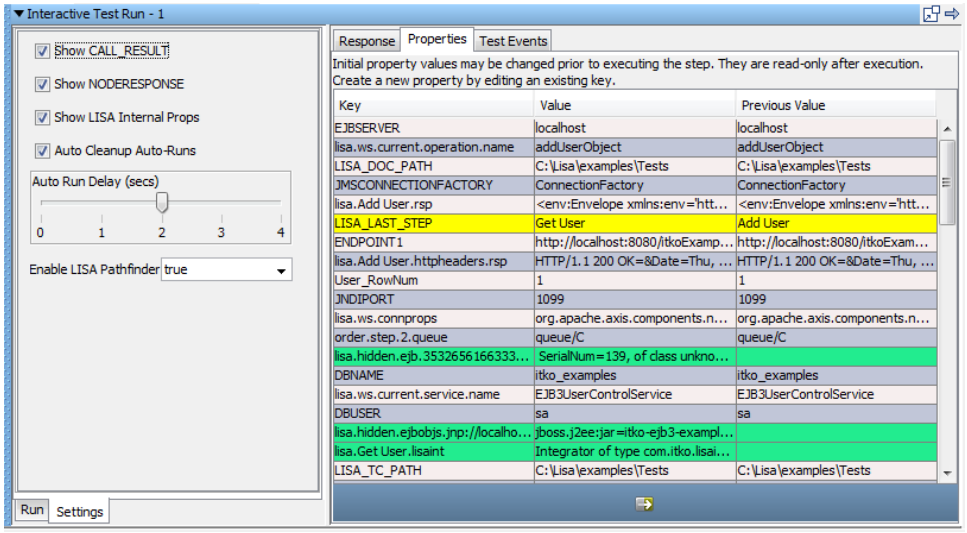
Certains éditeurs incluent deux icônes que vous pouvez utiliser pour enregistrer l'état de l'ITR et lancer un navigateur externe .

Remarque : Si la réponse est un fichier XML supérieur à 5 Mo, il est affiché en texte brut sans vue DOM. Vous pouvez ajuster cette limite avec la propriété **gui.viewxml.maxResponseSize.**

Onglet Properties (Propriétés) des résultats de l'ITR

L'onglet Properties (Propriétés) affiche l'état de l'étape après son exécution (Value (Valeur)) et immédiatement avant son exécution (Previous Value (Valeur précédente)).

Pour afficher ou masquer des propriétés internes DevTest, sélectionnez ou désélectionnez la case à cocher Show LISA Internal Props (Afficher les propriétés internes LISA) dans l'onglet Settings (Paramètres).

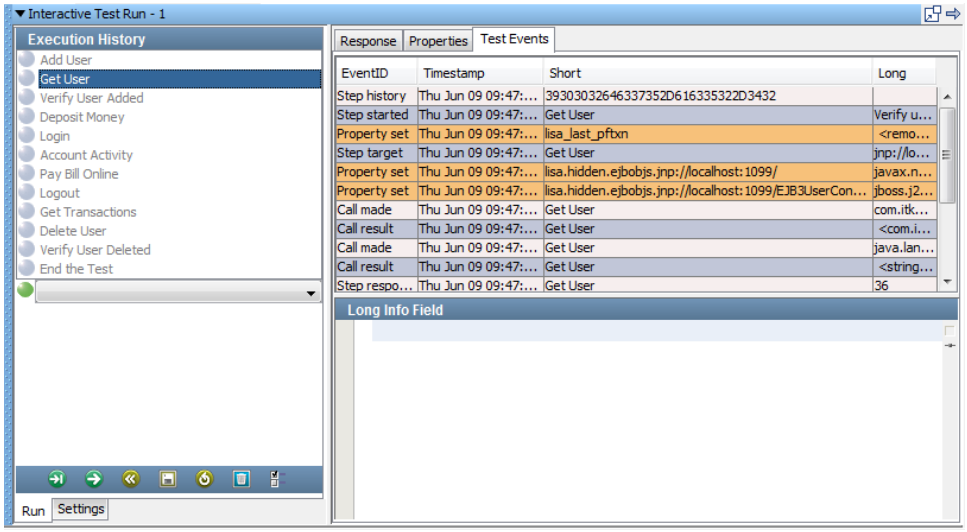


Les propriétés que l'étape définit sont mises en surbrillance en vert. Les propriétés modifiées dans l'étape sont mises en surbrillance en jaune.

Onglet Test Events (Evénements de test) des résultats de l'ITR

L'onglet Test Events (Evénements de test) affiche les événements déclenchés lors de l'exécution de l'étape, dans l'ordre de déclenchement.

Pour afficher ou masquer les événements, cochez ou décochez les cases Show CALL_RESULT (Afficher les valeurs CALL_RESULT) et Show NODERESPONSE (Afficher les valeurs NODERESPONSE) dans l'onglet Settings (Paramètres).



L'onglet Test Events (Evénements de test) contient les colonnes suivantes :

EventID (ID d'événement)

Nom de l'événement.

Timestamp (Horodatage)

Date et heure de l'événement.

Short (Nombre court)

Description brève de l'événement.

Long (Nombre long)

Description longue de l'événement. La description longue peut être tronquée dans l'affichage. Pour afficher le texte complet, cliquez sur la cellule et son contenu complet s'affiche dans le panneau Long Info Field (Informations détaillées).

Vous pouvez entrer la description complète de l'événement dans le panneau Long Info Field.

Un événement est généré pour toutes les assertions déclenchées ou évaluées.

Si un avertissement a été généré, la ligne est jaune.

Si une assertion a été déclenchée, la ligne est violette.

Si une propriété a été définie, la ligne est orange.

Si une erreur a été générée ou si un échec ou une interruption du test s'est produit, la ligne est rouge.

Remarque : Si l'étape appelle un sous-processus qui contient une assertion, l'onglet Test Events (Evénements de test) inclut l'événement approprié pour l'assertion. Par exemple, l'onglet Test Events peut inclure un événement Assertion fired (Assertion déclenchée) qui s'est produit dans le sous-processus.

Utilitaire de comparaison de texte graphique

Un moteur et un visualiseur de comparaison de contenu XML graphique sont disponibles pour comparer les fichiers XML et afficher graphiquement leurs différences.

Démarrage de l'utilitaire de comparaison de texte graphique

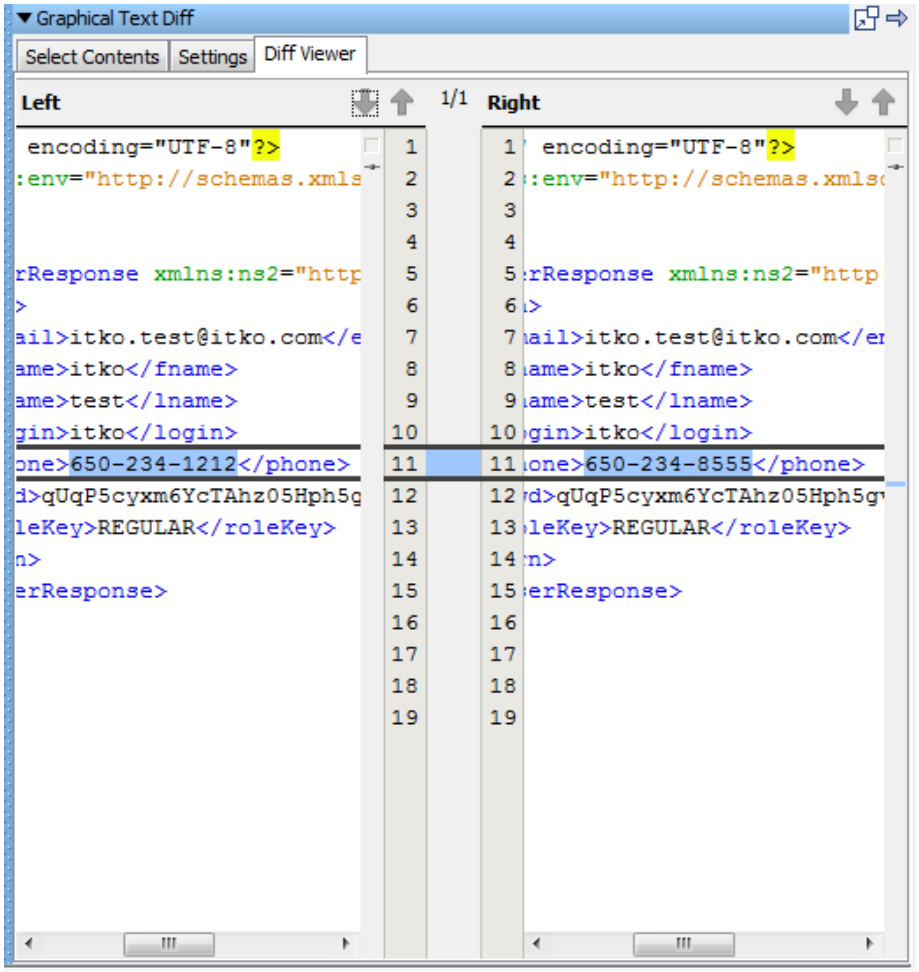
Procédez comme suit:

1. Dans le menu principal, sélectionnez Actions, Graphical Text Diff (Outil de comparaison de texte graphique).

   La fenêtre Graphical Text Diff s'ouvre. A partir de ce panneau, vous pouvez comparer le contenu de deux fichiers XML.

2. Pour lancer la comparaison, cliquez sur Compare (Comparer)  .

   La fenêtre suivante s'ouvre et affiche la comparaison en cours. L'onglet Diff Viewer (Visionneuse de différences) affiche la comparaison.



Pour démarrer l'utilitaire Graphical Text Diff (Outil de comparaison de texte graphique) à partir de l'utilitaire Interactive Test Run (Exécution d'un test interactif) :

1. Sélectionnez l'onglet Test Events (Evénements de test) dans l'ITR
2. Cliquez avec le bouton droit de la souris sur une ligne de la table et sélectionnez Select Left Text to Compare (Sélectionnez le texte à comparer à gauche).
3. Cliquez avec le bouton droit de la souris sur la ligne de table à comparer, puis sélectionnez Compare To (Comparer à).

   L'utilitaire Graphical Text Diff (Outil de comparaison de texte graphique) s'ouvre pour la comparaison.

Dans l'utilitaire Graphical Text Diff, vous pouvez également charger le contenu en sélectionnant un fichier.

Procédez comme suit:

1. Dans la fenêtre Graphical Text Diff (Outil de comparaison de texte graphique), cliquez sur l'onglet Select Contents (Sélectionner du contenu).
2. Pour charger le fichier, cliquez sur Load from File (Charger à partir d'un fichier).

   Les résultats de la comparaison sont affichés dans le composant de panneau de barre d'état global.

Remarque : Vous pouvez également définir les options de comparaison dans l'onglet Settings (Paramètres). Pour plus d'informations sur les options de comparaison, consultez la section Comparaison côte à côte graphique de contenu XML de la rubrique *Utilisation de CA Application Test.*

Propriétés de l'utilitaire de comparaison de texte graphique

L'utilitaire de comparaison de contenu XML graphique utilise les propriétés suivantes. Vous pouvez remplacer ces propriétés dans le fichier **local.properties** ou lors de l'exécution.

lisa.graphical.xml.diff.engine.max.differences

Cette propriété configure le nombre maximum de différences détectées avant que l'algorithme de comparaison de contenu XML s'arrête. Définissez-la sur -1 pour évaluer toutes les différences. Lorsqu'il y a un grand nombre de différences, ce paramètre permet de terminer l'algorithme de comparaison de contenu XML plus rapidement.

Par défaut : 100

lisa.graphical.xml.diff.report.max.linewidth

Cette propriété configure la largeur maximum d'une ligne de texte dans le rapport de résultats de comparaison de contenu XML. Si le contenu XML d'entrée comprend de longues lignes de texte, cette propriété permet de consommer moins de mémoire pour le rapport de résultats de la comparaison de contenu XML.

Valeur par défaut : 80

lisa.graphical.xml.diff.report.max.numberoflines

Cette propriété configure le nombre maximum de lignes de contexte pour une différence dans le rapport de résultats de la comparaison de contenu XML. Si le contenu XML qui est comparé comporte différents longs éléments, cette propriété permet de consommer moins de mémoire pour le rapport de résultats de comparaison de contenu XML.

Par défaut : 5

Remarque : Dans la plupart des cas, ces valeurs ne doivent pas être modifiées. Modifier ces valeurs par défaut pour ces propriétés peut supposer une plus grande consommation d'UC, une mémoire insuffisante ou les deux pour le moteur de comparaison de contenu XML graphique.

Simulation de test rapide

DevTest Workstation vous permet d'exécuter un scénario de test rapide avec une configuration minimale. Un test rapide consiste à exécuter le test qui se trouve en mémoire, plutôt que de charger un document de scénario de test. Il utilise une configuration de simulation prédéfinie simple comportant peu d'options. Le test a un nombre minimal d'instances. Il est donc facile de le simuler ou de l'exécuter, mais il n'offre pas la même fonctionnalité qu'un test simulé à l'aide d'un document de simulation (page 235), comme dans le cas d'un scénario de test complet (page 316). Un test rapide vous permet de sélectionner et de surveiller des événements et des mesures, ainsi que d'afficher un rapport de performances standard.

Procédez comme suit:

1. Ouvrez un scénario de test dans DevTest Workstation.
2. Dans le menu principal, sélectionnez Actions, Stage a Quick Test (Simuler un test rapide).

   La boîte de dialogue Stage a Quick Test s'ouvre.

   Si des coordinateurs et des simulateurs sont associés à un registre, les serveurs de coordination ou de simulation associés sont affichés.
3. Spécifiez les paramètres suivants :

   Run Name (Nom de l'exécution)

   Nom du test rapide.

   Number of Instances (Nombre d'instances)

   Nombre d'utilisateurs simultanés (instances) à utiliser. Votre licence détermine le nombre maximum d'utilisateurs que vous pouvez spécifier.

   Stage Instances To (Simuler les instances sur)

   Nom du serveur de coordination sur lequel le test est simulé.

   If test ends, restart it (Si le test se termine, redémarrez-le.)

   Cochez cette case pour exécuter le test en continu jusqu'à l'arrêt manuel.
4. Cliquez sur OK.

   La fenêtre Test Monitor (page 307) (Moniteur de tests) s'ouvre.

Fenêtre Test Monitor (Moniteur de tests) pour un test rapide

Lorsque vous simulez un test rapide (page 306), la fenêtre Test Monitor (Moniteur de tests) s'ouvre dans DevTest Workstation. La fenêtre Test Monitor pour un test rapide est similaire à la fenêtre Test Monitor d'un scénario de test.

Le test est d'abord simulé, mais il n'est pas encore exécuté.

Vous pouvez sélectionner les mesures et les événements à surveiller lors de l'exécution du test.

Le moniteur Test Monitor comprend deux onglets :

- L'onglet Perf Stats (page 308) (Statistiques de performances) affiche les mesures collectées dans une fonction temporelle. Une fois que le test démarre, vous ne pouvez pas modifier cet affichage.
- L'onglet Events (page 311) (Evénements) affiche les événements enregistrés pendant l'exécution du test.

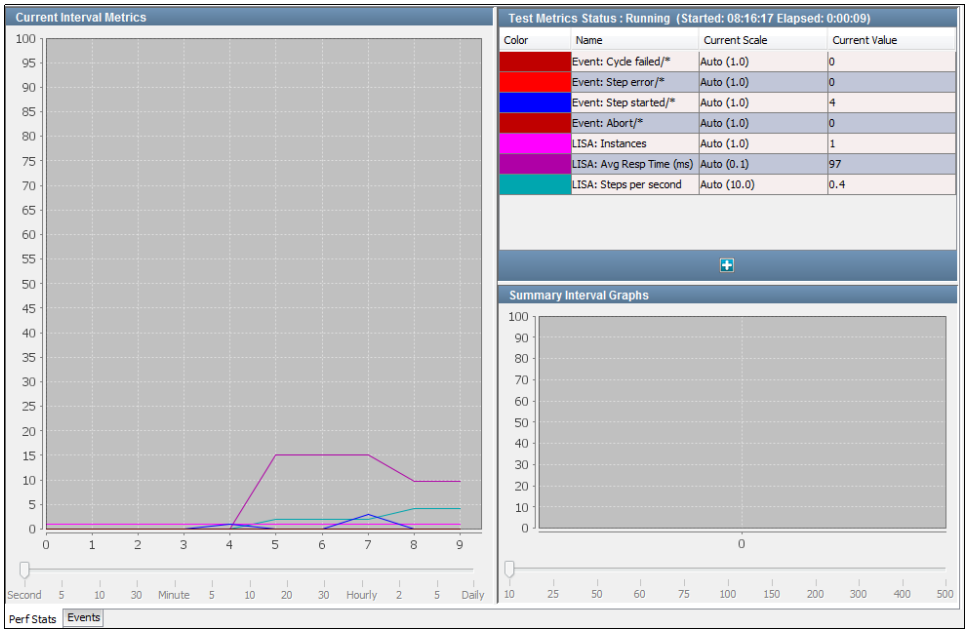
Si vous exécutez le scénario de test et que des échecs se produisent dans le test, un troisième onglet, Failures (page 313) (Echecs) s'affiche.

Si vous exécutez un scénario de test de référence, l'onglet Consolidated Baseline (Référence consolidée) est également affiché. Pour plus d'informations, reportez-vous à la rubrique *Utilisation de CA Continuous Application Insight.*

Vous pouvez limiter la taille des événements d'échec en ajoutant la propriété **lisa.coord.failure.list.size** au fichier **local.properties.**

Onglet Perf Stats (Statistiques de performances) pour un test rapide

L'onglet Perf Stats (Statistiques de performances) vous permet d'ajouter les mesures et les événements que vous voulez surveiller pendant l'exécution du test.



L'onglet Perf Stats comprend les panneaux suivants :

- Test Metrics Status (Statut des mesures de test)
- Current Interval Metrics (Mesures de l'intervalle actuel)
- Summary Interval Graphs (Graphiques d'intervalle récapitulatifs)

Test Metrics Status (Statut des mesures de test)

Le panneau Test Metrics Status (Statut des mesures de test) répertorie les mesures surveillées actuellement.

Certaines mesures sont affichées par défaut. Les mesures obéissent à un code couleur afin de faciliter leur distinction.

Vous pouvez ajouter d'autres mesures en cliquant sur Add (Ajouter)  . Un menu déroulant affiche toutes les catégories de mesures que vous pouvez ajouter.

Le panneau Test Metrics Status contient les colonnes suivantes :

Color (Couleur)

Code couleur utilisé dans les graphiques.

Name (Nom)

Nom de la mesure. Si la mesure a été filtrée à l'aide de son nom abrégé, celui-ci s'affiche après une barre oblique dans le champ Name (Nom). Un astérisque indique que seul le nom de l'événement doit être utilisé pour la mesure.

Current Scale (Echelle actuelle)

Echelle verticale utilisée dans les graphiques. Vous pouvez ajuster l'échelle du graphique dans le graphique Current Interval Metrics (Mesures de l'intervalle actuel) pour une mesure. Pour ajuster l'échelle, double-cliquez sur la cellule Current Scale (Echelle actuelle) et sélectionnez une nouvelle valeur dans la liste déroulante. L'échelle de la mesure est mise à jour dans le graphique Current Interval Metrics (Mesures de l'intervalle actuel).

Remarque : Le paramètre par défaut pour l'option Current Scale est un ajustement automatique de l'échelle à 100. Le nombre entre parenthèses indique la valeur requise par la mise à l'échelle automatique pour ajuster toutes les données sur le même graphique 0 - 100. Dans le graphique : valeur * échelle = coordonnée y. Si vous modifiez l'échelle, la valeur actuelle ne change pas. Toutefois, le calcul permettant de déterminer la coordonnée y utilisée dans le graphique peut renvoyer une coordonnée y différente. Si la coordonnée est supérieure à 100, les limites de l'axe Y doivent être étendues, afin de prendre en compte de plus grandes valeurs.

Current Value (Valeur actuelle)

Valeur instantanée de la mesure.

Pour des informations détaillées sur les mesures, consultez la section Génération de mesures (page 265) de la rubrique *Utilisation de CA Application Test.*

Current Interval Metrics (Mesures de l'intervalle actuel)

Pour chaque mesure figurant dans le panneau Test Metrics Status (Statut des mesures de test), le panneau Current Interval Metrics (Mesures de l'intervalle actuel) affiche la valeur à un intervalle spécifié pendant l'exécution.

Pour spécifier l'intervalle, utilisez le curseur au bas du panneau. Vous ne pouvez pas ajuster la valeur une fois que l'exécution a démarré.

Summary Interval Graphs (Graphiques d'intervalle récapitulatifs)

Le panneau Summary Interval Graphs affiche la moyenne de chaque mesure figurant dans le panneau Test Metrics Status (Statut des mesures de test) pour un intervalle spécifié.

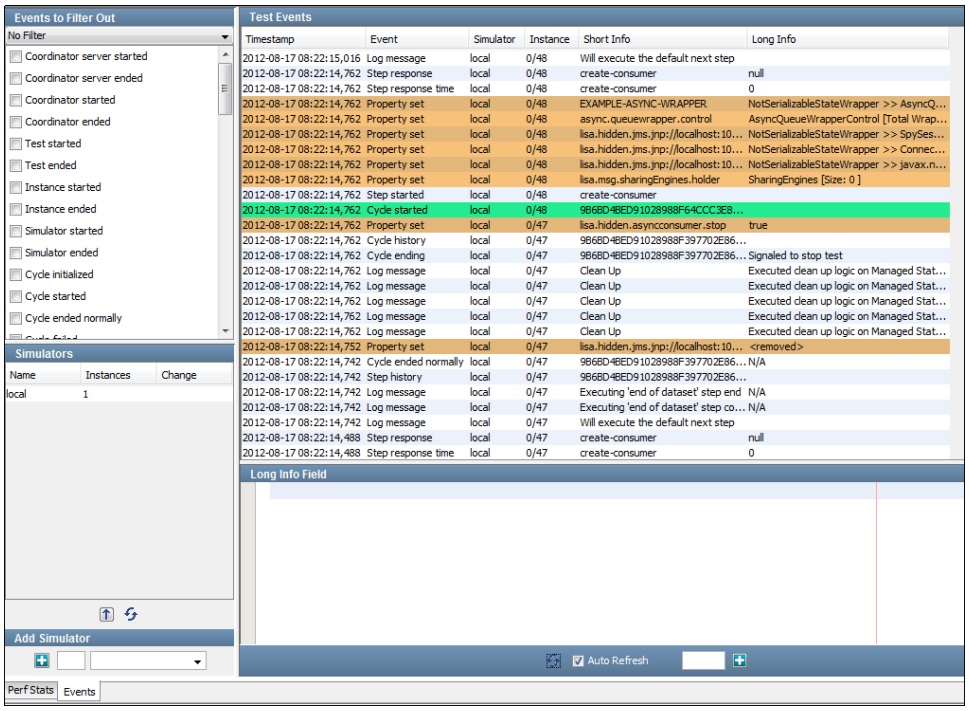
Pour spécifier le nombre d'exemples par intervalle, utilisez le curseur au bas du panneau. Vous ne pouvez pas ajuster la valeur une fois que l'exécution a démarré.

Onglet Events (Evénements) pour un test rapide

L'onglet Events (Evénements) affiche les événements générés pendant l'exécution.

L'onglet Events comprend les panneaux suivants :

- Events to Filter Out (Evénements à filtrer)
- Simulators (Simulateurs)
- Test Events (Evénements de test)



Events to Filter Out (Evénements à filtrer)

Le panneau Events to Filter Out vous permet de limiter les événements qui s'affichent pendant l'exécution. Il contient une liste d'événements. Si la case à cocher pour un événement est sélectionnée, l'événement ne s'affiche pas pendant l'exécution.

Le panneau inclut également une liste déroulante. Les options vous permettent de ne filtrer aucun des événements, d'inclure un ensemble prédéfini d'événements ou de filtrer tous les événements. Les options sont les suivantes :

- No Filter (Aucun filtre)
- Terse Event Set (Ensemble d'événements courts)
- Common Event Set (Ensemble d'événements communs)

- Verbose Event Set (Ensemble d'événements détaillés)
- Load Test Set (Charger un ensemble de tests)
- Custom Event Set (Ensemble d'événements personnalisés)
- Filter All Events (Filtrer tous les événements)

Remarque : Lors d'un test de charge, ces éléments d'interface utilisateur sont désactivés. Dans le cas d'un scénario de test défini par DevTest comme test de charge, vous ne pouvez pas changer le test pour envoyer, par exemple, des réponses d'étape spécifiques. L'envoi d'événements autres que de base rend la valeur du test de charge négative.

Simulators (Simulateurs)

Le panneau Simulators affiche le statut des simulateurs en cours d'utilisation.

Test Events (Evénements de test)

Le panneau Test Events affiche les événements. L'événement le plus récent s'affiche en haut et le plus ancien en bas.

Vous pouvez répertorier les événements en temps réel en cochant la case Auto Refresh (Actualisation automatique) en bas du panneau ou actualiser la liste manuellement en cliquant sur l'icône Refresh (Actualiser) si la case Auto Refresh est décochée. Seuls les événements qui n'ont pas été filtrés sont affichés.

Les informations suivantes sont affichées pour chaque événement :

Timestamp (Horodatage)

Heure de l'événement.

Event (Evénement)

Nom de l'événement.

Simulator (Simulateur)

Nom du simulateur dans lequel l'événement est généré.

Instance

ID de l'instance et numéro de l'exécution (valeurs séparées par une barre oblique).

Short Info (Informations brèves)

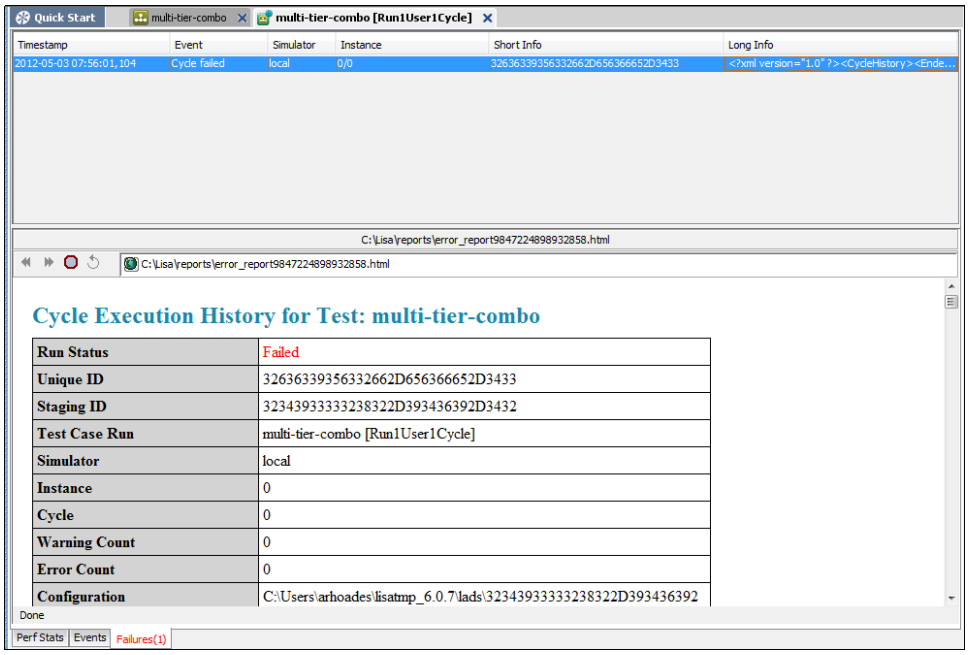
Informations brèves sur l'événement.

Long Info (Informations détaillées)

Informations détaillées sur l'événement, le cas échéant.

Onglet Failures (Echecs) pour un test rapide

S'il y a une erreur dans l'exécution du test simulé, un onglet Failures (Echecs) s'affiche en bas de la fenêtre.



Pour afficher le rapport sur un test ayant échoué, double-cliquez sur le champ Long Info (Informations détaillées) sur le côté droit de la fenêtre.

Démarrage et arrêt de tests rapides

Lorsque la fenêtre Test Monitor (Moniteur de tests) est ouverte, elle vous permet de démarrer et d'arrêter un test rapide à l'aide des icônes de la barre d'outils principale.

Pour démarrer un test rapide, cliquez sur Play (Lire)  dans la barre d'outils principale.

Lorsque le test rapide démarre, l'icône Play est remplacée par une icône Stop (Arrêter).

Pour arrêter un test rapide, cliquez sur Stop  dans la barre d'outils principale.

Si vous cliquez sur Stop (Arrêter) à nouveau, vous êtes invité à confirmer si vous voulez arrêter l'exécution du test.

- Lorsque vous arrêtez un test, vous indiquez que vous ne voulez pas démarrer de nouvelles instances. Aucune nouvelle instance n'est démarrée et un scénario de test en cours d'exécution ne reprend pas au début. Les tests en cours d'exécution effectuent toutes les étapes, puis continue à l'étape de fin.
- Lorsque vous éliminez un test, vous indiquez que vous voulez arrêter toutes les instances dans les plus brefs délais. Aucune nouvelle instance n'est démarrée et un scénario de test en cours d'exécution ne continue pas à l'étape suivante. Aucune étape de fin n'est exécutée. Les rapports indiquent que le test est toujours en cours d'exécution, car le test ne passe jamais par une étape de fin.

Pour relire un test rapide, cliquez sur Replay (Relire)  dans la barre d'outils principale.

Pour fermer un test rapide, cliquez sur Close (Fermer)  dans la barre d'outils principale.

Menu Actions

Lorsque vous exécutez un test et cliquez sur Play (Lire) pour démarrer le test, les options suivantes sont ajoutées au menu Actions.

- Pause Metrics Collection (Interrompre la collecte de mesures)
- Un-pause Metrics Collection (Reprendre la collecte de mesures)
- Save Recent Metrics to XML Document (Enregistrer les mesures récentes dans un document XML)
- Save Recent Metrics to CSV Document (Enregistrer les mesures récentes dans un document CSV)
- Save Interval Data to CSV Document (Enregistrer les données d'intervalle dans un document CSV)

- Save Interval Data to XML Document (Enregistrer les données d'intervalle dans un document XML)
- Save Recent Events to CSV Document (Enregistrer les événements récents dans un document CSV)
- Stage This Test (Simuler ce test)
- Stop Test (Arrêter le test)
- Restart Test (Reloads Documents) (Relancer le test (recharger les documents))

Simulation d'un scénario de test

Vous pouvez exécuter un scénario de test à partir de DevTest Workstation, en spécifiant la configuration, le document de simulation et le serveur de coordination.

La barre d'outils principale dans DevTest Workstation inclut une icône Lab Status (Statut du laboratoire). Si vous simulez un scénario de test dans un laboratoire cloud, l'icône indique lorsque le laboratoire est provisionné, puis prêt à l'emploi.

Procédez comme suit:

1. Ouvrez un scénario de test dans DevTest Workstation.
2. Dans le menu principal, sélectionnez Actions, Stage Test (Simuler le test).

   La boîte de dialogue Stage Test Case (Simuler le scénario de test) s'ouvre.
3. Sélectionnez la configuration (page 108) dans la liste déroulante Configuration**.**

   Si vous laissez ce champ vide, la configuration par défaut est utilisée.
4. Sélectionnez le document de simulation (page 236) dans la liste déroulante Staging doc (Document de simulation).
5. Sélectionnez le serveur de coordination (page 13) ou, si disponible, un laboratoire cloud (page 365) dans la liste déroulante Coordinator Server (Serveur de coordination).

   Les noms de serveur de coordination incluent le laboratoire associé. Par exemple, Coordinator@Default indique que le laboratoire par défaut est utilisé. Dans le cas de laboratoires cloud, le laboratoire est démarré et le scénario de test est simulé sur le coordinateur du laboratoire.
6. Cliquez sur Stage (Simuler).
   - Si vous avez sélectionné un serveur de coordination, la fenêtre Test Monitor (page 317) (Moniteur de tests) s'ouvre. Vous pouvez sélectionner les mesures et les événements à surveiller et démarrer le scénario de test.
   - Si vous avez sélectionné un laboratoire cloud, un message indique qu'un délai est nécessaire au provisionnement du laboratoire. Vous êtes notifié lorsque le processus est terminé. Cliquez sur OK. Lorsque le message Provisioning Complete (Provisionnement terminé) s'affiche, cliquez sur OK. La fenêtre Test Monitor (page 317) (Moniteur de tests) s'ouvre. Vous pouvez sélectionner les mesures et les événements à surveiller et démarrer le scénario de test.

Fenêtre Test Monitor (Moniteur de tests) pour un scénario de test

Lorsque vous simulez un scénario de test (page 316), la fenêtre Test Monitor (Moniteur de tests) s'ouvre dans DevTest Workstation. La fenêtre Test Monitor pour un scénario de test est similaire à la fenêtre Test Monitor d'un test rapide.

Le test est d'abord simulé, mais il n'est pas encore exécuté.

Vous pouvez sélectionner les mesures et les événements à surveiller lors de l'exécution du test.

Le moniteur Test Monitor comprend deux onglets :

- L'onglet Perf Stats (page 318) (Statistiques de performances) vous permet de sélectionner des mesures et de les afficher dans une fonction temporelle.
- L'onglet Events (page 321) (Evénements) vous permet de sélectionner et d'afficher des événements comme ils se produisent pendant l'exécution du test.

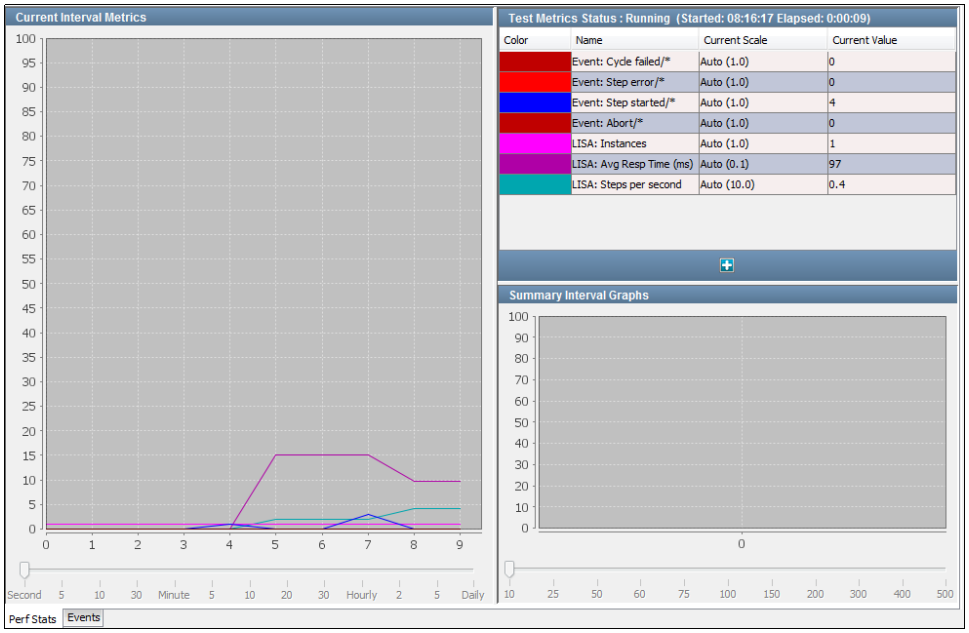
Si vous exécutez le scénario de test et que des échecs se produisent dans le test, un troisième onglet, Failures (page 323) (Echecs) s'affiche.

Si vous exécutez un scénario de test de référence, l'onglet Consolidated Baseline (Référence consolidée) est également affiché. Pour plus d'informations, reportez-vous à la rubrique *Utilisation de CA Continuous Application Insight.*

Vous pouvez limiter la taille des événements d'échec en ajoutant la propriété **lisa.coord.failure.list.size** au fichier **local.properties.**

Onglet Perf Stats (Statistiques de performances) pour un scénario de test

L'onglet Perf Stats (Statistiques de performances) vous permet d'ajouter les mesures et les événements que vous voulez surveiller pendant l'exécution du test.



L'onglet Perf Stats comprend les panneaux suivants :

- Test Metrics Status (Statut des mesures de test)
- Current Interval Metrics (Mesures de l'intervalle actuel)
- Summary Interval Graphs (Graphiques d'intervalle récapitulatifs)

Test Metrics Status (Statut des mesures de test)

Le panneau Test Metrics Status (Statut des mesures de test) répertorie les mesures surveillées actuellement.

Certaines mesures sont affichées par défaut. Les mesures obéissent à un code couleur afin de faciliter leur distinction.

Le panneau Test Metrics Status contient les colonnes suivantes :

Color (Couleur)

Code couleur utilisé dans les graphiques.

Name (Nom)

Nom de la mesure. Si la mesure a été filtrée à l'aide de son nom abrégé, celui-ci s'affiche après une barre oblique dans le champ Name (Nom). Un astérisque indique que seul le nom de l'événement doit être utilisé pour la mesure.

Current Scale (Echelle actuelle)

Echelle verticale utilisée dans les graphiques. Vous pouvez ajuster l'échelle du graphique dans le graphique Current Interval Metrics (Mesures de l'intervalle actuel) pour une mesure. Pour ajuster l'échelle, double-cliquez sur la cellule Current Scale (Echelle actuelle) et sélectionnez une nouvelle valeur dans la liste déroulante. L'échelle de la mesure est mise à jour dans le graphique Current Interval Metrics (Mesures de l'intervalle actuel).

Remarque : Le paramètre par défaut pour l'option Current Scale est un ajustement automatique de l'échelle à 100. Le nombre entre parenthèses indique la valeur requise par la mise à l'échelle automatique pour ajuster toutes les données sur le même graphique 0 - 100. Dans le graphique : valeur * échelle = coordonnée y. Si vous modifiez l'échelle, la valeur actuelle ne change pas. Toutefois, le calcul permettant de déterminer la coordonnée y utilisée dans le graphique peut renvoyer une coordonnée y différente. Si la coordonnée est supérieure à 100, les limites de l'axe Y doivent être étendues, afin de prendre en compte de plus grandes valeurs.

Current Value (Valeur actuelle)

Valeur instantanée de la mesure.

Les mesures sont basées sur des exemples. Par exemple, la mesure d'instances correspond au nombre d'utilisateurs virtuels exécutant un test lorsque l'échantillon est extrait. Le nombre d'instances qui attendent de pouvoir exécuter des tests peut être un nombre plus élevé. Par exemple, si un rythme est configuré dans le document de simulation, la mesure d'instances affiche le nombre d'utilisateurs virtuels exécutant un test. Cette valeur peut être inférieure au nombre d'utilisateurs virtuels alloués dans le document de simulation. Supposez que vous avez 250 utilisateurs virtuels dans un état stable. Si le nombre d'instances est 100, les autres 150 utilisateurs virtuels sont mis en attente, car le rythme défini les bloque.

Pour des informations détaillées sur les mesures, consultez la section Génération de mesures (page 265) de la rubrique *Utilisation de CA Application Test.*

Current Interval Metrics (Mesures de l'intervalle actuel)

Pour chaque mesure figurant dans le panneau Test Metrics Status (Statut des mesures de test), le panneau Current Interval Metrics (Mesures de l'intervalle actuel) affiche la valeur à un intervalle spécifié pendant l'exécution.

Pour spécifier l'intervalle, utilisez le curseur au bas du panneau. Vous ne pouvez pas ajuster la valeur une fois que l'exécution a démarré.

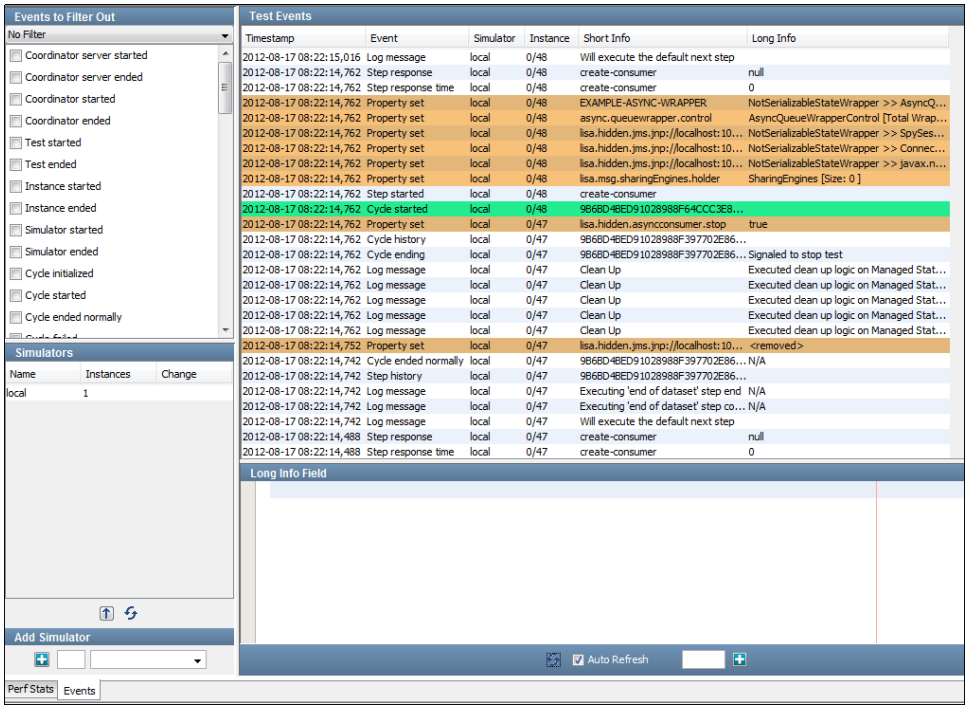
Summary Interval Graphs (Graphiques d'intervalle récapitulatifs)

Le panneau Summary Interval Graphs affiche la moyenne de chaque mesure figurant dans le panneau Test Metrics Status (Statut des mesures de test) pour un intervalle spécifié.

Pour spécifier le nombre d'exemples par intervalle, utilisez le curseur au bas du panneau. Vous ne pouvez pas ajuster la valeur une fois que l'exécution a démarré.

Onglet Events (Evénements) d'un scénario de test

L'onglet Events (Evénements) affiche les événements générés pendant l'exécution.



L'onglet Events comprend les panneaux suivants :

- Events to Filter Out (Evénements à filtrer)
- Simulators (Simulateurs)
- Test Events (Evénements de test)

Events to Filter Out (Evénements à filtrer)

Le panneau Events to Filter Out vous permet de limiter les événements qui s'affichent pendant l'exécution. Il contient une liste d'événements. Si la case à cocher pour un événement est sélectionnée, l'événement ne s'affiche pas pendant l'exécution.

Le panneau inclut également une liste déroulante. Les options vous permettent de ne filtrer aucun des événements, d'inclure un ensemble prédéfini d'événements ou de filtrer tous les événements. Les options sont les suivantes :

- No Filter (Aucun filtre)
- Terse Event Set (Ensemble d'événements courts)
- Common Event Set (Ensemble d'événements communs)

- Verbose Event Set (Ensemble d'événements détaillés)
- Load Test Set (Charger un ensemble de tests)
- Custom Event Set (Ensemble d'événements personnalisés)
- Filter All Events (Filtrer tous les événements)

Remarque : Lors d'un test de charge, ces éléments d'interface utilisateur sont désactivés. Dans le cas d'un scénario de test défini par DevTest comme test de charge, vous ne pouvez pas changer le test pour envoyer, par exemple, des réponses d'étape spécifiques. L'envoi d'événements autres que de base rend la valeur du test de charge négative.

Simulators (Simulateurs)

Le panneau Simulators affiche le statut des simulateurs en cours d'utilisation.

Test Events (Evénements de test)

Le panneau Test Events affiche les événements. L'événement le plus récent s'affiche en haut et le plus ancien en bas.

Vous pouvez répertorier les événements en temps réel en cochant la case Auto Refresh (Actualisation automatique) en bas du panneau ou actualiser la liste manuellement en cliquant sur l'icône Refresh (Actualiser) si la case Auto Refresh est décochée. Seuls les événements qui n'ont pas été filtrés sont affichés.

Les informations suivantes sont affichées pour chaque événement :

Timestamp (Horodatage)

Heure de l'événement.

Event (Evénement)

Nom de l'événement.

Simulator (Simulateur)

Nom du simulateur dans lequel l'événement est généré.

Instance

ID de l'instance et numéro de l'exécution (valeurs séparées par une barre oblique).

Short Info (Informations brèves)

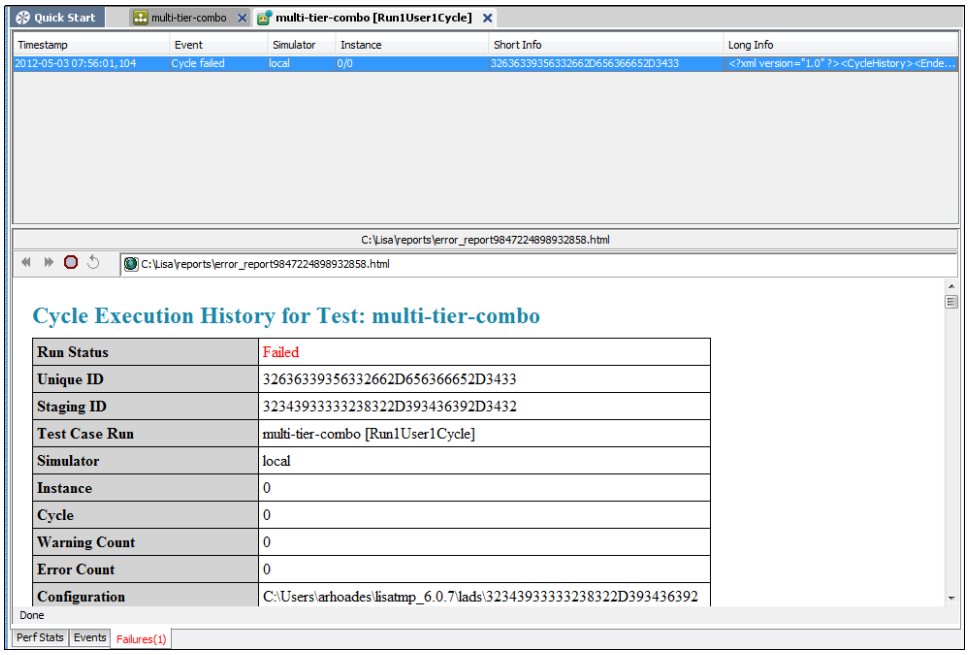
Informations brèves sur l'événement.

Long Info (Informations détaillées)

Informations détaillées sur l'événement, le cas échéant.

Onglet Failures (Echecs) d'un scénario de test

S'il y a une erreur dans l'exécution du test simulé, un onglet Failures (Echecs) s'affiche en bas de la fenêtre.



Pour afficher le rapport sur un test ayant échoué, double-cliquez sur le champ Long Info (Informations détaillées) sur le côté droit de la fenêtre.

Démarrage et arrêt de scénarios de test

Lorsque la fenêtre Test Monitor (Moniteur de tests) est ouverte, elle vous permet de démarrer et d'arrêter un scénario de test à l'aide des icônes de la barre d'outils principale.

Pour démarrer un scénario de test, cliquez sur Play (Lire)  dans la barre d'outils principale.

Lorsque le scénario de test démarre, l'icône Play est remplacée par une icône Stop (Arrêter).

Pour arrêter un scénario de test, cliquez sur Stop  dans la barre d'outils principale.

Si vous cliquez sur Stop (Arrêter) à nouveau, vous êtes invité à confirmer si vous voulez arrêter l'exécution du test.

- Lorsque vous arrêtez un test, vous indiquez que vous ne voulez pas démarrer de nouvelles instances. Aucune nouvelle instance n'est démarrée et un scénario de test en cours d'exécution ne reprend pas au début. Les tests en cours d'exécution effectuent toutes les étapes, puis continue à l'étape de fin.
- Lorsque vous éliminez un test, vous indiquez que vous voulez arrêter toutes les instances dans les plus brefs délais. Aucune nouvelle instance n'est démarrée et un scénario de test en cours d'exécution ne continue pas à l'étape suivante. Aucune étape de fin n'est exécutée. Les rapports indiquent que le test est toujours en cours d'exécution, car le test ne passe jamais par une étape de fin.

Pour relire un scénario de test, cliquez sur Replay (Relire)  dans la barre d'outils principale.

Pour fermer un scénario de test, cliquez sur Close (Fermer)  dans la barre d'outils principale.

Exécution d'une suite de tests

Une suite de tests vous permet de grouper des scénarios de test et des suites de tests associés, et de les exécuter dans un test unique. Un document de suite de tests spécifie le contenu de la suite, les rapports à générer et les mesures à collecter. Ces rapports et ces mesures se rapportent à la suite dans sa totalité. Chaque test de la suite produit toutefois ses propres rapports et mesures. Chaque test conserve la capacité d'utiliser son propre document de simulation, sa configuration et son document d'audit. Par conséquent, vous pouvez exécuter des tests dans une suite dans un environnement distribué.

Lorsqu'une suite est incluse dans une suite, ses tests sont extraits et exécutés en tant que tests individuels dans la suite actuelle. Les valeurs par défaut de la suite incluse et les paramètres de démarrage et de désinstallation sont ignorés.

Dans DevTest Workstation, vous pouvez configurer et simuler la suite de tests, surveiller les tests pendant leur exécution et afficher les rapports à la fin des tests.

Procédez comme suit:

1. Ouvrez la suite de tests dans DevTest Workstation.
2. Dans le menu principal, sélectionnez Actions, Run (Exécuter).

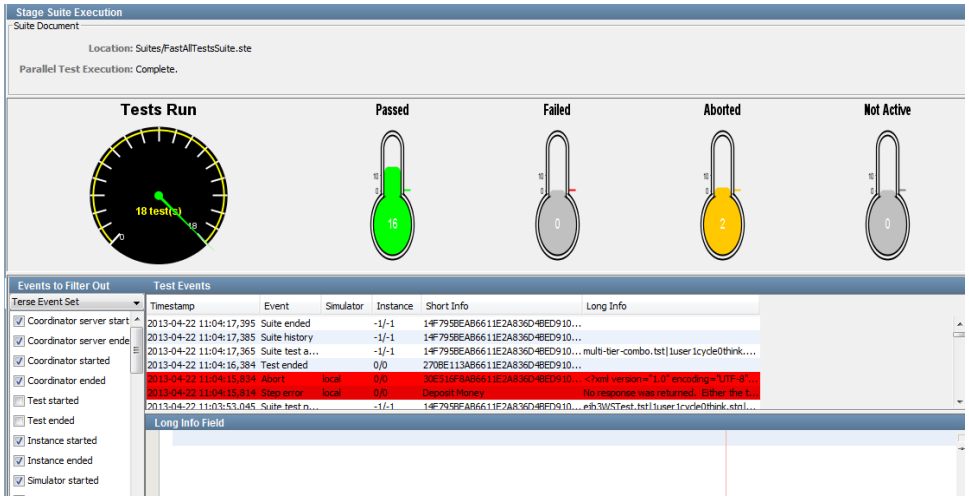
   Cette action ouvre un menu qui vous permet de sélectionner une exécution locale ou une exécution avec un registre spécifié.

   La boîte de dialogue Run Suite Locally (Exécuter la suite localement) s'ouvre.
3. Entrez le nom de la suite de tests.
4. Sélectionnez la configuration dans la liste déroulante.
5. Cliquez sur Stage (Simuler).

La fenêtre Stage Suite Execution (page 326) (Simuler l'exécution de la suite) s'ouvre et les tests démarrent.

Simulation de l'exécution d'une suite

Lorsque vous cliquez sur Stage (Simuler) pendant l'exécution d'une suite de tests (page 325) dans DevTest Workstation, l'onglet Stage Suite Execution (Simuler l'exécution de la suite) s'ouvre et les tests démarrent.



Le panneau supérieur affiche les informations suivantes :

- L'emplacement et le nom du document de suite de tests
- Le nom du test actuel

Le panneau du milieu affiche le compteur Tests Run (Exécutions de test) et quatre thermomètres.

- Ce compteur indique le nombre de tests terminés et le nombre total de tests.
- Les thermomètres ont les étiquettes suivantes : **Passed** (Réussite), **Failed** (Echec), **Aborted** (Interrompu) et **Not Active** (Inactif). Les couleurs des thermomètres indiquent que vous avez atteint le nombre présélectionné de scénarios réussis. Le comportement connu lorsque vous exécutez un nombre de scénarios de test dans une suite compris entre 50 et 100 est le suivant :
  - Si le nombre de tests réussis est supérieur à 50, le thermomètre est orange.
  - Si le nombre de tests réussis est supérieur à 75, le thermomètre est rouge.
  - Si le nombre de tests réussis est supérieur à 100, le thermomètre est gris.

Le panneau inférieur contient les onglets suivants :

- Onglet Events (page 327) (Evénements) : répertorie les événements demandés lorsqu'ils se produisent.
- Onglet Results (page 329) (Résultats) : affiche le statut de chaque test.

Onglet Events (Evénements) de la simulation d'exécution d'une suite

L'onglet Events (Evénements) répertorie les événements que vous avez sélectionnés dans le panneau gauche de l'onglet.

Events to Filter Out (Evénements à filtrer)

La zone Events to Filter Out contient une liste des événements disponibles. Une case à cocher est associée à chaque événement. Pour sélectionner les événements que vous ne voulez pas surveiller, utilisez cette liste.

Une approche consiste à sélectionner les événements à filtrer manuellement. Une autre approche consiste à sélectionner un des ensembles d'événements suivants dans la liste déroulante et à personnaliser les sélections de manière appropriée :

- Terse Event Set (Ensemble d'événements courts)
- Common Event Set (Ensemble d'événements communs)
- Verbose Event Set (Ensemble d'événements détaillés)
- Load Test Set (Charger un ensemble de tests)

Vous pouvez décocher toutes les cases en sélectionnant No Filter (Aucun filtre). Vous pouvez cocher toutes les cases en sélectionnant Filter All Events (Filtrer tous les événements).

Test Events (Evénements de test)

La zone Test Events répertorie les événements lorsqu'ils se produisent, le plus récent se trouvant au sommet de la liste. Vous pouvez répertorier les événements en temps réel en cochant la case Auto Refresh (Actualisation automatique) en bas du panneau. Vous pouvez actualiser la liste manuellement en cliquant sur l'icône Refresh (Actualiser) si la case Auto Refresh est décochée.

Les informations suivantes sont affichées pour chaque événement :

Timestamp (Horodatage)

Heure de l'événement.

Event (Evénement)

Nom de l'événement.

Simulator (Simulateur)

Nom du simulateur dans lequel l'événement est généré.

Instance

ID de l'instance et numéro de l'exécution (valeurs séparées par une barre oblique).

Short Info (Informations brèves)

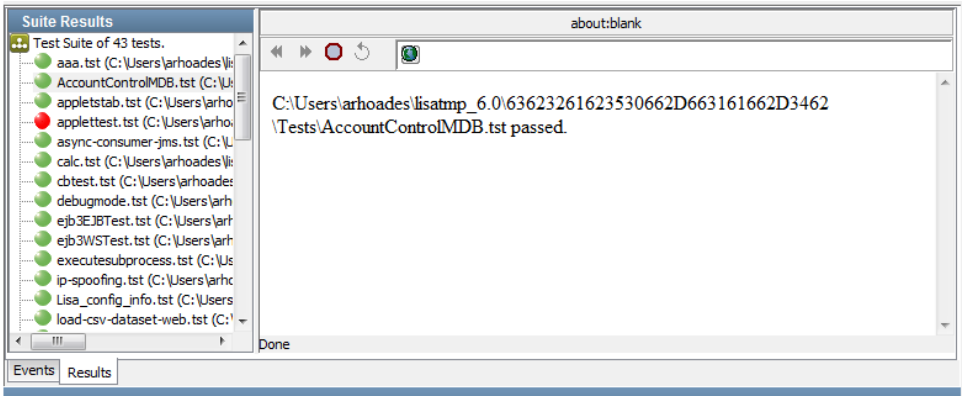
Informations brèves sur l'événement.

Long Info (Informations détaillées)

Informations détaillées sur l'événement, le cas échéant.

Onglet Results (Résultats) de la simulation d'exécution d'une suite

L'onglet Results indique le statut de chaque test de la suite de tests.



Suite Results (Résultats de la suite)

La zone Suite Results affiche une liste de tous les tests de la suite, avec des icônes.

- L'icône verte indique que le test est terminé et réussi.
- L'icône rouge indique que le test est terminé et a échoué.
- L'icône bleue indique que le test est en cours d'exécution.

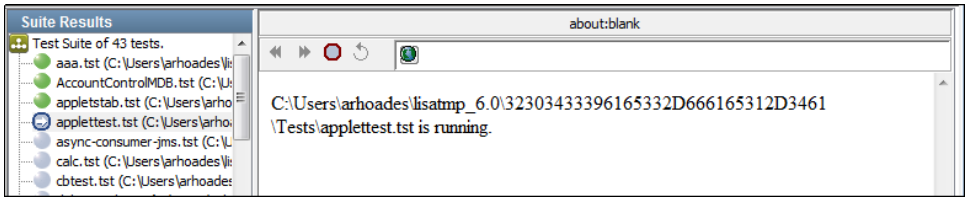
Pour afficher le moniteur de tests pour les tests en cours d'exécution, cliquez sur View Test (Afficher le test)  en bas du panneau.

Pour les tests terminés, vous pouvez sélectionner le nom d'un test pour afficher son statut dans la zone de texte sur la droite. Cela vous permet de déterminer les raisons des échecs de tests.

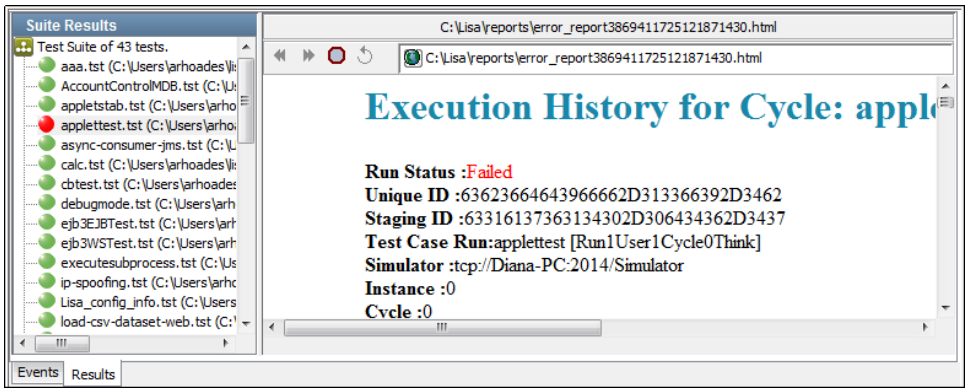
Fenêtre Status (Statut)

La fenêtre Status affiche le statut du test sélectionné à gauche.

Le test suivant est en cours d'exécution :



Le test suivant a échoué :



Un statut d'échec pour le test affiche les informations disponibles sur les raisons de l'échec.

Une barre d'outils est disponible au bas du panneau Stage Suite Execution (Simuler l'exécution de la suite).

Le premier ensemble d'icônes permet de gérer les tests de la suite. Pour utiliser ces icônes, sélectionnez d'abord un test dans la section Test List (Liste de tests) de l'onglet Results (Résultats). Les fonctions suivantes sont disponibles pour un test sélectionné :

	Stop Test (Arrêter le test)	Arrête le test après avoir atteint la fin/l'échec logique suivant, mais ne commence aucun nouveau cycle.
	Kill Test (Eliminer le test)	Arrête le test immédiatement une fois que l'étape actuelle se termine.
	View Test (Afficher le test)	Cet outil fonctionne uniquement pour les tests en cours d'exécution et permet de lancer le moniteur de tests pour le test sélectionné.

Le second ensemble d'icônes permet de gérer la suite de tests ou son exécution :

	Run Suite (Exécuter la suite)	Démarre/redémarre le test de la suite.
	Close Suite (Fermer la suite)	Ferme la fenêtre Stage Suite Execution (Simuler l'exécution de la suite).

	View Test (Afficher le test)	Démarre le moniteur de tests pour le test sélectionné.

Pour plus d'informations sur le moniteur de tests, consultez la rubrique Utilisation du moniteur de registre.

Utilisation de l'optimiseur de test de charge

L'optimiseur Load Test Optimizer (Optimiseur de test de charge) est inclus dans le Registry Monitor (Moniteur de registre) et permet d'exécuter un test de charge simple sur le système testé. Un test de charge détermine le nombre d'utilisateurs pris en charge par le système testé.

Dans l'optimiseur de test de charge, le système testé s'exécute en continu. D'autres utilisateurs simulés continuent d'y accéder, jusqu'à ce qu'un seuil soit atteint. Ce seuil est généralement un temps de réponse moyen prédéfini.

L'optimiseur de test de charge permet de déterminer le nombre d'utilisateurs pris en charge par le système testé. Vous pouvez définir un optimiseur pour une mesure DevTest, comme le temps de réponse moyen. Vous pouvez également définir un optimiseur pour une mesure provenant d'une source externe, comme une source SNMP, JMX ou Windows PerfMon. Le nombre d'utilisateurs augmente selon une fréquence prédéfinie et vous informe lorsqu'un seuil de mesure prédéfini est atteint. Par exemple, vous pouvez augmenter le nombre d'instances de test de 5 instances toutes les 10 secondes jusqu'à ce que le temps de réponse moyen atteigne 2 secondes.

Pour pouvoir optimiser un test, il doit être en cours d'exécution.

Pour configurer et démarrer un optimiseur, procédez comme suit :

1. Dans le moniteur de registre, sélectionnez un test dans la liste des tests en cours d'exécution et cliquez sur Optimize Test (Optimiser le test)  .

   Le panneau Optimizers (Optimiseurs) s'ouvre avec le nom du document de simulation dans la partie supérieure.

2. Configurez les paramètres suivants :

   Metric (Mesure)

   Mesure à utiliser pour l'optimisation. Sélectionnez un simulateur dans la liste déroulante.

   Simulator (Simulateur)

   Simulateur utilisé pour générer les instances de test. Sélectionnez un simulateur dans la liste déroulante.

   Threshold Low (Seuil inférieur)

   Valeur de mesure à partir de laquelle l'optimiseur signale que le système requiert des utilisateurs virtuels supplémentaires. Entrez une valeur numérique.

   Threshold High (Seuil supérieur)

   Valeur de mesure à partir de laquelle l'optimiseur signale que le système ne peut plus prendre en charge des utilisateurs virtuels supplémentaires. Entrez une valeur numérique.

Increment (#instances) (Incrément (nombre d'instances))

Nombre d'utilisateurs virtuels supplémentaires à ajouter au système selon la fréquence de mise à jour. Entrez une valeur numérique.

Update Frequency (millis) (Fréquence de mise à jour (en millisecondes))

Nombre de millisecondes entre chaque augmentation.

3. Cliquez sur Start Optimizer (Démarrer l'optimiseur)  .

   Lorsque l'optimiseur augmente le nombre d'utilisateurs virtuels, ce nombre s'affiche dans la section Optimizers (Optimiseurs) et dans la colonne Instances du laboratoire Tests, dans le moniteur de registre.

   Lorsque l'optimiseur s'exécute, vous pouvez changer les paramètres. Pour mettre à jour l'optimiseur avec les informations modifiées, cliquez sur Update (Mettre à jour)  .

4. Pour fermer l'optimiseur, cliquez sur Close (Fermer)  .

Exécution d'un scénario de test d'intégration de Selenium

Les étapes du test d'intégration de Selenium vous permettent d'importer des scripts de test pour des interfaces utilisateur Web à partir de Selenium Builder dans DevTest Solutions. Pour enregistrer ces scripts de test, vous devez disposer de Selenium Builder, qui est uniquement pris en charge dans Firefox. Après avoir importé le test dans DevTest, vous pouvez l'exécuter dans Mozilla FireFox, Google Chrome, Internet Explorer 8.0 ou version ultérieure. Vous pouvez également exécuter le test dans un navigateur local ou distant.

Vous pouvez exécuter les tests d'intégration de Selenium comme les autres scénarios de test dans DevTest. Toutefois, pour pouvoir exécuter ces tests dans un navigateur autre que Firefox, vous devez effectuer des tâches supplémentaires. Pour obtenir des informations générales sur l'exécution d'un scénario de test, consultez la rubrique Exécution de scénarios et de suites (page 291).

Exécution d'un test d'intégration de Selenium sur Google Chrome (local)

Cette rubrique décrit la procédure d'exécution d'un scénario de test d'intégration de Selenium sur Google Chrome sur votre ordinateur local.

Procédez comme suit:

1. Téléchargez le pilote Selenium pour Chrome et enregistrez-le dans un répertoire local.
   a. Accédez au site http://www.seleniumhq.org/download/.
   b. Recherchez le pilote Chrome dans la section **Third Party Browser Drivers NOT DEVELOPED by seleniumhq** et téléchargez-le.
2. Ajoutez les propriétés suivantes au fichier de configuration de projet que vous utilisez pour exécuter des scénarios de test sur Chrome.
   - **Key (Clé) :** selenium.broswer.type

     Value (Valeur) : Chrome
   - **Key (Clé) :** selenium.chrome.driver.path

     Value (Valeur) : chemin d'accès complet au pilote Chrome téléchargé. Par exemple : C:\lisa-se\chromedriver.exe.

   Remarque : Ajouter ces propriétés au fichier project.config avant de les ajouter à d'autres fichiers de configuration de projet.
3. Cliquez avec le bouton droit de la souris sur le fichier de configuration de projet sélectionné dans le panneau Project (Projet) et sélectionnez Make Active (Activer).
4. Exécutez le test.

Exécution d'un test d'intégration de Selenium sur Microsoft Internet Explorer (local)

Cette rubrique décrit la procédure d'exécution d'un scénario de test d'intégration de Selenium sur Internet Explorer sur votre ordinateur local.

Procédez comme suit:

1. Téléchargez le pilote Selenium 32 bit Windows IE et enregistrez-le dans un répertoire local.
   a. Accédez au site http://www.seleniumhq.org/download/.
   b. Recherchez le pilote 32 bit Windows IE dans la section **Internet Explorer Driver Server** et téléchargez-le.

   Remarque : En raison d'un problème connu lié aux performances du pilote 64 bits, il est recommandé d'installer la version 32 bits, même si vous utilisez un système 64 bits.
2. Ajoutez les propriétés suivantes au fichier de configuration de projet que vous utilisez pour exécuter des scénarios de test sur Internet Explorer.
   - **Key (Clé) :** selenium.broswer.type

     Value (Valeur) : IE
   - **Key (Clé) :** selenium.ie.driver.path

     Value (Valeur) : chemin d'accès complet au pilote Internet Explorer téléchargé. Par exemple : C:\lisa-se\IEDriverServer.exe.

   Remarque : Ajouter ces propriétés au fichier project.config avant de les ajouter à d'autres fichiers de configuration de projet.
3. Cliquez avec le bouton droit de la souris sur le fichier de configuration de projet sélectionné dans le panneau Project (Projet) et sélectionnez Make Active (Activer).
4. Modifiez vos paramètres de sécurité Internet Explorer.
   a. Ouvrez Internet Explorer.
   b. Sélectionnez Outils, Options Internet.
   c. Cliquez sur l'onglet Security (Sécurité).
   d. Vérifiez que la case à cocher Enable Protected Mode (Activer le mode protégé) est sélectionnée ou desélectionnée de manière cohérente avec les zones suivantes. Si ce paramètre est incohérent, Selenium ne démarrera pas.
      - Internet
      - Local Intranet (Intranet local)
      - Trusted Sites (Sites approuvés)
      - Restricted Sites (Sites sensibles)
   e. Cliquez sur OK pour enregistrer les modifications et fermer la fenêtre Internet options (Options Internet).

5. Exécutez le test.

Exécution d'un test d'intégration de Selenium sur un navigateur distant

Cette rubrique décrit la procédure d'exécution d'un scénario de test d'intégration de Selenium sur un navigateur distant. Le navigateur distant peut être Mozilla Firefox, Google Chrome ou Internet Explorer 8.0 ou version ultérieure.

Procédez comme suit:

1. Téléchargez le fichier du serveur Selenium Server et enregistrez-le dans un répertoire local.
   a. Accédez au site http://www.seleniumhq.org/download/.
   b. Localisez le fichier JAR autonome de Selenium Server dans la section **Selenium Server (formerly the Selenium RC Server)** et téléchargez-le.
2. Vérifiez que le pilote pour le navigateur que vous voulez utiliser est disponible sur l'ordinateur distant.
3. Sur l'ordinateur distant, exécutez la commande suivante dans une invite de commande :
   ```
   java -jar selenium-server-standalone-2.xx.0.jar -role hub
   ```
4. Sur l'ordinateur distant, exécutez la commande suivante dans une nouvelle invite de commande :
   ```
   java -jar selenium-server-standalone-2.xx.0.jar -role node -hub
   http://localhost:4444/grid/register
   -Dwebdriver.chrome.driver=c:\lisa-se\chromedriver.exe
   -Dwebdriver.ie.driver=c:\lisa-se\IEDriverServer.exe
   ```
5. Sur l'ordinateur local, ajoutez les propriétés suivantes au fichier de configuration de projet que vous utilisez pour les scénarios de test en cours d'exécution sur le navigateur distant.
   - **Key (Clé) :** selenium.broswer.type

     Values (Valeurs) : IE, Firefox ou Chrome
   - **Key (Clé) :** selenium.remote.url

     Value (Valeur) : URL du concentrateur Selenium Server distant. Par exemple, **http://nom_hôte_distant:4444/wd/hub**.

   Si vous voulez exécuter des tests d'intégration de Selenium sur plusieurs navigateurs, créez un fichier de configuration de projet pour chaque type de navigateur. Vous pouvez alors exécuter les tests sur plusieurs navigateurs en activant les différents fichiers de configuration actifs pour chaque test. Pour plus d'informations sur les fichiers de configuration, consultez la rubrique Configurations (page 108).

   Remarque : Vous devez ajouter ces propriétés au fichier project.config avant de les ajouter à d'autres fichiers de configuration de projet.
6. Cliquez avec le bouton droit de la souris sur le fichier de configuration de projet sélectionné dans le panneau Project (Projet) et sélectionnez Make Active (Activer).

7. Exécutez le test.

   Le test est exécuté sur le navigateur sélectionné sur l'ordinateur distant.

   Remarque : Pour plus d'informations sur l'utilisation de Selenium Server dans une configuration de grille, reportez-vous au site https://code.google.com/p/selenium/wiki/Grid2.

Supposition de test de charge

Par défaut, DevTest examine différentes caractéristiques d'un scénario de test ou d'une suite lors de l'exécution et peut conclure que vous exécutez un test de charge. Si DevTest détermine que vous exécutez un test de charge, une reconfiguration automatique sera effectuée.

Vous pouvez modifier la configuration automatique en définissant la propriété **lisa.load.auto.reconfigure** sur false.

La reconfiguration pour l'analyse de charge entraîne plusieurs modifications. La modification la plus importante consiste à limiter les événements envoyés à partir des simulateurs exécutant le test. Les simulateurs envoient uniquement les événements figurant dans l'ensemble de filtres LoadTest. Les événements StepStarted (Etape lancée), Step response (Réponse d'étape), Step response time (Temps de réponse d'étape), Cycle started (Cycle lancé), Log (Journal), Profile et d'autres événements similaires ne sont pas envoyés au coordinateur, ni au composant DevTest qui a démarré le test (DevTest Workstation ou l'utilitaire Test Runner (Exécuteur de tests)).

La raison à cela est qu'avec les tests de charge, la surcharge liée à l'envoi des événements dépasse rapidement la création de charge. Le coordinateur devient donc rapidement surchargé, particulièrement avec les tests qui n'ont pas de délai de réflexion.

Si DevTest suppose que vous exécutez un test de charge, des messages similaires aux suivants s'afficheront dans le fichier journal du coordinateur :

```
INFO com.itko.lisa.coordinator.CoordinatorImpl - Configuring for
load test (vusers >= 150) (Configuration du test de charge,
utilisateurs virtuels : 150)
INFO com.itko.lisa.coordinator.CoordinatorImpl - Configuring for
load test
INFO  com.itko.lisa.net.RemoteEventDeliverySupport - configuring
for load test
```

Un message supplémentaire peut s'afficher vous invitant à désactiver CAI dans le document de simulation.

Si votre test fait partie d'une suite, la fenêtre Test Monitor (Moniteur de tests) ne fournit en général aucune information. Comme les tests de charge peuvent générer des événements plus rapidement que l'interface utilisateur peut les traiter, certains événements sont ignorés. Pour afficher ces mesures d'exécution, modifiez la propriété suivante :

```
lisa.load.auto.reconfigure=false
```

Pour ajuster les paramètres qui permettent à CA Application Test de supposer qu'il s'agit d'un test de charge, modifiez les propriétés suivantes :

```
lisa.loadtest.aggressive.detection=false
lisa.coordinator.step.per.sec.load.threshold=100
```

```
lisa.coordinator.vuser.load.threshold=150
```

Si un document de simulation contient un Default Report Generator (Générateur de rapports par défaut), il sera admis qu'il ne s'agit pas d'un test de charge.

Si le nombre d'étapes non silencieuses par seconde dépasse 100 ou que le nombre d'utilisateurs virtuels dépasse 150, il sera admis qu'il s'agit d'un test de charge.

Si vous définissez la propriété **lisa.load.auto.reconfigure** sur true, un test comprenant un document de simulation dont le délai de réflexion est de 0 %, ou un test comprenant uniquement des étapes avec un délai de réflexion de 0 sera considéré comme un test de charge.

D'autres résultats de la supposition de test de charge sont les suivants :

- La console Server Console (Console de serveur) affiche un astérisque à côté d'un nom de test.
- La console Reporting Console (Console de génération de rapports) n'affiche pas la quantité normale de détails.
- Si le test de charge a été activé automatiquement, un événement sera généré pour informer de la modification.

Utilitaire Test Runner (Exécuteur de tests)

L'utilitaire de ligne de commande Test Runner (Exécuteur de tests) est une version de DevTest Workstation sans périphérique de contrôle, proposant la même fonctionnalité, mais sans aucune interface utilisateur. Vous pouvez donc l'exécuter comme une application autonome.

L'exécuteur de tests vous permet d'exécuter des tests en tant qu'applications par lot. Vous abandonnez l'opportunité de surveiller les tests en temps réel, mais vous pouvez toujours demander les rapports pour les consulter ultérieurement.

- Sous Windows, l'utilitaire Test Runner (Exécuteur de tests) est disponible dans le répertoire **LISA_HOME\bin** sous la forme d'un fichier exécutable Windows (**TestRunner.exe**).
- Sur UNIX, l'utilitaire Test Runner est disponible sous la forme d'un exécutable UNIX (**TestRunner**) et d'un script UNIX (**TestRunner.sh**).

L'utilitaire Test Runner vous permet d'inclure des tests DevTest dans un flux de travaux de compilation continu. Vous pouvez également utiliser Test Runner avec JUnit pour exécuter des tests JUnit standard dans Ant ou un autre outil de compilation.

L'utilitaire Test Runner fournit les options suivantes :

```
TestRunner [-h] [[-r StagingDocument] [-t TestCaseDocument] [-cs
CoordinatorServerName]] | [-s TestSuiteDocument] [-m TestRegistryName] [-a]
[-config configurationFileName]
```

Pour afficher les informations d'aide pour l'utilitaire Test Runner, utilisez l'option **-h** ou **--help.**

```
TestRunner -h
```

Pour afficher le numéro de version, utilisez l'option **--version.**

Utilisation dans une compilation automatisée

Vous pouvez inclure des scénarios de test DevTest dans un processus de test et de compilation automatique. DevTest fournit le logiciel supplémentaire requis pour l'utilisation de Java Unit et de Java Ant, ainsi qu'un exemple de script de compilation Ant.

Les scénarios de test de DevTest sont exécutés et signalés comme des tests JUnit natifs.

Vous pouvez utiliser l'utilitaire Test Runner (Exécuteur de tests) dans les tests JUnit standard à l'aide d'une classe Java personnalisée. Pour plus d'informations, consultez la section DevTest Solutions avec Ant et JUnit de la rubrique *Administration.*

Cette section comprend les rubriques suivantes :

Exécution d'un fichier MAR avec l'utilitaire Test Runner (Exécuteur de tests)

Pour exécuter une archive de modèle (MAR) (page 278) avec l'utilitaire Test Runner (Exécuteur de tests), spécifiez l'option suivante :

- **-mar** ou **--mar** nom_fichier_MAR

Exemple

Dans les exemples suivants, un fichier MAR nommé **test1.mar** est exécuté.

```
TestRunner -mar C:\test1.mar
```

Exécution d'un scénario de test avec l'utilitaire Test Runner (Exécuteur de tests)

Pour exécuter un scénario de test unique avec l'utilitaire Test Runner (Exécuteur de tests), spécifiez les options suivantes :

- **-t** ou --**testCase** nom_document_scénario_test
- **-r** ou --**stagingDoc** nom_document_simulation

Si vous voulez effectuer une simulation à distance, spécifiez également les options suivantes :

- **-cs** ou --**coordinatorService** nom_serveur_coordination

  Si aucune option **-cs** n'est fournie pour désigner un serveur de coordination à récupérer à partir du registre, un coordinateur spécifié dans le fichier MAR/MARI est utilisé. Si aucun coordinateur n'est spécifié, le coordinateur par défaut spécifié dans le registre est utilisé. S'il n'y a aucun coordinateur par défaut, le test est exécuté localement. Si l'option -cs local est utilisée, le test est exécuté localement.
- **-m** ou **--testRegistry** nom_registre

Pour plus d'options, consultez la rubrique Autres options de l'utilitaire Test Runner (Exécuteur de tests) (page 345).

Exemple

Dans l'exemple suivant, le scénario de test à plusieurs niveaux qui se trouve dans le projet **Examples** (Exemples) est exécuté.

```
TestRunner -t ../examples/Tests/multi-tier-combo.tst -r
../examples/StagingDocs/Run1User1Cycle.stg -a
```

Exécution d'une suite avec l'utilitaire Test Runner (Exécuteur de tests)

Pour exécuter une suite avec l'utilitaire Test Runner (Exécuteur de tests), spécifiez l'option suivante :

- **-s** ou **--testSuite** nom_document_suite

Aucun audit n'est effectué.

Pour effectuer une simulation à distance, spécifiez également l'option suivante :

- **-m** ou **--testRegistry** nom_registre

Les documents de projet de simulation à distance doivent avoir un nom unique pour tous les projets. Cette restriction s'applique non seulement aux suites de projet, mais aussi à la simulation à distance de scénarios de test de projet qui font référence à d'autres actifs (comme des ensembles de données) du projet.

Pour plus d'options, consultez la rubrique Autres options de l'utilitaire Test Runner (Exécuteur de tests) (page 345).

Exemple

Dans l'exemple suivant, la suite AllTestsSuite du projet **Examples** (Exemples) est exécutée :

```
TestRunner -s ../examples/Suites/AllTestsSuite.ste
```

Dans l'exemple suivant, le registre s'exécute sur un autre ordinateur :

```
TestRunner -s ../examples/Suites/AllTestsSuite.ste -m
somecomputer/Registry
```

Autres options de l'utilitaire Test Runner (Exécuteur de tests)

Cette rubrique décrit les options complémentaires que vous pouvez utiliser lors de l'exécution d'une archive de modèle (MAR), d'un scénario de test ou d'une suite avec l'utilitaire Test Runner (Exécuteur de tests).

Vous pouvez utiliser des propriétés lorsque vous spécifiez des noms de fichier. Toutefois, dans ce contexte, seules les propriétés système et les propriétés définies dans vos fichiers de propriété (**lisa.properties**, **local.properties** et **site.properties**) fonctionnent.

A l'issue des tests, vous pouvez afficher les rapports.

Spécification des informations de sécurité au moyen d'une liste de contrôle d'accès

Les options **-u** *nom_utilisateur* ou **--username**=*nom_utilisateur* et **-p** *mot_passe* ou **--password**=*mot_passe* permettent de transférer des informations de sécurité au moyen d'une liste de contrôle d'accès à DevTest.

Démarrage automatique du test

L'option **-a** ou **--autoStart** permet de démarrer automatiquement le test. Vous ne devez pas donc appuyer sur Entrée une fois que le test a été simulé.

Spécification de la configuration

L'option **-config** ou **--configFile** vous permet de spécifier la configuration à utiliser pour une exécution de test.

L'utilitaire Test Runner (Exécuteur de tests) ne comprend pas les projets DevTest ; vous devez donc fournir le chemin d'accès complet au fichier de configuration. Par exemple :

```
-config \path\to\lisa\home\examples\Configs\project.config
```

Génération d'un rapport HTML

L'option **-html** ou **--htmlReport** vous permet d'indiquer à l'utilitaire Test Runner (Exécuteur de tests) de générer un rapport HTML récapitulatif pour un scénario de test. Vous ne pouvez pas utiliser cette option pour les suites.

Le paramètre suivant cette option doit être un nom de fichier complet. Par exemple :

```
-html \some\directory\MyReport.html
```

Modification de l'intervalle de mise à jour

L'option **-u** ou --**update** vous permet de modifier l'intervalle de mise à jour par défaut de 5 secondes. Cet intervalle fait référence à la fréquence à laquelle l'utilitaire Test Runner (Exécuteur de tests) écrit un message de statut dans le fichier journal.

```
-u 10
```

Instances multiples de l'utilitaire Test Runner (Exécuteur de tests)

Vous pouvez simuler plusieurs instances de l'utilitaire Test Runner (Exécuteur de tests) à partir d'une station de travail unique vers DevTest Server.

512 Mo pour chaque instance

Fichier journal de l'utilitaire Test Runner (Exécuteur de tests)

Les sorties de journalisation sont écrites dans le fichier **trunner.log.** Pour plus d'informations sur l'emplacement de ce fichier, consultez la section Présentation des fichiers journaux de la rubrique *Administration.*

Le niveau de journalisation utilisé est le même que celui défini dans le fichier **LISA_HOME\logging.properties.**

Pour modifier le niveau de journalisation, modifiez la propriété **log4j.rootCategory** dans le fichier **logging.properties,** de :

```
{{log4j.rootCategory=INFO,A1}}
```

à

```
{{log4j.rootCategory=DEBUG,A1}}
```

LISA Invoke

LISA Invoke est une application Web de type REST, qui vous permet d'effectuer les tâches suivantes avec une URL :

- Exécuter des scénarios de test
- Exécuter des suites
- Exécuter des archives de modèle (MAR)

Vous pouvez effectuer ces tâches de façon synchrone ou asynchrone.

La réponse consiste en un document XML.

Le fichier **lisa.properties** inclut les propriétés de configuration suivantes pour LISA Invoke :

```
lisa.portal.invoke.base.url=/lisa-invoke
lisa.portal.invoke.report.url=/reports
lisa.portal.invoke.server.report.directory={{lisa.tmpdir}}{{lisa.portal.invoke.report.url}}
lisa.portal.invoke.test.root={{LISA_HOME}}
```

Vous pouvez remplacer ces propriétés dans le fichier **local.properties.**

Pour afficher la page d'accueil de LISA Invoke, accédez à la page http://*nom_hôte*:1505/lisa/, où *nom_hôte* correspond à l'ordinateur sur lequel le registre s'exécute.

Exécution de scénarios de test à l'aide de LISA Invoke

Pour exécuter des scénarios de test à l'aide de LISA Invoke, utilisez la syntaxe suivante :

```
/lisa-invoke/runTest?testCasePath=testCasePath&stagingDocPath=s
tagingDocPath&[configPath=configPath]&[async=true]&[coordName=c
sName]
```

Les paramètres sont les suivants :

testCasePath

Chemin d'accès au scénario de test à appeler.

stagingDocPath

Chemin d'accès au document de simulation. Si aucun n'est fourni, un document de simulation par défaut est créé.

configPath

Chemin d'accès à la configuration. Si aucun n'est fourni, le fichier project.config du projet est utilisé.

async

Si la valeur est true, la réponse inclut une clé de rappel. Si aucune valeur n'est fournie, le paramètre est défini sur false.

coordName

Chemin d'accès au coordinateur. Si aucun n'est fourni, le nom de coordinateur par défaut est utilisé.

Exemple d'appel synchrone

L'URL suivante effectue un appel synchrone du scénario de test **AccountControlMDB** dans le projet **Examples** (Exemples).

```
http://localhost:1505/lisa-invoke/runTest?testCasePath=examples
/Tests/AccountControlMDB.tst&stagingDocPath=examples/StagingDoc
s/1user1cycle0think.stg
```

La réponse XML suivante indique que le scénario de test a réussi.

```
<?xml version="1.0" encoding="UTF-8"?>
<invokeResult>
  <method name="RunTest">
    <params>
      <param name="stagingDocPath"
value="examples/StagingDocs/1user1cycle0think.stg" />
      <param name="coordName" value="Coordinator" />
      <param name="configPath" value="" />
```

```
      <param name="testCasePath"
value="examples/Tests/AccountControlMDB.tst" />
      <param name="callbackKey"
value="64343533653737312D343765312D3439" />
    </params>
  </method>
  <status>OK</status>
  <result>
    <status>ENDED</status>

<reportUrl><![CDATA[http://localhost:1505/index.html?lisaPortal=
reporting/printPreview_functional.html#Idstr=61653261643936342D
613636392D3435&curtstr=T]]></reportUrl>
    <runId>61653261643936342D613636392D3435</runId>
    <pass count="1" />
    <fail count="0" />
    <warning count="0" />
    <error count="0" />
    <message>AccountControlMDB,Run1User1Cycle0Think</message>
  </result>
</invokeResult>
```

Exemple d'appel asynchrone

L'URL suivante effectue un appel asynchrone du scénario de test **AccountControlMDB** dans le projet **Examples** (Exemples).

```
http://localhost:1505/lisa-invoke/runTest?testCasePath=examples
/Tests/AccountControlMDB.tst&stagingDocPath=examples/StagingDoc
s/1user1cycle0think.stg&async=true
```

La réponse XML suivante affiche la clé de rappel dans l'élément result.

```
<?xml version="1.0" encoding="UTF-8"?>
<invokeResult>
  <method name="RunTest">
    <params>
      <param name="stagingDocPath"
value="examples/StagingDocs/1user1cycle0think.stg" />
      <param name="coordName" value="" />
      <param name="configPath" value="" />
      <param name="testCasePath"
value="examples/Tests/AccountControlMDB.tst" />
      <param name="callbackKey"
value="61663038653562382D663566372D3432" />
      <param name="async" value="true" />
    </params>
  </method>
```

```
  <status>OK</status>
  <result>
    <callbackKey>61663038653562382D663566372D3432</callbackKey>
    <message>The LISA test
'examples/Tests/AccountControlMDB.tst' was launched
asynchronously at Mon Mar 26 16:05:39 PDT 2012.</message>
  </result>
</invokeResult>
```

Exécution de suites de tests à l'aide de LISA Invoke

Pour exécuter des suites de tests à l'aide de LISA Invoke, utilisez la syntaxe suivante :

```
/lisa-invoke/runSuite?suitePath=suitePath&[configPath=configPath]&[async=true
]
```

Les paramètres sont les suivants :

suitePath

Chemin d'accès à la suite que vous voulez appeler.

configPath

Chemin d'accès à la configuration.

async

Si la valeur est true, la réponse inclut une clé de rappel. Si aucune valeur n'est fournie, le paramètre est défini sur false.

Exécution d'archives de modèle à l'aide de LISA Invoke

Pour exécuter des archives de modèles (fichiers MAR) à l'aide de LISA Invoke, utilisez la syntaxe suivante :

```
/lisa-invoke/runMar?marOrMariPath=[marOrMariPath]&[async=true]
```

Les paramètres sont les suivants :

marOrMariPath

Chemin d'accès au fichier MAR ou d'informations MAR que vous voulez appeler.

async

Si la valeur est true, la réponse inclut une clé de rappel. Si aucune valeur n'est fournie, le paramètre est défini sur false.

Appel du service de rappel à l'aide de LISA Invoke

Pour utiliser le service de rappel de LISA Invoke, vous devez avoir effectué un appel asynchrone d'un scénario de test ou d'une suite. La réponse XML contient la clé de rappel.

Pour appeler le service de rappel, utilisez la syntaxe suivante :

```
/lisa-invoke/callback?testOrSuitePath=[testOrSuitePath]&callbackKey=[callback
Key]&command=[status|kill|stop]
```

Les paramètres sont les suivants :

testOrSuitePath

Chemin d'accès au scénario de test ou à la suite.

callbackKey

Clé de rappel incluse dans la réponse de l'appel asynchrone.

command

Action à effectuer : status, kill ou stop.

Réponses de LISA Invoke

Cette section décrit les éléments qui peuvent s'afficher dans le document XML que LISA Invoke renvoie.

L'élément **method** indique le type d'exécution effectué.

L'élément **status** contient le statut de l'exécution : OK ou ERROR (Erreur).

L'élément **result** contient un ou plusieurs éléments enfants suivants :

status

Statut d'un scénario de test : RUNNING (En cours d'exécution) ou ENDED (Arrêté).

reportURL

URL du rapport dans la console Reporting Console (Console de génération de rapports).

runId

Identificateur unique de l'exécution.

pass

Nombre de tests réussis.

fail

Nombre de tests ayant échoué.

warning

Nombre de tests renvoyant des avertissements.

error

Nombre de tests renvoyant des erreurs.

message

Informations propres au type d'exécution.

tc

Nom d'un scénario de test inclus dans une suite.

callbackKey

Chaîne que vous pouvez utiliser pour effectuer des actions supplémentaires sur le scénario de test ou la suite.

API REST

L'API REST vous permet d'utiliser le style architectural REST pour effectuer des tâches DevTest.

Les API sont groupées selon les catégories suivantes :

- Serveurs de coordination
- Serveurs de simulation
- Laboratoires
- Serveurs de VSE

La documentation d'API complète est disponible ici. Cette documentation fournit des informations sur la méthode d'interaction avec les différents services RESTful.

Remarque : Vous devez utiliser Google Chrome ou Mozilla Firefox pour accéder à la documentation d'API.

Appel de l'API REST à partir de l'interface Web

DevTest Solutions inclut une interface Web simple que vous pouvez utiliser pour explorer l'API REST.

Remarque : Vous devez utiliser Google Chrome, Internet Explorer ou Mozilla Firefox pour accéder à l'interface Web.

Par défaut, l'interface Web utilise la variable **localhost** dans l'URL. Pour accéder à l'interface Web à partir d'autres hôtes que celui spécifié par la variable **localhost**, définissez la propriété suivante dans le fichier **local.properties** :

```
lisa.invoke2.swagger.basepath=http://<public_hostname_or_ip_address>:1505/api/Dcm
```

Procédez comme suit:

1. Vérifiez que le registre est en cours d'exécution.
2. Entrez **http://localhost:1505/api/swagger/** dans un navigateur pris en charge. Si le registre se trouve sur un ordinateur distant, remplacez **localhost** par le nom ou l'adresse IP de l'ordinateur.

   L'interface Web s’affiche.
3. Cliquez sur l'option Show/Hide (Afficher/Masquer) pour une catégorie.
4. Cliquez sur un des chemins d'API.
5. Utilisez le champ Response Content Type (Type de contenu de réponse) pour spécifier si la réponse est au format XML ou JSON.
6. Si la section Parameters (Paramètres) s'affiche, entrez la valeur requise pour chaque paramètre.
7. Cliquez sur Try it out! (Essayer).

   L'URL de demande et la réponse sont affichées.

Utilisation du module d'extension HP ALM - Quality Center

Le module d'extension HP ALM - Quality Center permet de charger et d'exécuter un scénario de test DevTest comme test Quality Center à partir d'une suite HP ALM - Quality Center. Vous pouvez importer et exécuter des tests DevTest à partir de Quality Center. Cette intégration vous permet de profiter de toutes les fonctionnalités de Quality Center associées à la puissance de test de DevTest. Le chargement d'un scénario de test DevTest dans Quality Center vous permet d'exécuter des tests DevTest en temps réel. Vous obtenez également une capture complète des résultats de test et des rappels DevTest renvoyés par les systèmes testés. Les tests DevTest peuvent être exécutés dans le flux de travaux de Quality Center et renvoyer les résultats afin de conserver le contexte et le statut du processus de test.

Remarque : Pour plus d'informations sur l'installation du module d'extension HP ALM - Quality Center, reportez-vous à la section *Installation*.

Cette section comprend les rubriques suivantes :

Configuration de tests DevTest dans HP ALM - Quality Center

Le module d'extension utilise VAPI-XP pour l'intégration à HP ALM - Quality Center.

Pour accéder aux modèles JavaScript et VBScript référencés dans cette procédure, cliquez sur **Démarrer,** Tous les programmes**,** DevTest**,** Quality Center Plugin (Module d'extension Quality Center). Du point de vue fonctionnel, les modèles sont équivalents.

Le test DevTest peut être un fichier de scénario de test, un fichier MAR ou un fichier d'informations MAR.

Pour les scénarios de test, vous pouvez également joindre un document de simulation et un fichier de configuration. Si vous ne joignez aucun document de simulation, un document de simulation est automatiquement créé avec les caractéristiques suivantes :

- Un utilisateur
- Un cycle
- Aucun délai de réflexion

Vous ne pouvez joindre aucun fichier supplémentaire aux fichiers MAR et aux fichiers d'informations MAR.

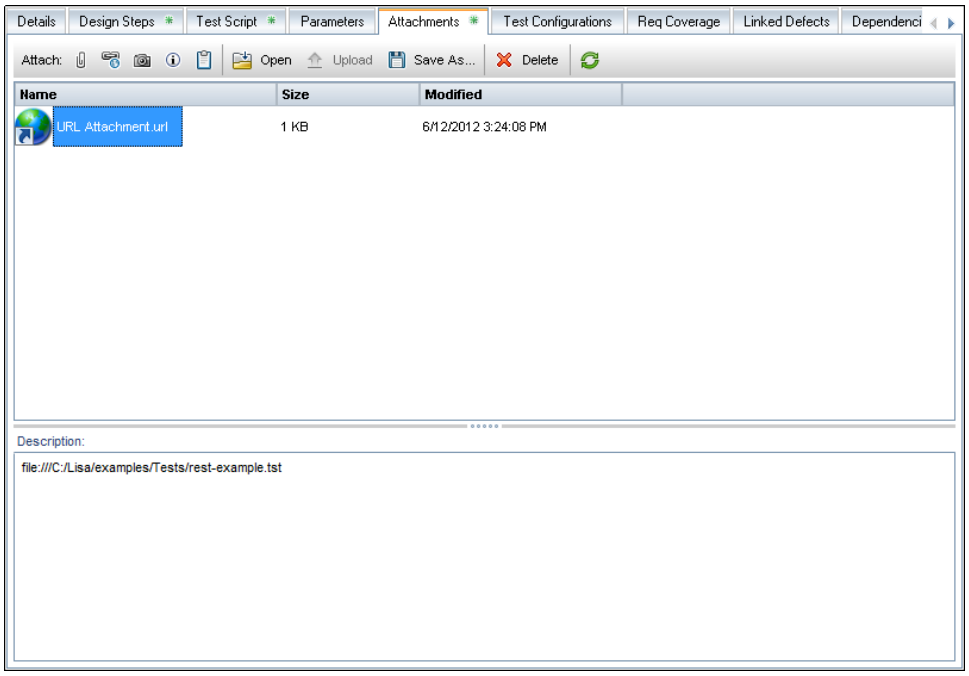
Dans cette procédure, vous créez une ou plusieurs pièces jointes d'URL. Exemple d'URL pour un fichier de scénario de test :

```
file:///C:/Lisa/examples/Tests/rest-example.tst
```

Exemple d'URL pour un fichier d'informations MAR :

```
file:///C:/Lisa/examples/MARInfos/rest-example.mari
```

Le graphique suivant présente l'onglet Attachments (Pièces jointes). L'URL pour un fichier de scénario de test a été ajoutée.



Procédez comme suit:

1. Créez un test VAPI-XP dans Quality Center.
2. Sélectionnez l'onglet Test Script et remplacez le contenu par défaut par le contenu du modèle JavaScript ou VBScript inclus avec le module d'extension.
3. Sélectionnez l'onglet Attachments et ajoutez une URL au fichier de scénario de test, au fichier MAR ou au fichier d'informations MAR.
4. (Facultatif) Ajoutez des URL au document de simulation et au fichier de configuration. Ces fichiers sont autorisés pour les scénarios de test, mais ne le sont pas pour les fichiers MAR ou les fichiers d'informations MAR.
5. Enregistrez le test VAPI-XP.

Exécution de tests DevTest dans HP ALM - Quality Center

Une fois que vous avez configuré un test DevTest dans HP ALM - Quality Center, vous pouvez l'exécuter à partir du module Test Plan (Planification de test) ou à partir du module Test Lab (Laboratoire de tests).

Pour des scénarios de test :

- Si un coordinateur est en cours d'exécution, il est utilisé pour exécuter le scénario de test.
- Dans le cas contraire, l'utilitaire Test Runner (Exécuteur de tests) est utilisé pour exécuter le scénario de test localement.

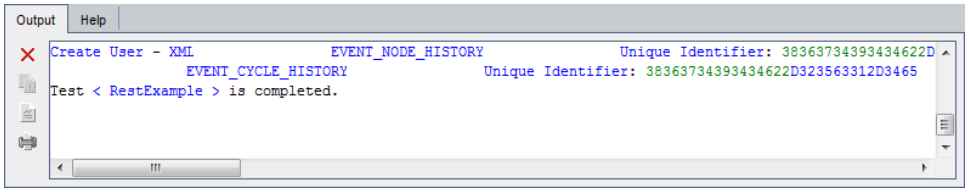
Pour des fichiers MAR et des fichiers d'informations MAR :

- Si le fichier spécifie un coordinateur, le coordinateur doit être en cours d'exécution.
- Si le fichier ne spécifie aucun coordinateur, le module d'extension crée et démarre un coordinateur local.

Par défaut, la commande Reload (Recharger) est définie pour être exécutée à chaque exécution de test. Cette commande remplit les étapes de conception du test qui sont nécessaires à une exécution de test réussie. Toutes les modifications apportées au test sont appliquées avant l'exécution du test. Pour certains tests, le processus de rechargement peut prendre du temps. Si vous savez que le fichier de test DevTest sous-jacent n'a pas été modifié, vous pouvez commenter cette ligne dans le fichier de script pour ignorer l'étape.

Pour exécuter le test à partir du module Test Plan, cliquez sur la flèche verte dans l'onglet Test Script. La sortie s'affiche dans la fenêtre du navigateur.

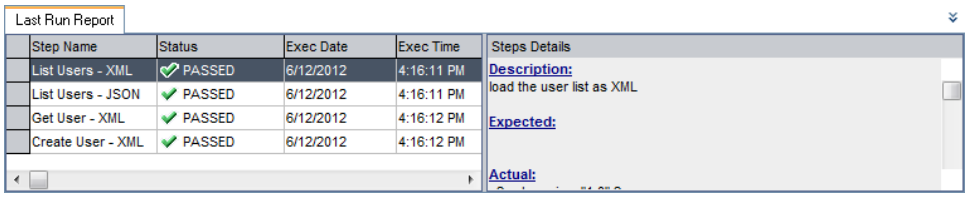
Le graphique suivant illustre les résultats de l'exécution du scénario de test **rest-example** dans le module Test Plan (Planification de test).



Pour exécuter le test à partir du module Test Lab, ajoutez le test à un ensemble de tests. Vous pouvez ensuite exécuter le test ou tout l'ensemble de tests.

Selon la structure du test, une exécution de test renvoie des résultats différents. Si le test contient plusieurs cycles exécutés, une liste de résultats d'historique de cycle s'affiche avec le statut Pass (Réussite) ou Fail (Echec). Si le test comprend uniquement un cycle, une liste d'étapes est affichée pour ce test.

Le graphique suivant illustre les résultats de l'exécution du scénario de test **rest-example** dans le module Test Lab (Laboratoire de tests).


Last Run Report
Step Name
Status
Exec Date
Exec Time
List Users - XML
PASSED
6/12/2012
4:16:11 PM
List Users - JSON
PASSED
6/12/2012
4:16:11 PM
Get User - XML
PASSED
6/12/2012
4:16:12 PM
Create User - XML
PASSED
6/12/2012
4:16:12 PM
Steps Details
Description:
load the user list as XML
Expected:
Actual:

Interface de ligne de commande LisaQCRunner

Le fichier exécutable **LisaQCRunner** du répertoire **LISA_HOME\bin** vous permet d'exécuter des tests DevTest Quality Center à partir de la ligne de commande. Vous pouvez conserver les résultats dans la base de données Quality Center.

Format du fichier exécutable :

```
LisaQCRunner [-h hôte] [-P port] [-u utilisateur] [-p mot_passe] [-D domaine] [-l
projet] run|debug|reload nom_test|all
```

L'hôte par défaut est localhost. Le port par défaut est 8080. L'utilisateur par défaut est admin. Le mot de passe par défaut est admin. Le domaine par défaut est DEFAULT. Le projet par défaut est Test.

L'argument de projet est le projet Quality Center et non le projet DevTest.

run

Exécute le nom de test que vous spécifiez comme argument. La sortie s'affiche dans la fenêtre de commande. Les résultats sont conservés dans Quality Center.

debug

Exécute le nom de test que vous spécifiez comme argument. La sortie s'affiche dans la fenêtre de commande. Les résultats ne sont pas conservés dans Quality Center.

reload

Permet de recharger le nom de test spécifié en tant qu'argument ou tous les tests DevTest (si l'argument est all).

L'exemple suivant présente l'exécution du scénario de test **rest-example.**

```
LisaQCRunner -h machine.example.com -P 8080 -u admin -p mypassword -D DEFAULT -l
myproject run RunWithTestandStage

Connecting...
Connected
Running RunWithTestAndStage with the following parameters:
Test Doc: file:///c:/lisa/examples/tests/rest-example.tst
Staging Doc: file:///c:/lisa/examples/stagingdocs/1user1cycle0think.stg
Config:
Mar:
List Users - XML     EVENT_NODE_HISTORY     Unique Identifier:
63366561643365642D643834302D3461 ...
List Users - JSON     EVENT_NODE_HISTORY     Unique Identifier:
63366561643365642D643834302D3461 ...
Get User - XML     EVENT_NODE_HISTORY     Unique Identifier:
63366561643365642D643834302D3461 ...
```

```
Create User - XML     EVENT_NODE_HISTORY     Unique Identifier:
63366561643365642D643834302D3461 ...
Disconnecting...
Disconnected
```

Dépannage du module d'extension HP ALM - Quality Center

Pour améliorer ses performances, la passerelle DevTest conserve toujours une référence au serveur COM DevTest. De cette manière, le serveur n'est pas instancié à chaque appel d'API. Lorsque le processus hébergeant le pont prend fin, cette référence est supprimée, sauf si l'hôte est une application native, telle qu'un navigateur Web ; dans ce cas, le processus **LisaQCRunner.exe** reste actif. Ce comportement constitue un problème uniquement lorsqu'un état incorrect est renvoyé (par exemple, à cause d'un arrêt brutal). Dans ce cas, arrêtez manuellement le processus **LisaQCRunner.exe** actif avant de poursuivre.

Echec de l'exécution du scénario de test Test Director

1. Vérifiez les URL de l'onglet Attachment (Pièce jointe) et vérifiez que le chemin d'accès et le nom de membre soient corrects. Le fichier journal de Test Director peut contenir une exception indiquant que le membre est introuvable.
2. Dans le cas de problèmes plus complexes, démarrez DevTest Workstation, accédez au test et exécutez le test en mode ITR. Notez les exceptions à l'origine de l'échec du scénario de test.
3. Vérifiez que les réponses XML et le fichier WSDL n'ont pas été modifiés. Les scénarios de test DevTest dépendent de commandes XPATH pour accéder à des valeurs spécifiques via des réponses XML. Si les réponses XML ont été modifiées, les commandes XPATH (filtres et assertions) peuvent ne pas trouver leurs cibles et causer un échec des scénarios de test.

Scénario de test Test Director : autorisation refusée

Si le message d'erreur You do not have the required permissions to execute this action s'affiche lors de l'exécution des tests :

1. Vérifiez qu'un administrateur HPQC a accordé les autorisations requises à votre ID pour exécuter des tests.
2. Vérifiez que DevTest Workstation est installé sur l'ordinateur à partir duquel le test est exécuté. Si DevTest Workstation est installé, vérifiez que le module d'extension Quality Center l'est également.
3. La version de navigateur est compatible avec les versions certifiées de HPQC.
4. Vérifiez que les fichiers .DLL de HPQC sont installés dans le dossier C:\Program Files\Common\Mercury Interactive\Quality Center.
5. Essayez d'exécuter le scénario de test à partir des modules Test Plan et Test Lab de QC, pour confirmer que les erreurs sont identiques.

Visionneuse HTTP and SSL Debug Viewer (Visionneuse de débogage HTTP et SSL)

La visionneuse HTTP and SSL Debug Viewer vous permet d'observer les activités HTTP et SSL dans DevTest Workstation. Cette fonctionnalité peut être utile pour effectuer un diagnostic.

Accédez à la visionneuse en sélectionnant Help (Aide), HTTP/SSL Debug (Débogage HTTP/SSL) à partir du menu principal.

La barre verticale sur le côté gauche indique la catégorie de chaque ligne :

- La couleur verte est utilisée pour les requêtes HTTP.
- La couleur bleu marine est utilisée pour les réponses HTTP.
- Les rayures sont utilisées pour les connexions SSL. Une barre violette supplémentaire s'affiche pour le récapitulatif d'établissement de liaison SSL.

Le code couleur suivant s'applique aux lignes :

- Le vert est utilisé pour les en-têtes de requêtes HTTP.
- Le bleu marine est utilisé pour les en-têtes de réponses HTTP.
- Le noir est utilisé pour les sorties normales.
- Le gris est utilisé pour les sorties non importantes.
- Le bleu-vert est utilisé pour les sorties intéressantes.
- Le magenta est utilisé pour les sorties importantes.
- L'orange sombre est utilisé pour les sorties d'avertissement.
- Le rouge est utilisé pour les sorties d'erreur.
- Le violet est utilisé pour les sorties récapitulatives.

Le nombre mis entre parenthèses au début de chaque ligne est un identificateur de thread. Plusieurs threads peuvent être utilisés sur la même connexion.

La visionneuse crée la sortie SSL en analysant les messages d'un journal de débogage.

La sortie SSL inclut un récapitulatif (page 363) du processus d'établissement de liaison. Lorsqu'un problème SSL est diagnostiqué, commencez par vérifier le récapitulatif d'établissement de liaison. Pour plus d'informations, consultez la sortie qui s'affiche avant le récapitulatif. La visionneuse peut ne pas contenir toutes les données nécessaires à la résolution du problème.

Vous pouvez copier la sortie et la coller dans une fenêtre distincte en tant que texte brut. Les couleurs ne sont pas incluses dans la version collée.

La visionneuse vous permet de spécifier les lignes à afficher : toutes les lignes, les lignes HTTP uniquement ou les lignes SSL uniquement.

Récapitulatif de l'établissement de liaison SSL

La sortie SSL dans la visionneuse HTTP and SSL Debug Viewer (page 362) (Visionneuse de débogage HTTP et SSL) inclut un récapitulatif des événements qui se sont produits pendant le processus d'établissement d'une liaison. Lorsqu'un problème SSL est diagnostiqué, commencez par vérifier le récapitulatif d'établissement de liaison.

Remarque : Pour comprendre cette rubrique, vous devez posséder des connaissances de base sur le protocole SSL ou son successeur, TLS.

Le graphique suivant illustre un exemple de récapitulatif.

```
[SSL Handshake Summary] Thread [Thread-48]
[SSL Handshake Summary] Acting as a Client
[SSL Handshake Summary]  *†‡ indicates linked optional steps
[SSL Handshake Summary]
[SSL Handshake Summary]  1  RUN                            Client Hello -->
[SSL Handshake Summary]  2  RUN                                         <-- Server Hello
[SSL Handshake Summary]  3* RUN                                         <-- Server Certificate (Public Key)
[SSL Handshake Summary]  4† SKIPPED                                     <-- Request Client Certificate
[SSL Handshake Summary]  5* ASSUMED  Verify and Trust Server Certificate v
[SSL Handshake Summary]  6‡ RUN                                         <-- Server Key Exchange
[SSL Handshake Summary]  7  RUN                                         <-- Server Hello Done
[SSL Handshake Summary]  8† SKIPPED      Client Certificate (Public Key) -->
[SSL Handshake Summary]  9† SKIPPED                                       v Verify and Trust Client Certificate
[SSL Handshake Summary] 10  RUN                    Client Key Exchange -->
[SSL Handshake Summary] 11† SKIPPED      Certificate Verify Confirmation -->
[SSL Handshake Summary] 12  RUN                Client Change Cipher Spec -->
[SSL Handshake Summary] 13  RUN                           Client Finished -->
[SSL Handshake Summary] 14  RUN                                         <-- Server Change Cipher Spec
[SSL Handshake Summary] 15  RUN                                         <-- Server Finished
```

La première ligne affiche le nom de thread.

La deuxième ligne indique si le journal de débogage SSL utilisé par la visionneuse fonctionne en tant que client ou serveur. Si une session a repris, la deuxième ligne affiche également un message correspondant.

Les lignes restantes indiquent les étapes du processus d'établissement d'une liaison.

Toutes les étapes possibles s'affichent dans le récapitulatif, même les étapes facultatives dans le protocole d'établissement d'une liaison. Dans les étapes facultatives, un symbole s'affiche à droite du numéro d'étape. Les étapes facultatives qui sont liées l'une à l'autre sont affichées avec des ensembles de symboles différents. Par exemple, un astérisque est utilisé pour l'étape 3 et l'étape 5, qui concernent le certificat du serveur.

Chaque étape comporte les statuts suivants :

- RUN (Exécuté) : l'étape a été effectuée.
- SKIPPED (Ignoré) : l'étape n'a pas été effectuée.
- ASSUMED (Supposé) : l'étape est supposée avoir été effectuée, selon d'autres événements qui se sont produits.
- UNKNOWN (Inconnu(e)) : statut initial de toutes les étapes. Si le statut n'a pas été modifié, l'étape n'a probablement pas été effectuée.

Chaque étape inclut une description brève d'une action que le client ou le serveur a effectuée. Par exemple, la première étape indique que le client envoie un message Hello au serveur. Si l'action implique l'envoi d'un message, une flèche vers la gauche ou la droite illustre la direction du flux de message. Si l'action n'implique aucun envoi de message, une flèche vers le bas s'affiche.

Si un problème SSL se produit, le récapitulatif fournit des conseils pour vous aider à en déterminer les causes. L'exemple suivant représente la sortie qui s'affiche lorsqu'une étape de test tente d'effectuer une demande HTTPS à un port non SSL.

```
SEND TLSv1 ALERT:  fatal, description = handshake_failure
javax.net.ssl.SSLHandshakeException: Remote host closed connection during handshake
Vérifiez que le serveur est sécurisé (connexion au serveur non sécurisé via SSL) et
que vous vous connectez au port approprié.
```

Chapitre 14: Laboratoires DevTest Cloud

Vous pouvez utiliser une infrastructure cloud pour provisionner les environnements de développement et de test.

Ce chapitre traite des sujets suivants :

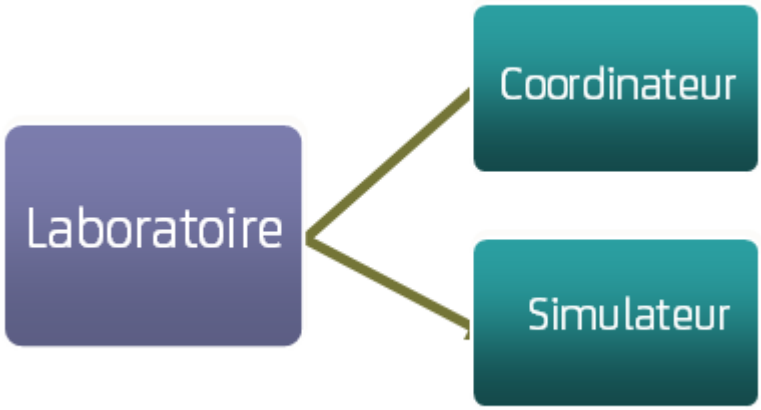
Laboratoires et membres de laboratoire

Un *laboratoire* est un conteneur logique pour un ou plusieurs membres de laboratoire.

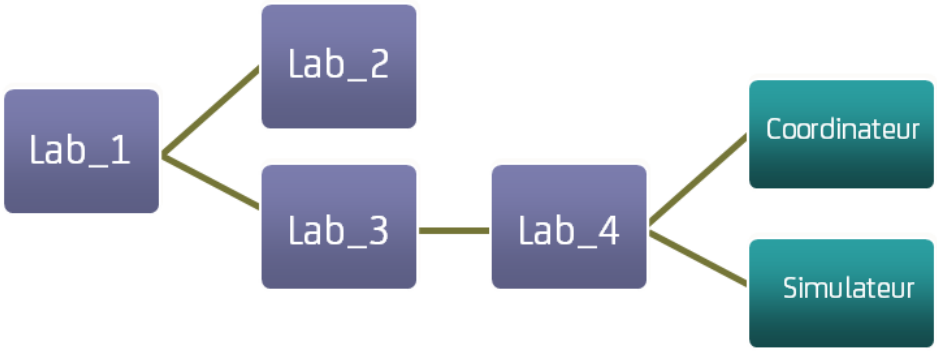
Un *membre de laboratoire* peut être un serveur DevTest ou un autre serveur.

- Les types valides de serveurs DevTest sont les coordinateurs, les simulateurs et les environnements de service virtuel.
- Les serveurs autres que DevTest sont par exemple les serveurs de base de données et les serveurs Web.

Le graphique suivant représente un laboratoire avec deux membres : un coordinateur et un simulateur.



Un laboratoire peut contenir un ou plusieurs laboratoires enfants. Le graphique suivant représente la nature hiérarchique des laboratoires. Lab_1 est le parent de Lab_2 et Lab_3. Lab_3 est le parent de Lab_4.



Le nom complet d'un laboratoire enfant comprend une barre oblique comme séparateur. Par exemple, le nom complet de Lab_4 dans le graphique précédent est Lab_1/Lab_3/Lab_4.

Un laboratoire **Default** (Par défaut) est inclus dans DevTest. Si vous démarrez un coordinateur, un simulateur ou l'environnement de service virtuel sans spécifier un laboratoire, le laboratoire Default est utilisé.

Un membre de laboratoire a l'un des statuts suivants :

- Uninitialized (Non initialisé)
- Starting (En cours de démarrage)
- Running (En cours d'exécution)
- Unknown (Inconnu)

Si un membre de laboratoire est un serveur DevTest, le statut suit le cycle Uninitialized (Non initialisé), Starting (En cours de démarrage), puis Running (En cours d'exécution).

Si un membre de laboratoire est un serveur autre que DevTest, le statut passe directement d'Uninitialized (Non initialisé) à Running (En cours d'exécution).

Gestionnaire virtuel de laboratoires (VLM)

L'environnement cloud dans lequel les laboratoires (page 366) sont exécutés est appelé un gestionnaire virtuel de laboratoires (VLM).

Le fournisseur de VLM suivant est pris en charge :

- VMware vCloud Director 1.0 et 1.5

Chaque fournisseur de VLM a un préfixe unique. Le tableau suivant répertorie ces préfixes.

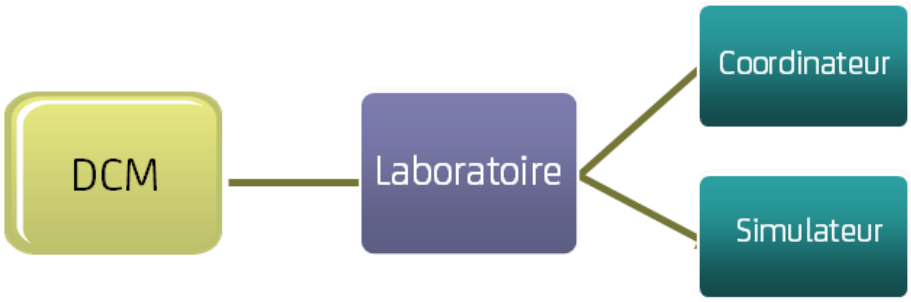
Fournisseur de VLM	**Préfixe**
VMware vCloud Director	VCD

Dans DevTest Workstation, le préfixe et deux-points s'affichent au début d'un nom de laboratoire complet pour indiquer le fournisseur de VLM utilisé. Par exemple :

VCD:MyOrganization/MyCatalog/MyLab

DevTest Cloud Manager

DevTest Cloud Manager (DCM) est le composant de DevTest Solutions qui interagit avec les laboratoires (page 366).

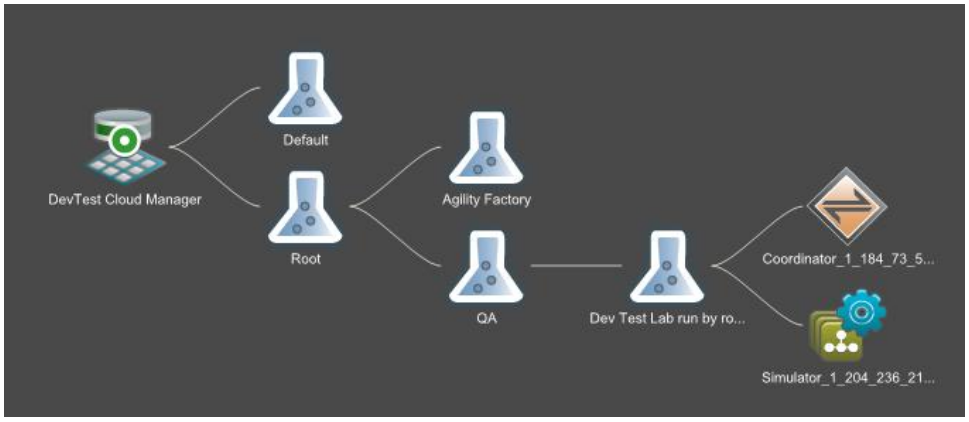


Les responsabilités de DCM incluent notamment :

- Envoi de propriétés relatives au cloud à un laboratoire
- Notification au fournisseur de VLM (Virtual Lab Manager, gestionnaire virtuel de laboratoires) que le démarrage d'un laboratoire est requis.
- Envoi de l'archive de modèle (MAR) (page 278) à un laboratoire

DCM vous permet d'afficher, de démarrer, de surveiller et d'arrêter des laboratoires à partir de la console Server Console (Console de serveur).

Cette console Server Console affiche le DCM dans le graphique de réseau. Dans le graphique suivant, le DCM interagit avec le laboratoire Default (Par défaut) et un laboratoire nommé Root (Racine). Le laboratoire Root a deux laboratoires enfants : Agility Factory et QA. Le laboratoire QA a un laboratoire enfant nommé Dev Test Lab. Le nom complet du laboratoire Dev Test Lab est Root/QA/Dev Test Lab.



Configuration des propriétés DCM

Pour activer le provisionnement des laboratoires de développement et de test, configurez leurs propriétés sur l'ordinateur dans lequel le registre se trouve.

Certaines propriétés sont applicables à tous les fournisseurs de gestionnaire virtuel de laboratoires (VLM). D'autres propriétés sont spécifiques à un fournisseur de VLM particulier.

- Propriétés générales (page 370)
- Propriétés vCloud Director (page 372)
- Exemple de propriétés de DCM (page 373)

Propriétés générales de DCM

Cette section décrit les propriétés applicables à tous les fournisseurs de VLM.

Vous devez configurer la propriété suivante dans le fichier **local.properties** :

- lisa.net.bindToAddress

Vous devez configurer les propriétés suivantes dans le fichier **site.properties** :

- lisa.dcm.labstartup.min
- lisa.dcm.lisastartup.min
- lisa.dcm.lisashutdown.min
- lisa.dcm.lab.cache.sec
- lisa.dcm.lab.factories
- lisa.net.timeout.ms

lisa.net.bindToAddress

Nom de domaine complet ou adresse IP de l'ordinateur dans lequel le registre se trouve. Le nom de domaine ou l'adresse IP doivent permettre leur adressage par un ordinateur connecté à un réseau situé à l'extérieur du réseau dans lequel le registre réside.

Syntaxe :

```
lisa.net.bindToAddress=<nom_domaine_complet_ou_adresse_IP>
```

lisa.dcm.labstartup.min

Délai en minutes pendant lequel DevTest patiente avant le démarrage d'un laboratoire par le fournisseur de VLM

Syntaxe :

```
lisa.dcm.labstartup.min=<nombre_entier>
```

lisa.dcm.lisastartup.min

Délai en minutes pendant lequel DevTest patiente après démarrage d'un laboratoire par le fournisseur de VLM pour l'initialisation correcte de DevTest

Syntaxe :

```
lisa.dcm.lisastartup.min=<nombre_entier>
```

lisa.dcm.lisashutdown.min

Délai en minutes pendant lequel DevTest patiente à l'issue de l'exécution de test pour arrêter un laboratoire

Syntaxe :

```
lisa.dcm.lisashutdown.min=<nombre_entier>
```

lisa.dcm.lab.cache.sec

DevTest gère un cache de la configuration de projet VLM. Cette propriété spécifie la fréquence à laquelle DevTest accède au fournisseur de VLM pour déterminer si le cache doit être mis à jour (par exemple, en cas d'ajout d'un environnement). La valeur est exprimée en secondes.

Syntaxe :

```
lisa.dcm.lab.cache.sec=<nombre_entier>
```

Valeur par défaut : 180

lisa.dcm.lab.factories

Nom de l'objet qui contient la logique de prise en charge cloud pour un fournisseur de VLM spécifique. La seule valeur valide est **com.itko.lisa.cloud.vCloud.vCloudDirectorCloudSupport**.

Syntaxe :

```
lisa.dcm.lab.factories=<nom_objet>
```

lisa.net.timeout.ms

Valeur du délai d'expiration (en millisecondes) utilisée par le système de messagerie sous-jacent. Cette propriété n'est pas spécifique au cloud. Modifiez la valeur pour l'intégration cloud, car certaines opérations peuvent tarder plus longtemps que le délai par défaut.

Syntaxe :

```
lisa.net.timeout.ms=<nombre_entier>
```

Valeur par défaut : 30

Propriétés vCloud Director

Cette section décrit les propriétés propre à VMware vCloud Director.

Configurez les propriétés suivantes dans le fichier **site.properties** :

- lisa.dcm.vCLOUD.baseUri

Configurez également les propriétés suivantes.

- lisa.dcm.vCLOUD.userId
- lisa.dcm.vCLOUD.password

DevTest crée les autorisations personnalisées **lisa.dcm.vCLOUD.userId** et **lisa.dcm.vCLOUD.password**, puis les affecte à chaque rôle. Vous pouvez spécifier des valeurs initiales pour ces autorisations personnalisées en configurant les propriétés **lisa.dcm.vCLOUD.userId** et **lisa.dcm.vCLOUD.password** dans le fichier **site.properties.** Si vous ne spécifiez pas les valeurs initiales, définissez les valeurs à partir de la console Server Console (Console de serveur). Consultez la rubrique sur l'ajout, la mise à jour et la suppression des rôles.

lisa.dcm.vCLOUD.baseUri

URL de connexion de l'API vCloud.

Syntaxe :

```
lisa.dcm.vCLOUD.baseUri=<url>
```

lisa.dcm.vCLOUD.userId

Nom d'utilisateur qui peut se connecter à vCloud Director. En général, le format est constitué du nom d'utilisateur, suivi d'une arobase (@), suivie par le nom de l'organisation.

Syntaxe :

```
lisa.dcm.vCLOUD.userId=<user-name>@<organization-name>
```

lisa.dcm.vCLOUD.password

Mot de passe pour le nom d'utilisateur défini dans la propriété lisa.dcm.vCLOUD.userId.

Syntaxe :

```
lisa.dcm.vCLOUD.password=<password>
```

Exemple de propriétés de DCM

L'exemple suivant présente une configuration basée sur vCloud Director.

La propriété suivante se trouve dans le fichier **local.properties** :

```
lisa.net.bindToAddress=myserver.example.com
```

Les propriétés suivantes se trouvent dans le fichier **site.properties** :

```
lisa.dcm.labstartup.min=6
lisa.dcm.lisastartup.min=4
lisa.dcm.lisashutdown.min=2
lisa.dcm.lab.cache.sec=240
lisa.dcm.lab.factories=com.itko.lisa.cloud.vCloud.vCloudDirectorCloudSupport

lisa.dcm.vCLOUD.baseUri=https://www.example.com/api/versions
lisa.dcm.vCLOUD.userId=administrator@System
lisa.dcm.vCLOUD.password=mypassword

lisa.net.timeout.ms=60000
```

Configuration de vCloud Director

Pour activer le provisionnement de laboratoires de développement et de test avec VMware vCloud Director en tant que fournisseur de VLM (gestionnaire virtuel de laboratoires), configurez vCloud Director.

Cette rubrique utilise les concepts vCloud Director suivants :

- catalogue
- Application virtuelle (vApp)
- Modèle d'application virtuelle (vApp template)

Les applications virtuelles et les modèles d'application virtuelle de vCloud Director correspondent aux laboratoires dans DevTest.

Pendant la procédure de configuration, vous créez des images de machine virtuelle pour les composants de DevTest Server suivants :

- Coordinateur
- Simulator (Simulateur)
- Virtual Service Environment (Environnement de service virtuel)

Si vous voulez que les laboratoires incluent uniquement un coordinateur et un simulateur, ne créez pas d'image pour le VSE.

Si vous voulez que les laboratoires incluent uniquement un environnement de service virtuel, ne créez pas d'images pour le coordinateur et le simulateur.

Chaque composant de serveur doit être configuré de sorte à démarrer en mode **remoteInit**. Ce mode permet de démarrer le composant de serveur, puis d'attendre que DevTest Cloud Manager (DCM) envoie les paramètres d'initialisation requis.

Pour configurer vCloud Director :

1. A l'aide d'une application comme VMware Workstation, créez une image de machine virtuelle pour chaque composant de serveur à inclure dans un laboratoire.
   - Pour démarrer le composant de serveur en mode **remoteInit**, configurez chaque image.
   - Le nom d'un coordinateur doit inclure le terme **Coordinator**.
   - Le nom d'un simulateur doit inclure le terme **Simulator**.
   - Le nom d'un environnement de service virtuel doit inclure le terme **VSE**.
2. Connectez-vous à la console Web vCloud Director en tant qu'administrateur.
3. Importez chaque image de machine virtuelle que vous avez créée en tant que modèle d'application virtuelle.
4. Créez une application virtuelle à partir des modèles d'application virtuelle.

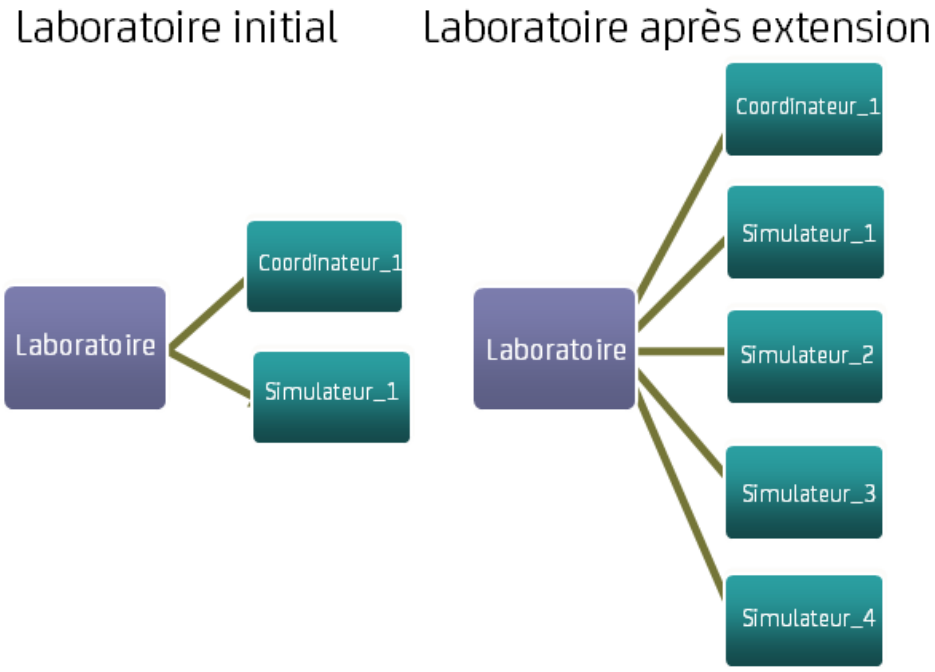
5. Ajoutez l'application virtuelle à un catalogue.

Extension dynamique des laboratoires de test

DevTest peut déterminer qu'une capacité supplémentaire est requise pour répondre aux besoins d'un test en cours d'exécution et étendre automatiquement le laboratoire.

Utilisez un document de simulation qui contient le modèle de distribution Dynamic Simulator Scaling with DCM (Mise à l'échelle du simulateur dynamique avec DCM). Pour plus d'informations, consultez la rubrique Sélection de la distribution *(page 245)*.

Le graphique suivant affiche un exemple. Dans la partie gauche, le laboratoire a initialement un coordinateur et un simulateur. Dans la partie droite, le laboratoire contient un coordinateur et quatre simulateurs à l'issue de l'extension.



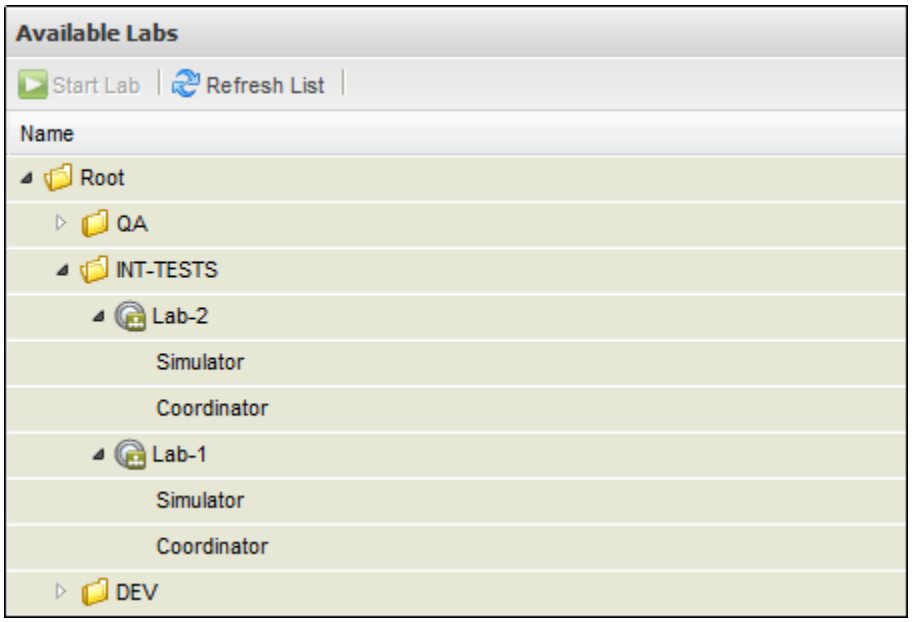
Liste des laboratoires disponibles

Vous pouvez utiliser la console Server Console (Console de serveur) ou LISA Invoke pour répertorier les laboratoires de développement et de test qui sont disponibles dans l'environnement cloud.

Liste des laboratoires disponibles à partir de la console Server Console

Pour des instructions sur l'accès à la console Server Console (Console de serveur), consultez la section Ouverture de la (Console de serveur) de la rubrique *Administration.*

Le graphique suivant affiche une liste des laboratoires disponibles dans la console Server Console. L'arborescence est développée afin d'afficher les deux laboratoires enfants.



Pour répertorier les laboratoires disponibles :

1. Affichez le panneau Network (Réseau) dans la console Server Console (Console de serveur).
2. Développez le noeud DevTest Cloud Manager.
3. Cliquez sur le noeud Available Labs (Laboratoires disponibles).

   Les laboratoires disponibles s'affichent dans le panneau droit. L'arborescence est initialement réduite.

Liste des laboratoires disponibles à l'aide de LISA Invoke

Vous pouvez utiliser LISA Invoke (page 347) pour répertorier les laboratoires disponibles. La syntaxe est la suivante :

```
/lisa-invoke/listRunnableLabs
```

La réponse consiste en un document XML. Les informations suivantes sont fournies pour chaque laboratoire.

- Nom du laboratoire
- Clé de laboratoire
- Nom de chaque membre de laboratoire

La clé de laboratoire est un paramètre requis dans les opérations **startLab**, **getLab** et **killLab**.

Exemple

L'URL suivante demande la liste des laboratoires disponibles.

```
http://localhost:1505/lisa-invoke/listRunnableLabs
```

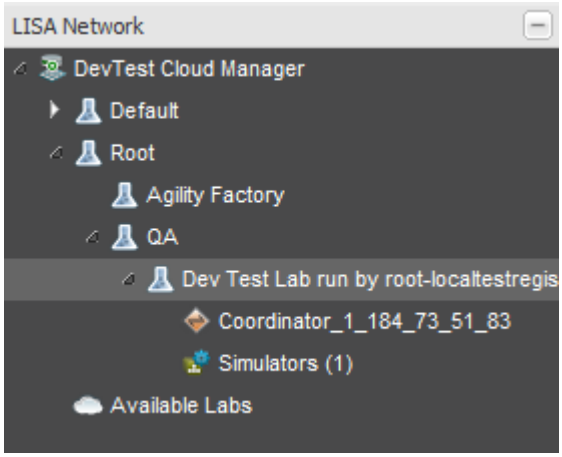
La réponse suivante contient une liste de trois laboratoires disponibles.

```
<?xml version="1.0" encoding="UTF-8" ?>
<invokeResult>
  <method name="ListRunnableLabs">
    <params />
  </method>
  <status>OK</status>
  <result>
    <lab name="MM-Test" key="VCD:7">
      <labMember name="Simulator" />
      <labMember name="Coordinator" />
    </lab>
    <lab name="MM-Test run by rich-47" key="VCD:47">
      <labMember name="Simulator_1_184_72_204_206" />
      <labMember name="Simulator_2_107_21_199_227" />
      <labMember name="Coordinator_1_184_73_83_115" />
    </lab>
    <lab name="VSE-Cluster run by rich-46" key="VCD:46">
      <labMember name="V_S_E_Member_1_23_21_11_148" />
      <labMember name="V_S_E_Member_2_23_21_3_45" />
      <labMember name="V_S_E_Member_4_184_73_65_217" />
      <labMember name="V_S_E_Member_3_174_129_88_179" />
    </lab>
  </result>
</invokeResult>
```

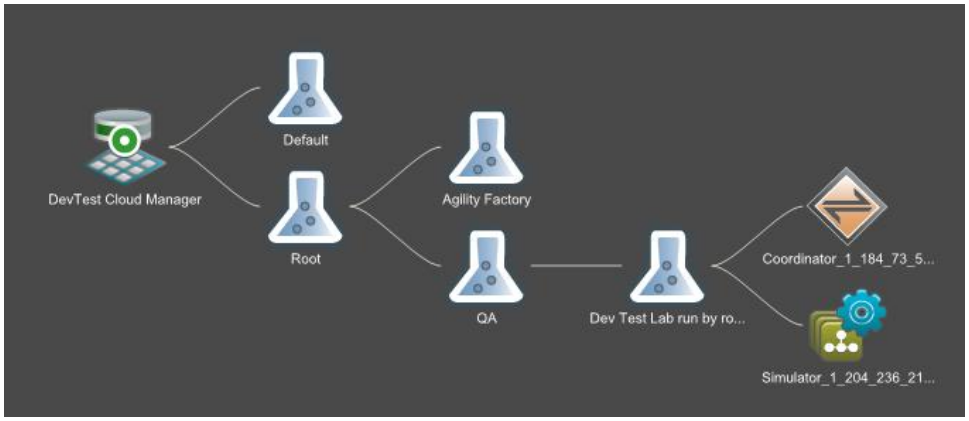
Démarrage d'un laboratoire

Lorsque vous démarrer un laboratoire cloud, vous démarrez une instance de ce laboratoire. Un laboratoire peut avoir plusieurs instances. Toutes sont indépendantes l'une de l'autre.

Dans la console Server Console (Console de serveur), le panneau Network (Réseau) affiche le laboratoire démarré dans une arborescence. Le graphique suivant illustre le panneau Network (Réseau).



Le panneau droit affiche le laboratoire démarré dans le graphique de réseau. L'illustration suivante affiche le graphique de réseau.



La propriété **lisa.dcm.labstartup.min** contrôle le délai en minutes pendant lequel DevTest patiente avant le démarrage du laboratoire par le fournisseur de VLM (gestionnaire virtuel de laboratoires). Pour plus d'informations, consultez la rubrique Configuration des propriétés DCM *(page 369)*.

Remarque : Vous ne devez pas démarrer le laboratoire Default (Par défaut) explicitement.

Trois méthodes s'offrent à vous pour démarrer un laboratoire :

- A partir de la ligne de commande. (page 379)
- A partir de la console Server Console (page 379)
- A l'aide de LISA Invoke (page 380)

Démarrage d'un laboratoire à partir de la ligne de commande

Vous pouvez démarrer un laboratoire en appelant un des fichiers exécutables de DevTest Server et en spécifiant un nom de serveur et un nom de laboratoire.

Par exemple, la commande suivante permet de lancer un laboratoire nommé MyLab, dont le parent est MyParentLab. Le laboratoire MyLab comprend un membre de laboratoire : un coordinateur nommé Dev-Coord.

```
CoordinatorServer -n Dev-Coord -l MyParentLab/MyLab
```

Démarrage d'un laboratoire à partir de la console Server Console (Console de serveur)

Cette procédure suppose que vous avez répertorié les laboratoires disponibles (page 376) à partir de la console Server Console.

Procédez comme suit:

1. Sélectionnez le laboratoire dans la liste des laboratoires disponibles.
2. Cliquez sur Start Lab (Démarre le laboratoire).
3. Patientez pendant le démarrage du laboratoire.

   Lorsque la séquence de démarrage est terminée, un message indique que le laboratoire a été démarré.

   Le panneau Network (Réseau) affiche le laboratoire démarré dans une arborescence. Le panneau droit affiche le laboratoire démarré dans le graphique de réseau.
4. Patientez pendant le démarrage des membres du laboratoire.

   Lorsque la séquence de démarrage pour les membres de laboratoire est terminée, le statut du membre est modifié sur Running (En cours d'exécution). De plus, les statistiques sur le membre de laboratoire s'affichent dans l'onglet Component Health Summary (Récapitulatif de l'intégrité des composants).

Démarrage d'un laboratoire à l'aide de LISA Invoke

Vous pouvez utiliser LISA Invoke (page 347) pour démarrer un laboratoire. La syntaxe est la suivante :

```
/lisa-invoke/startLab?labKey=LAB:key
```

Les clés de laboratoire disponibles sont incluses dans la réponse à l'opération **listRunnableLabs**. Pour plus d'informations, consultez la rubrique Liste des laboratoires disponibles (page 376).

La réponse est un document XML qui inclut un message de statut.

Exemple

L'URL suivante effectue une demande de démarrage pour le laboratoire dont la clé est **VCD:49**.

```
http://localhost:1505/lisa-invoke/startLab?labKey=VCD:49
```

La réponse suivante indique que le laboratoire a été démarré.

```
<?xml version="1.0" encoding="UTF-8" ?>
<invokeResult>
  <method name="StartLab">
    <params />
  </method>
  <status>OK</status>
  <result>
    <lab name="MM-Test run by rich-49" key="VCD:49" />
  </result>
</invokeResult>
```

Obtention d'informations sur un laboratoire

Vous pouvez utiliser LISA Invoke (page 347) pour obtenir des informations de base sur un laboratoire en cours d'exécution. La syntaxe est la suivante :

```
/lisa-invoke/getLab?labKey=LAB:key
```

Les clés de laboratoire disponibles sont incluses dans la réponse à l'opération **listRunnableLabs**. Pour plus d'informations, consultez la rubrique Liste des laboratoires disponibles (page 376).

La réponse consiste en un document XML. Les informations suivantes sont fournies :

- Nom du laboratoire
- Clé de laboratoire
- Nom, nom de service et adresse IP de chaque membre de laboratoire

Si le laboratoire est en cours de démarrage, une partie des informations n'est pas disponible.

Si le laboratoire est introuvable, la réponse contient un message d'erreur.

Exemple

L'URL suivante effectue une demande d'informations sur le laboratoire dont la clé est **VCD:50**.

```
http://localhost:1505/lisa-invoke/getLab?labKey=VCD:50
```

La réponse suivante contient les informations sur le laboratoire.

```
<?xml version="1.0" encoding="UTF-8" ?>
<invokeResult>
  <method name="GetLab">
    <params />
  </method>
  <status>OK</status>
  <result>
    <lab name="MM-Test run by rich-50" key="VCD:50">
      <labMember name="Coordinator_1_50_16_15_3"
serviceName="tcp://50.16.15.3:2011/Coordinator_1_50_16_15_3" ip="50.16.15.3"
/>
      <labMember name="Simulator_2_23_20_80_144"
serviceName="tcp://23.20.80.144:2014/Simulator_2_23_20_80_144"
ip="23.20.80.144" />
      <labMember name="Simulator_1_23_21_5_69"
serviceName="tcp://23.21.5.69:2014/Simulator_1_23_21_5_69" ip="23.21.5.69" />
    </lab>
```

```
  </result>
</invokeResult>
```

Déploiement d'un fichier MAR sur un laboratoire

Vous pouvez déployer l'archive de modèle (MAR) pour un scénario de test ou une suite vers un laboratoire qui a été démarré (page 378). Le laboratoire doit inclure un coordinateur et, dans la plupart des cas, un simulateur.

Procédez comme suit:

1. Dans le panneau Network (Réseau) de la console Server Console (Console de serveur), cliquez sur le coordinateur.

   La fenêtre de détails pour le coordinateur s'affiche dans le panneau droit.

2. Dans le panneau droit, cliquez sur Deploy MAR (Déployer un fichier MAR).

   La boîte de dialogue Deploy MAR s'affiche.

3. Cliquez sur Browse (Parcourir) et sélectionnez le fichier .mar.

4. Cliquez sur Deploy (Déployer).

   Un message indique que l'archive de modèle a été déployée. Le scénario de test ou la suite est désormais en cours d'exécution.

5. Cliquez sur OK.

Arrêt d'un laboratoire

Vous pouvez utiliser la console Server Console (Console de serveur) ou LISA Invoke pour arrêter un laboratoire en cours d'exécution. Cette action est permanente et ne peut pas être annulée.

Si le laboratoire est un laboratoire enfant, les laboratoires parents ne sont pas arrêtés.

Remarque : Vous ne pouvez pas supprimer le laboratoire Default (Par défaut).

Arrêt d'un laboratoire à partir de la console Server Console (Console de serveur)

Dans le graphique du réseau, cliquez avec le bouton droit de la souris sur le laboratoire et sélectionnez Stop (Arrêter).

Lorsque l'action est terminée, un message indique que le laboratoire a été arrêté.

Arrêt d'un laboratoire à l'aide de LISA Invoke

Vous pouvez utiliser LISA Invoke (page 347) pour arrêter un laboratoire. La syntaxe est la suivante :

```
/lisa-invoke/killLab?labKey=LAB:key
```

Les clés de laboratoire disponibles sont incluses dans la réponse à l'opération **listRunnableLabs**. Pour plus d'informations, consultez la rubrique Liste des laboratoires disponibles (page 376).

La réponse est un document XML qui inclut un message de statut.

Exemple

L'URL suivante effectue une demande d'arrêt pour le laboratoire dont la clé est **VCD:49**.

```
http://localhost:1505/lisa-invoke/killLab?labKey=VCD:49
```

La réponse suivante indique que le laboratoire a été arrêté.

```
<?xml version="1.0" encoding="UTF-8" ?>
<invokeResult>
  <method name="KillLab">
    <params />
  </method>
  <status>OK</status>
```

```
  <result>
    <message>Lab with key VCD:49 has been deleted.</message>
  </result>
</invokeResult>
```

Chapitre 15: Service de validation en continu

Le service Continuous Validation Service (Service de validation en continu) vous permet de planifier des tests et des suites de tests à exécuter régulièrement sur une période étendue.

Le service de validation en continu comprend un tableau de bord qui gère une liste de tous les tests planifiés (services) et leur statut. Dans le tableau de bord CVS Dashboard (Tableau de bord du service CVS), vous pouvez sélectionner un test et surveiller chacune de ses exécutions.

Un moniteur contient un test unique ou une suite de tests complète. Un service contient un ou plusieurs moniteurs.

Un coordinateur gère les tests, qui s'exécutent sur un simulateur. L'état est enregistré dans une base de données du registre DevTest.

Vous pouvez exécuter un service ou des moniteurs spécifiques à partir du tableau de bord CVS Dashboard (Tableau de bord du service CVS).

Conditions préalables

Le service de validation en continu est exécuté dans un environnement DevTest Server.

Un serveur de coordination et un serveur de simulation doivent être en cours d'exécution et enregistrés dans le registre DevTest.

Pour plus d'informations sur la configuration de l'environnement du DevTest Server, consultez la rubrique *Installation.*

Ce chapitre traite des sujets suivants :

Ouverture du tableau de bord CVS Dashboard (Tableau de bord du service CVS)

Vous pouvez ouvrir le tableau de bord CVS Dashboard (Tableau de bord du service CVS) à partir de DevTest Workstation ou à partir d'un navigateur Web.

Pour ouvrir le tableau de bord CVS Dashboard à partir de DevTest Workstation :

Dans le menu principal, sélectionnez View (Afficher), CVS Dashboard.

Pour ouvrir le tableau de bord CVS Dashboard à partir d'un navigateur Web :

1. Vérifiez que le registre (page 11) est en cours d'exécution.
2. Dans un navigateur Web, entrez **http://localhost:1505/**.

   Si le registre se trouve sur un ordinateur distant, remplacez **localhost** par le nom ou l'adresse IP de l'ordinateur.

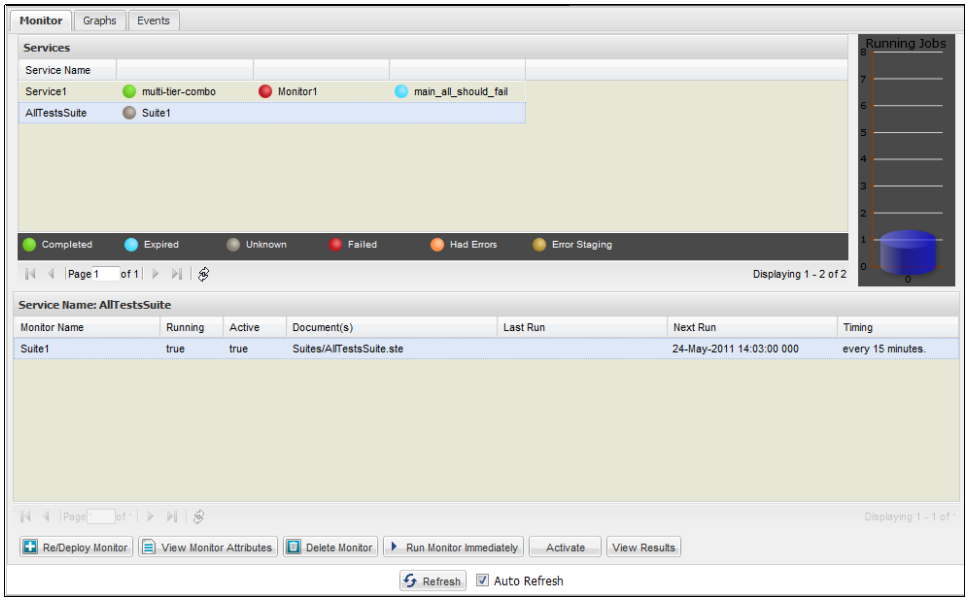
   La console DevTest s'affiche.
3. Cliquez sur Continuous Validation Service (Service de validation en continu).

Remarque : Fermer le tableau de bord CVS Dashboard ou fermer DevTest Workstation n'affecte pas les tâches planifiées du service de validation en continu. Lorsque vous vous reconnectez au registre, le tableau de bord affiche les données et les informations sur le statut actuelles.

Présentation du tableau de bord CVS Dashboard (Tableau de bord du service CVS)

Le tableau de bord CVS Dashboard (Tableau de bord du service CVS) comporte les onglets suivants:

- Monitor (page 389) (Moniteur) : affiche une liste de tous les moniteurs (tests/suites de tests) qui sont ajoutés au tableau de bord CVS Dashboard. Votre tableau de bord CVS Dashboard répertorie tous les moniteurs en cours d'exécution dans le registre DevTest associé, pas seulement ceux que vous initialisez.
- Graphs (page 393) (Graphiques) : affiche le statut du tableau de bord de manière graphique. Cet onglet indique également le pourcentage des tests réussis ou ayant échoué.
- Events (page 395) (Events) : affiche les événements de statut que les moniteurs enregistrent.



Pour actualiser la liste affichée dans le tableau de bord CVS Dashboard (Tableau de bord du service CVS), cliquez sur Refresh (Actualiser) en bas de la fenêtre.

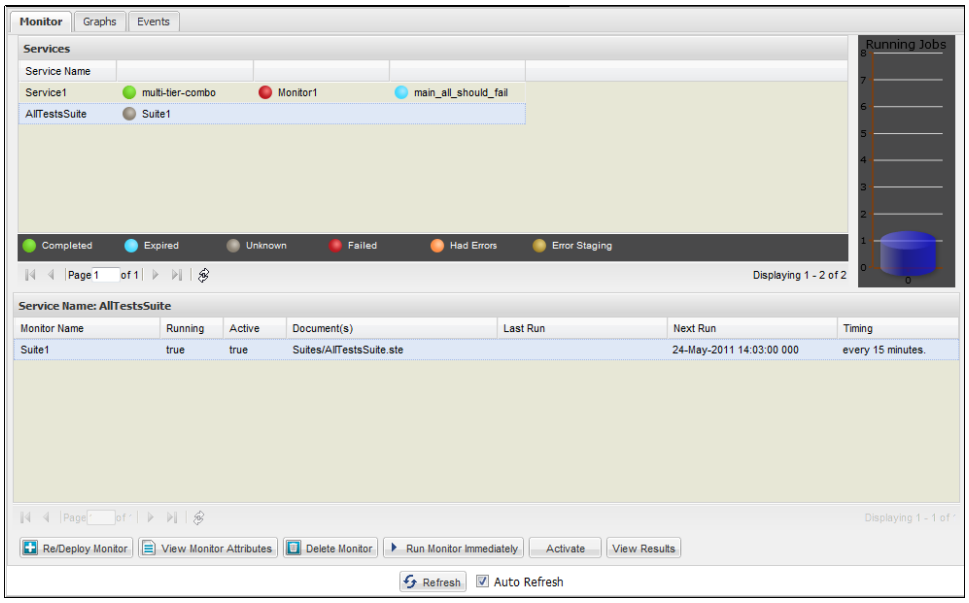
Onglet Monitor (Moniteur) du tableau de bord CVS Dashboard (Tableau de bord du service CVS)

Le tableau de bord CVS Dashboard (Tableau de bord du service CVS) s'ouvre avec l'onglet Moniteur actif.

Vous pouvez ajouter plusieurs moniteurs à exécuter dans un ou plusieurs services. Un service contient un ou plusieurs moniteurs.

Vous pouvez planifier l'exécution d'un service. Tous les moniteurs de ce service sont exécutés à l'heure planifiée. Les services sont exécutés en arrière-plan à des intervalles planifiés.

Vous pouvez ajouter les moniteurs dans un service (dans l'exemple suivant, **Service1**) ou les ajouter dans des services différents (dans l'exemple suivant, **Service1** et **AllTestsSuite**). Tous les moniteurs ajoutés dans un service (par exemple, **Service1**) sont affichés comme ajoutés après le nom du service.

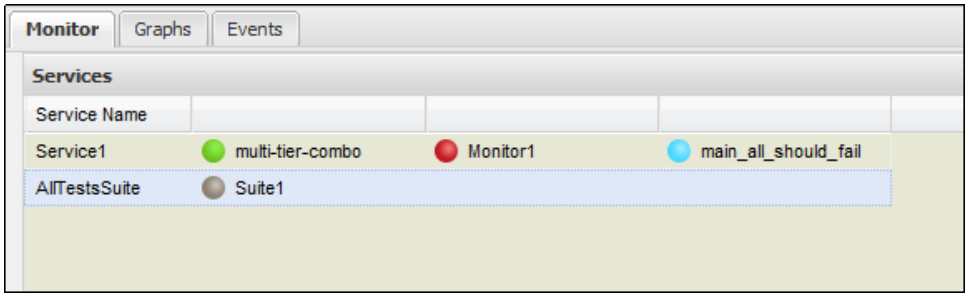


Panneau supérieur

Tous les tests d'un service qui ont été exécutés sont décrits avec un cercle coloré :

- **Vert :** tests terminé.
- **Rouge :** tests ayant échoué
- **Jaune :** tests contenant une erreur dans leurs documents de simulation

Les icônes décrivent l'achèvement de l'exécution de test, un échec, une erreur, une erreur de simulation ou une erreur inconnue. Ces icônes décrivent l'état du moniteur après son exécution.

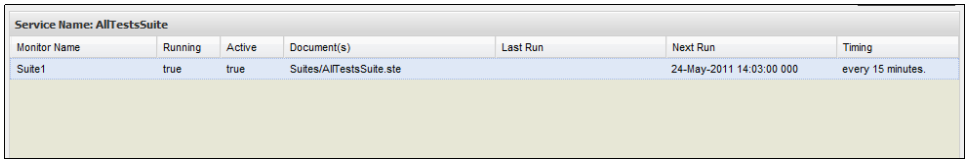


Sur le côté droit, le panneau supérieur affiche une représentation graphique des jobs actuellement en cours d'exécution. Par exemple, le graphique suivant indique qu'un job est actuellement en cours d'exécution.

Panneau inférieur

Le panneau inférieur affiche le statut de chaque moniteur qui est exécuté dans un service.

Ce panneau affiche également tous les moniteurs en cours d'exécution ou ayant terminé leur dernière exécution. Les moniteurs dont l'exécution est complètement terminée (sans aucune autre exécution planifiée) ne s'affichent pas dans cette liste.

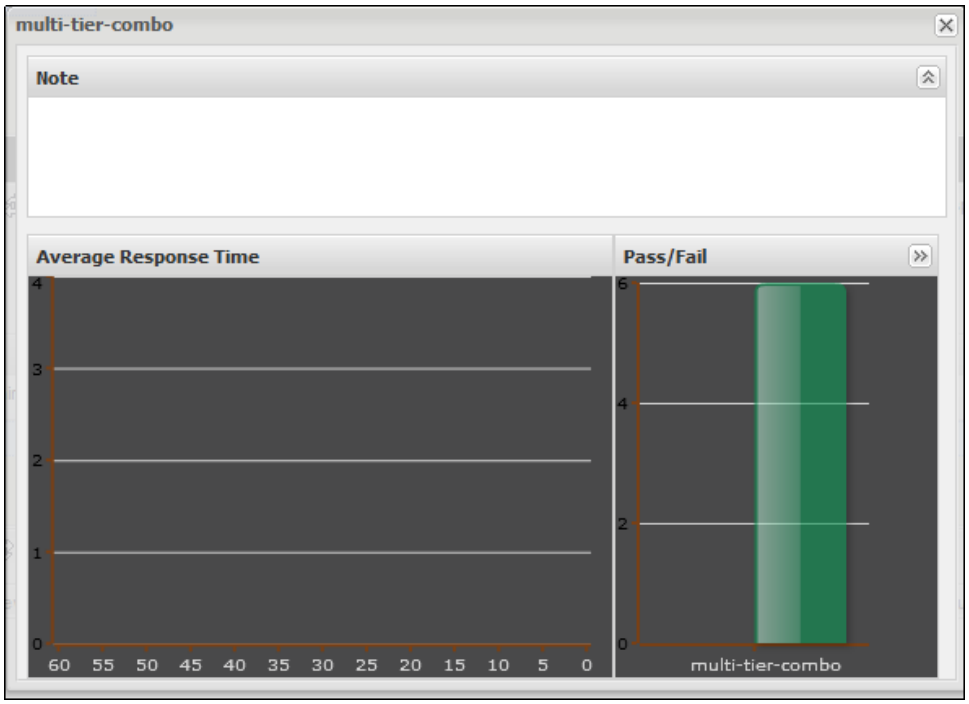


La table affiche les informations suivantes :

- **Monitor Name (Nom du moniteur)** : nom donné au moniteur
- **Running (En cours d'exécution)** : indique si le moniteur est en cours d'exécution (true/false).
- **Active (Actif)** : indique si le moniteur est actif (true/false).
- **Documents** : noms des différents documents, tels que le document de scénario de test, de simulation et de suite, qui sont associés au moniteur
- **Last Run (Dernière exécution)** : date et heure de la dernière exécution du moniteur
- **Next Run (Prochaine exécution)** : heure de début planifiée pour l'exécution suivante du moniteur
- **Timing (Calcul du temps)** : détails de planification pour le moniteur

Vous pouvez personnaliser les colonnes en les affichant selon l'ordre de tri, en les affichant ou en les masquant.

Pour consulter les informations relatives à un moniteur, double-cliquez sur celui-ci.



Barre d'outils

L'onglet Toolbar (Barre d'outils) inclut une barre d'outils comprenant les boutons suivants :

- **Re/Deploy Monitor (Redéployer/Déployer un moniteur)** : permet de déployer ou de redéployer un moniteur dans le tableau de bord. Reportez-vous à la rubrique Déploiement d'un moniteur vers le service de validation en continu (page 396).
- **View Monitor Attributes (Afficher les attributs du moniteur)** : permet d'afficher des informations relatives au moniteur.
- **Delete Monitor (Supprimer le moniteur)** : permet de supprimer un moniteur du tableau de bord.
- **Run Monitor Immediately (Exécuter le moniteur immédiatement)** : permet d'exécuter le moniteur immédiatement.
- **Deactivate (Désactiver)** : permet de désactiver le moniteur. Cela signifie qu'il est toujours inclus dans la liste, mais qu'il ne s'exécute pas. Vous pouvez également cliquer sur ce bouton pour réactiver un moniteur préalablement désactivé.
- **View Results (Afficher les résultats)** : permet d'afficher les résultats de l'exécution dans la visionneuse Report Viewer (Visionneuse de rapports).

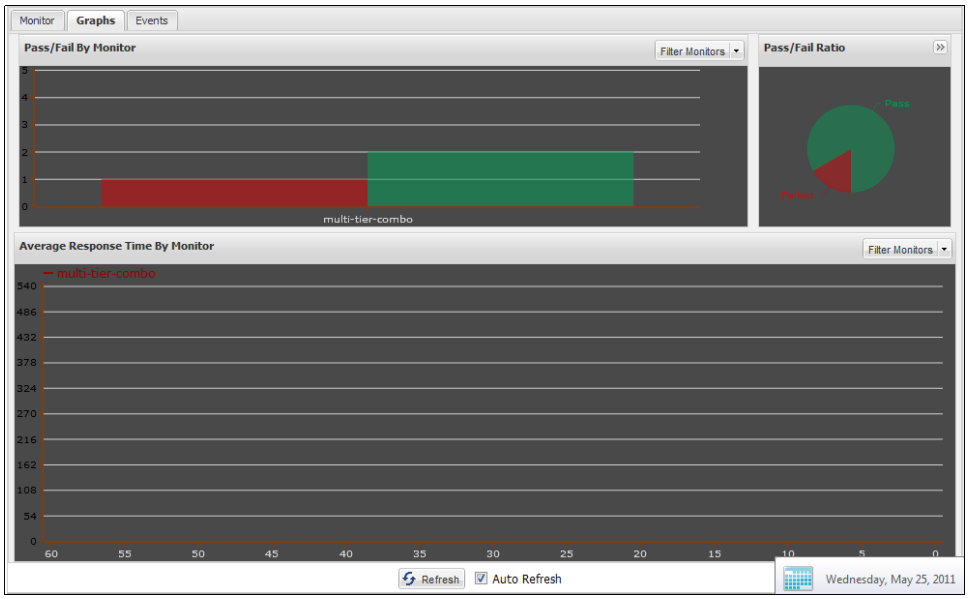
Lorsque vous exécutez le tableau de bord CVS Dashboard (Tableau de bord du service CVS) directement à partir de DevTest Workstation (en sélectionnant l'affichage du tableau de bord à partir de DevTest), les boutons suivants s'affichent également :

-  Actualise la liste dans le tableau de bord.
-  Actualise automatiquement la liste dans le tableau de bord après une certaine période.

Lorsque vous exécutez le tableau de bord CVS Dashboard (Tableau de bord du service CVS) directement à partir de votre navigateur, ces boutons ne s'affichent pas.

Onglet Graphs (Graphiques) du tableau de bord CVS Dashboard

L'onglet Graphs (Graphiques) affiche les tests figurant dans le tableau de bord CVS Dashboard (Tableau de bord du service CVS) sous la forme de graphiques. Cet onglet fournit un récapitulatif graphique des tests réussis et ayant échoué.



L'onglet Graphs consiste en trois affichages graphiques, qui affichent et suivent les 60 dernières minutes d'activité.

Le moniteur de filtre vous permet de sélectionner les moniteurs que vous voulez afficher.

Procédez comme suit:

1. Cliquez sur Filter Monitors (Moniteurs de filtre) en haut de la fenêtre.

   Une liste de moniteurs disponibles s'ouvre.

2. Sélectionnez l'onglet à vérifier.

   Les détails s'affichent dans les sections suivantes :

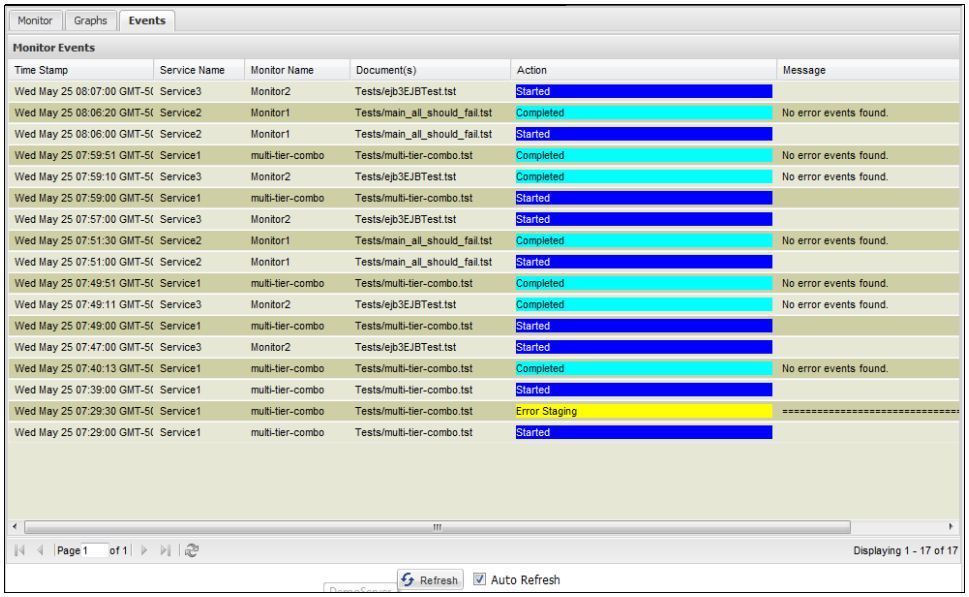
   - Pass/Fail by Monitor (Réussite/Echec par moniteur) **:** le graphique Pass/Fail by Monitor affiche des histogrammes empilés de résultats de réussite et d'échec pour chaque moniteur. Déplacer le curseur sur l'histogramme affiche le résultat sous la forme d'un texte dans le panneau Pass/Fail By Monitor supérieur.
   - Pass/Fail Ratio (Taux de réussite/d'échec) **:** le graphique Pass/Fail Ratio affiche le pourcentage du nombre total de tests réussis (en vert) et le nombre total de tests ayant échoué (en rouge) dans un graphique à secteurs.

- Average Response Time by Monitor (Temps moyen de réponse par moniteur) **:** le graphique Average Response Time by Monitor représente le temps de réponse moyen pour chaque moniteur au cours des 60 dernières minutes d'activité. Un code couleur est appliqué à chaque moniteur. Passez la souris sur un moniteur pour afficher le temps de réponse moyen dans une info-bulle.

Vous pouvez afficher le test ou la suite sous la forme d'un graphique Pass/Fail by Monitor (Réussite/Echec par moniteur).

Onglet Events (Evénements) du tableau de bord CVS Dashboard

L'onglet Events (Evénements) affiche différents événements correspondant aux étapes de l'exécution de moniteur, comme le démarrage, la fin et l'échec d'une exécution de test.



Les informations suivantes s'affichent :

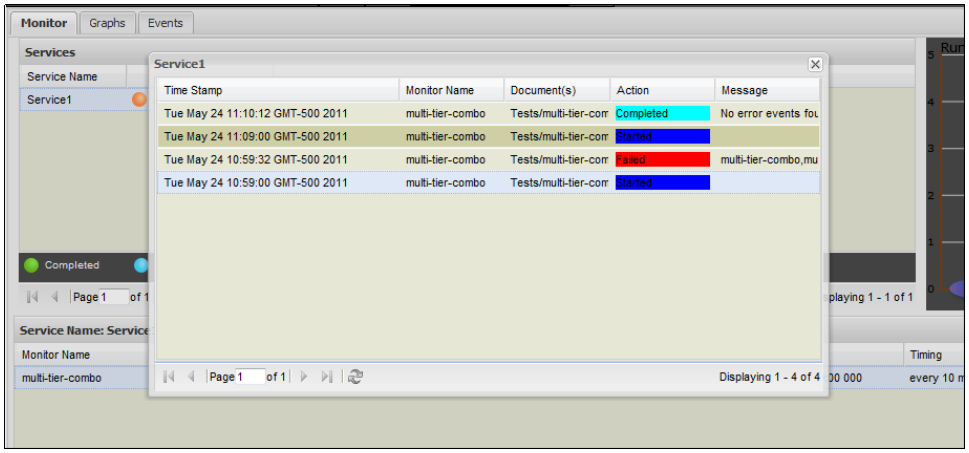
- Time Stamp (Horodatage) **:** heure à laquelle l'événement s'est produit.
- Service Name (Nom de service) **:** nom du service dans lequel le moniteur réside.
- Monitor Name (Nom du moniteur) **:** nom du moniteur dans lequel l'événement s'est produit.
- Document(s) **:** liste des documents associés au moniteur pour lequel l'événement s'est produit.
- Action **:** action signalée par l'événement. Les événements de démarrage sont bleus, les événements d'erreur de simulation sont jaunes, les événements de fin sont verts et les événements d'échec sont rouges.
- Message **:** message rapporté par l'événement. Si le texte dépasse la cellule de message, vous pouvez cliquer avec le bouton droit de la souris sur la cellule pour appeler une fenêtre de vue étendue.

Vous pouvez afficher les événements en temps réel en cochant la case Auto Refresh (Actualisation automatique) en bas du panneau ou vous pouvez actualiser la liste manuellement en cliquant sur l'icône Refresh (Actualiser) si la case Auto Refresh est décochée.

Vous pouvez ajuster les tailles des colonnes pour qu'elles puissent toutes s'afficher dans la fenêtre. Si vous double-cliquez sur un champ Message, vous pouvez afficher le texte complet du message.

Cycle de test avec des erreurs

Parfois, le cycle de test échoue. Le cycle qui comprend des erreurs peut être affiché dans la liste Actions dans une couleur différente. Vous pouvez double-cliquer sur un événement pour consulter les messages associés dans une fenêtre développée.



Déploiement d'un moniteur vers le service de validation en continu

Les rubriques suivantes sont disponibles.

Déploiement d'un moniteur via DevTest Workstation (page 397)
Déploiement d'un moniteur via le tableau de bord CVS Dashboard (page 397)
Déploiement d'un moniteur via le répertoire cvsMonitors (page 397)
Déploiement d'un moniteur via le gestionnaire CVS Manager (page 398)

Déploiement d'un moniteur via DevTest Workstation

La seule condition préalable à cette approche est l'existence d'un scénario de test ou d'une suite.

Procédez comme suit:

1. A partir du panneau Project (Projet) à gauche dans DevTest Workstation, cliquez avec le bouton droit de la souris sur un scénario de test ou une suite, puis sélectionnez Deploy as monitor to CVS (Déployer en tant que moniteur vers le service CVS).

   Un ensemble d'onglets s'affichent avec les informations relatives au moniteur. Pour en savoir plus ces informations, reportez-vous à la section Création de fichiers d'informations MAR de moniteur (page 284) de la rubrique *Utilisation de CA Application Test.*

2. Cliquez sur Deploy (Déployer) pour déployer le moniteur.

Déploiement d'un moniteur via le tableau de bord CVS Dashboard

Avant de pouvoir effectuer cette procédure, une archive de modèle doit être créée (page 288) pour le moniteur.

Procédez comme suit:

1. Dans l'onglet Monitor (Moniteur) du tableau de bord CVS Dashboard (Tableau de bord du service CVS), cliquez sur Re/Deploy Monitor (Redéployer/Déployer un moniteur).

   La fenêtre Re/Deploy Monitor s'ouvre pour le nouveau moniteur.

2. Entrez le nom du fichier MAR dans le champ Model Archive (Archive de modèle).

3. Si le fichier MAR a préalablement été déployé, cochez ou décochez la case Replace the monitor if it exists (Remplacer le moniteur s'il existe).

4. Cliquez sur Re/Deploy (Redéployer/déployer).

   Le moniteur est ajouté au tableau de bord CVS Dashboard.

Déploiement d'un moniteur via le répertoire cvsMonitors

Avant de pouvoir effectuer cette procédure, une archive de modèle doit être créée (page 288) pour le moniteur.

Procédez comme suit:

1. Accédez au répertoire **LISA_HOME\cvsMonitors**.

2. Ajoutez le fichier MAR au répertoire.

   Vous pouvez mettre à jour le moniteur ultérieurement en plaçant une nouvelle version du fichier MAR dans ce répertoire.

Déploiement d'un moniteur via le gestionnaire CVS Manager

Pour plus de détails sur l'option de ligne de commande permettant de déployer un moniteur de service de validation en continu, consultez la rubrique Gestionnaire CVS Manager (page 401).

Exécution immédiate d'un moniteur

Lorsque vous ajoutez les services dans le panneau supérieur, ils sont exécutés automatiquement en arrière-plan selon les intervalles planifiés.

Si vous voulez exécuter un moniteur immédiatement, vous pouvez utiliser la barre d'outils.

Procédez comme suit:

1. Déployez le moniteur.
2. Lorsqu'il est répertorié dans le panneau inférieur, sélectionnez-le.
3. Cliquez sur Run Monitor Immediately (Exécuter le moniteur immédiatement) pour exécuter le moniteur immédiatement.

   L'exécution du moniteur n'est pas affichée dans le panneau inférieur.
4. Pour afficher le moniteur exécuté, accédez à l'onglet Events (Evénements) du tableau de bord CVS Dashboard (Tableau de bord du service CVS) avec un suffixe **-now** (par exemple, **Test 3-now**).

Affichage des détails du test

Vous pouvez afficher les détails d'une exécution de test dans le tableau de bord CVS Dashboard (Tableau de bord du service CVS).

Procédez comme suit:

Effectuez l'une des opérations suivantes :

- Double-cliquez sur le service pour afficher ses détails.
- Double-cliquez sur le moniteur dont vous voulez afficher les détails dans le panneau supérieur.

La fenêtre de détails relatifs au test s'ouvre.

Vous pouvez également afficher l'onglet Events (Evénements) dans le tableau de bord CVS Dashboard. Vous pouvez ajuster les tailles des colonnes pour qu'elles puissent toutes s'afficher. Pour afficher le texte du message entier, double-cliquez sur le champ Message.

Tous les rapports générés pendant les exécutions de test planifiées sont disponibles dans la visionneuse Report Viewer (Visionneuse de rapports).

Paramètres de notification par courriel

Pour chaque nouvelle planification de test que vous effectuez dans l'utilitaire CVS, déployez un moniteur dans le tableau de bord CVS Dashboard (Tableau de bord du service CVS).

Dans la fenêtre du moniteur, vous pouvez également définir une adresse électronique. Si vous définissez une adresse électronique, vous recevez une notification par courriel à chaque exécution du moniteur dans la période spécifiée.

Pour définir la notification par courriel, mettez à jour le fichier **lisa.properties.**

Vous devez également modifier la configuration de serveur de messagerie et entrer l'hôte de messagerie, comme dans l'exemple suivant.

```
# Si vous utilisez des alertes de surveillance de performances,
l'adresse électronique suivante est celle de l'expéditeur de ces
alertes.
# lisa.alert.email.emailAddr=lisa@itko.com

# Le serveur de messagerie suivant est le serveur utilisé pour le
routage des courriels (serveur SMTP).
# lisa.alert.email.defHosts=localhost
```

Pour annuler la mise en commentaire de ces informations dans le fichier **lisa.properties**, supprimez le caractère **#** de la première colonne du fichier.

Redémarrez DevTest pour appliquer la configuration du serveur de messagerie.

- Notification Email (Courriel de notification) **:** entrez l'adresse électronique pour la notification des résultats du test par courriel.

A chaque exécution du test, vous recevez un courriel.

Gestionnaire CVS Manager

L'utilitaire de ligne de commande CVS Manager (Gestionnaire de services de validation en continu) vous permet de gérer l'ensemble des moniteurs déployés sur le service Continuous Validation Service (page 385) (Service de validation en continu). Cet utilitaire se trouve dans le répertoire **LISA_HOME\bin.**

L'utilitaire est au format suivant :

```
CVSManager [-h] [-m registry-spec] [-d archive-file] [-r archive-file] [-l] [-D]
[-A] [-e] [x] [-X] [-s name] [-n name] [-u username] [-p password] [--version]
```

-h, --help

Affiche le texte d'aide.

-m *registry-spec*, --registry=*registry-spec*

Définit le registre auquel se connecter.

-d *archive-file*, --deploy=*archive-file*

Déploie l'archive de modèle spécifiée sur le service de validation en continu en tant que moniteur. Le moniteur défini dans l'archive doit faire référence à une combinaison de nom de moniteur et de service qui n'existe pas.

-r *archive-file*, --redeploy=*archive-file*

Redéploie l'archive de modèle spécifiée sur le service de validation en continu en tant que moniteur. Le moniteur défini dans l'archive doit faire référence à une combinaison de nom de moniteur et de service existante.

-l, --list

Répertorie les moniteurs actuellement déployés avec des informations sur chaque moniteur.

-D, --pause

Interrompt l'exécution planifiée du moniteur indiqué.

-A, --resume

Reprend l'exécution planifiée du moniteur indiqué.

-e, --execute-now

Le moniteur indiqué est immédiatement exécuté, indépendamment de sa planification. Cette action n'affecte aucune exécution planifiée du moniteur.

-x, --remove

Supprime un moniteur du service de validation en continu. Utilisez les arguments de nom de service et de nom de moniteur pour indiquer le moniteur à supprimer.

-X, --remove-all

Supprime tous les moniteurs du service de validation en continu. Si un nom de service est spécifié, seuls les moniteurs portant ce nom de service sont supprimés.

-s *name*, --service-name=*name*

Spécifie le nom de service pour les moniteurs à affecter.

-n *name*, --monitor-name=*name*

Spécifie le nom du moniteur à affecter.

-u *username*, --username=*username*

Spécifie le nom de sécurité DevTest de l'utilisateur. Cette option est obligatoire.

-p *password*, --password=*password*

Spécifie le mot de passe de sécurité DevTest de l'utilisateur. Cette option est obligatoire.

--version

Imprimez le numéro de version et quittez l'utilitaire.

Exemple de déploiement de moniteur

Dans cet exemple, un moniteur est déployé vers le service de validation en continu.

```
CVSManager -d monitor.mar -u user -p password
```

Exemple de suppression de moniteur

Dans cet exemple, le même moniteur est supprimé (noms de service et de moniteur identiques).

```
CVSManager -x -s OrderManager -n CheckOrders -u user -p password
```

Dans cet exemple, tous les moniteurs du service OrderManager sont supprimés.

```
CVSManager -X -s OrderManager -u user -p password
```

Dans cet exemple, tous les moniteurs sont supprimés.

```
CVSManager -X -u user -p password
```

Chapitre 16: Rapports

Vous déterminez les données collectées pour les rapports en spécifiant des paramètres de génération de rapports dans l'un des domaines suivants :

- Tests rapides (page 306)
- Documents de simulation (page 235)
- Suites de tests (page 265)

Vous pouvez spécifier des événements ou des mesures spécifiques à collecter pour chaque scénario de test ou chaque suite de tests que vous exécutez. Sélectionnez un générateur de rapports pour stocker les données de rapports dans une base de données ou dans un fichier XML. Déterminez également la durée de conservation des données de rapports.

Une fois que les rapports ont été générés, vous pouvez les afficher et les gérer ultérieurement, les partager avec des collègues ou les exporter vers d'autres emplacements.

Ce chapitre traite des sujets suivants :

Types de générateur de rapports

Vous pouvez sélectionner les types de générateurs de rapports suivants dans l'onglet Reports (Rapports) de l'éditeur Staging Document Editor (page 237) (Editeur de documents de simulation) ou Test Suite Editor (page 266) (Editeur de suites de tests).

- Default Report Generator (Générateur de rapports par défaut) (page 404)
- Load Test Report Generator (Générateur de rapports de test de charge) (page 405)
- XML Report Generator (Générateur de rapports XML) (page 405)

Default Report Generator (Générateur de rapports par défaut)

Le générateur de rapports Default Report Generator (Générateur de rapports par défaut) capture des informations fonctionnelles et de mesures, et publie ces données dans la base de données de rapports référencée par le registre. Le portail de rapports utilise la base de données de rapports.

Ce créateur de rapports n'est pas pris en charge pour le test de charge ou le test des performances. Pour le test de charge, utilisez le générateur de rapports Load Test report generator (Générateur de rapports de test de charge).

Remarque : Si ce générateur de rapports est sélectionné, CA Application Test ne configurera pas automatiquement l'analyse de charge.

Chapitre 17: Load Test Report Generator (Générateur de rapports de test de charge)

Le générateur de rapports Load Test Report Generator (Générateur de rapports de test de charge) est conçu pour les tests de charge comptant des milliers d'utilisateurs virtuels.

Ce rapport capture des mesures de charge, mais ignore les mesures de niveau d'étape. Si les mesures de niveau d'étape étaient capturées, il y aurait trop de données et la base de données de rapports ralentirait le test.

Presque tous les événements sont désactivés. En conséquence, la console Reporting Console (Console de génération de rapports) contient peu d'informations pour l'exécution. Les informations principales obtenues se trouvent dans le panneau Test Run (Exécution de test) de DevTest Workstation.

Dans la zone Parameters (Paramètres), vous pouvez configurer le nombre d'erreurs après lequel le test s'arrête automatiquement. La valeur par défaut est 100.

Le générateur de rapports Default Report Generator (Générateur de rapports par défaut) n'est pas pris en charge pour le test de charge et des performances.

XML Report Generator (Générateur de rapports XML)

Ce générateur de rapports crée un fichier XML contenant toutes les données que vous pouvez capturer. Vous pouvez limiter les données capturées à l'aide des options de rapport dans l'éditeur Test Suite Editor (Editeur de suites de tests) ou Staging Document Editor (Editeur de documents de simulation). Pour afficher ce rapport, importez le fichier dans le portail Reporting Portal (Portail de rapports).

Vous pouvez utiliser les données de cette table pour vos besoins en rapports personnalisés.

Une fois que les tests, les suites ou les deux sont terminés, vous pouvez afficher les données de rapport dans la console de génération de rapports. Pour exporter les données XML dans un fichier, consultez la rubrique Exportation de rapports (page 442).

Ouverture du portail de rapports

Vous pouvez ouvrir le portail Reporting Portal (Portail de rapports) à partir de DevTest Workstation ou à partir d'un navigateur Web.

Pour ouvrir le portail Reporting Portal à partir de DevTest Workstation :

Dans le menu principal, sélectionnez View (Afficher), Reporting Console (Console de génération de rapports).

Pour ouvrir le portail Reporting Portal à partir d'un navigateur Web :

1. Vérifiez que le registre (page 11) est en cours d'exécution.
2. Dans un navigateur Web, entrez **http://localhost:1505/**.

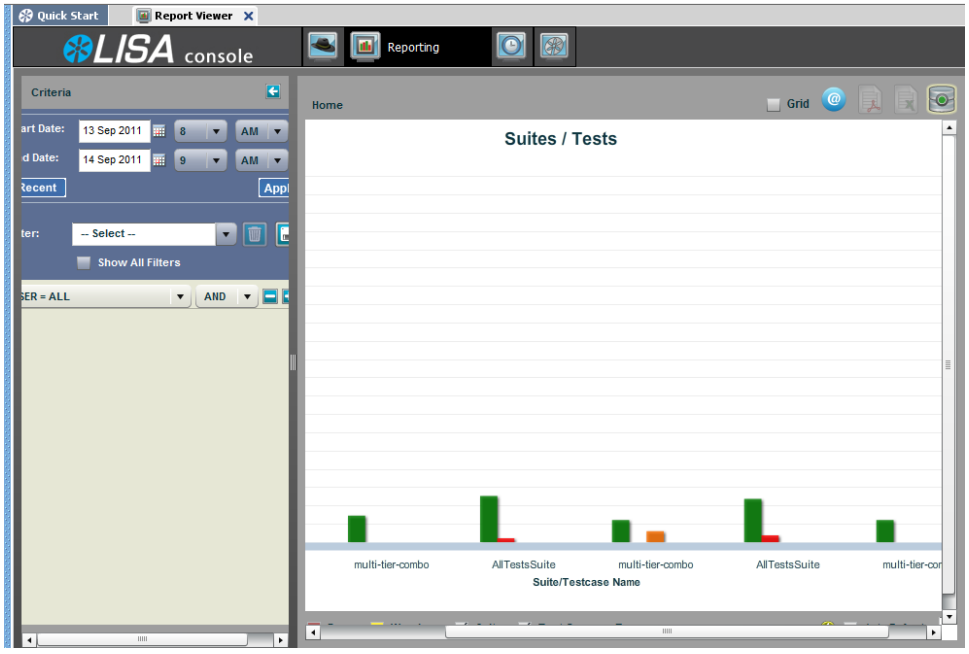
   Si le registre se trouve sur un ordinateur distant, remplacez **localhost** par le nom ou l'adresse IP de l'ordinateur.

   La console DevTest s'ouvre.
3. Cliquez sur Reporting Console (Console de génération de rapports).

Disposition du portail de rapports

Le portail Reporting Portal (Portail de rapports) vous permet d'afficher tous les rapports exécutés précédemment dans DevTest Workstation. Vous pouvez configurer les rapports à l'aide des documents de simulation, des tests rapides, des scénarios de test en cours d'exécution ou des suites de tests.

Cette section traite des rapports générés en exécutant le scénario de test à plusieurs niveaux, qui se trouve dans le répertoire Examples (Exemples).



Portail de rapports - Critères

Dans le panneau gauche, vous pouvez sélectionner les critères de date et heure, ainsi que les filtres à utiliser.

Start Date (Date de début)/End Date (Date de fin)

Sélectionnez les dates de début et de fin en cliquant sur Calendar (Calendrier)  . Par défaut, les dates de début et de fin sont comprises dans la plage de la dernière heure. Sélectionnez l'heure de début et de fin en cliquant sur les listes déroulantes 1-12 et AM/PM. Après avoir entré ces dates, cliquez sur Apply (Appliquer) pour appliquer vos modifications.

Recent (Récent)

Pour réinitialiser les critères de date et d'heure et rechercher automatiquement la dernière suite ou le dernier test exécuté et l'heure qui la ou le précède, cliquez sur ce bouton.

Filter (Filtre)

Cette option vous permet de créer des filtres de votre choix. Saisissez le nom du filtre, puis cliquez sur Save (Enregistrer). Vous pouvez également supprimer un filtre en cliquant sur Delete (Supprimer). Pour afficher tous les filtres, y compris ceux que vous avez créés, sélectionnez l'option Show All Filters (Afficher tous les filtres).

Sélectionnez un opérateur AND/OR pour les critères et cliquez sur Add  (Ajouter) ou Delete  (Supprimer).

Portail de rapports - Panneau droit

Le panneau droit contient les graphiques générés selon les critères sélectionnés. Pour plus d'informations, consultez la rubrique Vue graphique des rapports (page 413).

La barre d'outils de rapports s'affiche en haut du panneau droit. Les icônes qu'elle contient et leurs fonctions sont détaillées dans le tableau suivant.

Icône	Fonction
	Permet de modifier la vue de rapport de la vue graphique par défaut à la vue de grille, et inversement. Pour plus d'informations, consultez la rubrique Vue de grille des rapports (page 429).

	Permet de copier l'URL du rapport affiché dans le presse-papiers du système d'exploitation pour que vous puissiez partager le rapport avec d'autres utilisateurs.
	Permet d'exporter les données de rapport dans un fichier PDF ou Excel, afin d'afficher le rapport dans ces formats. Pour plus d'informations, consultez la rubrique Exportation de rapports (page 442).
	Permet d'afficher des informations sur la base de données de rapport. LISA Database Connection Successful --- DBUrl: jdbc:derby://localhost:1528/database/lisa.db;create=true Username: rpt Database: Apache Derby Version: 10.6.2.1 - (999685) Driver: Apache Derby Network Client JDBC Driver Version: 10.6.2.1 - (999685) Suite Runs: 4 Test Runs: 71

Vous pouvez filtrer les rapports selon les scénarios de test réussis ou ayant échoué. Vous pouvez également afficher ou masquer les erreurs et les avertissements, ainsi que les suites de tests ou les scénarios de test. Pour plus d'informations, consultez la rubrique Filtrage des rapports (page 410).

Le curseur Zoom vous permet de personnaliser la taille de l'affichage du rapport.

Le bouton  Refresh (Actualiser) vous permet d'actualiser l'affichage. Pour définir une actualisation automatique, cochez la case Auto Refresh (Actualisation automatique) et définissez un intervalle d'actualisation automatique.

Filtrage des rapports

Le panneau droit du portail Reporting Portal (page 407) (Portail de rapports) affiche les rapports.

Vous pouvez filtrer les rapports selon différents critères. Une fois que vous avez sélectionné les critères, la visionneuse de rapports affiche des graphiques pour les critères sélectionnés uniquement.


Pass (Réussite)

Permet d'afficher les scénarios de test/suites réussis.


Fail (Echec)

Permet d'afficher les scénarios de test/suites ayant échoué.


Aborted (Interrompu)

Permet d'afficher les scénarios de test/suites interrompus.


Errors (Erreurs)

Permet d'afficher les scénarios de test/suites ayant généré des erreurs.


Warnings (Avertissements)

Permet d'afficher les scénarios de test/suites ayant généré des avertissements.

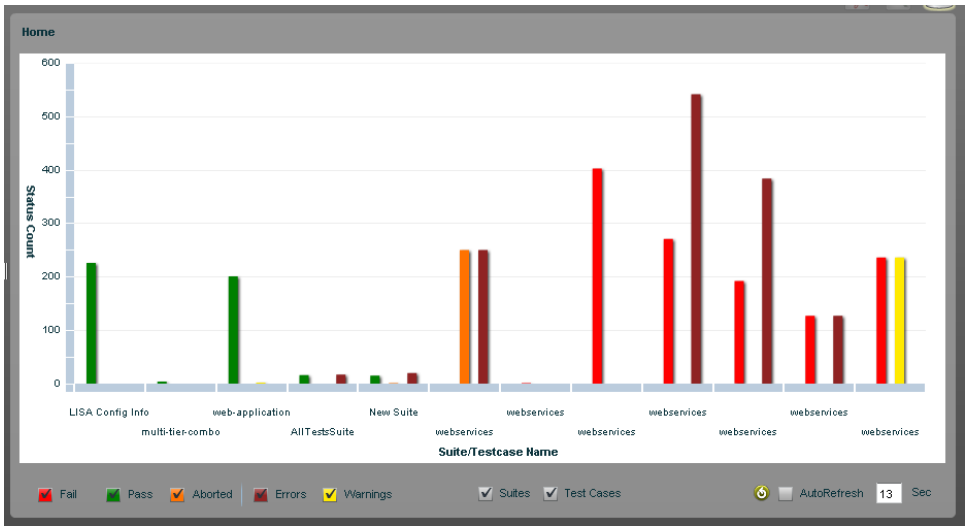

Suites

Affiche les résultats de la suite de tests.


Test Cases (Scénarios de test)

Affiche les résultats du scénario de test.

Le portail Reporting Portal (Portail de rapports) répertorie les rapports pour tous les scénarios de test et les suites de tests exécutés qui figurent actuellement dans sa base de données.



Dans le rapport précédent, aucun critère de recherche n'est spécifié, car tous les filtres sont sélectionnés. Par conséquent, le rapport affiche tous les scénarios de test et les suites de tests réussis, ayant échoué, interrompu, avec des erreurs et avec des avertissements.

Remarque : Pour une spécificité approfondie, vous pouvez combiner les critères de **résultat** avec les critères **d'exécution de test**. Par exemple, vous pouvez sélectionner Fail (Echec), Error (Erreur) et Test Case (Scénario de test) pour afficher uniquement les scénarios de test ayant échoué ou ceux terminés avec des erreurs.

Pour actualiser les rapports :

Cliquez sur Refresh (Actualiser) .

Pour actualiser les rapports automatiquement :

1. Cochez la case Auto Refresh (Actualisation automatique).
2. Entrez le délai en secondes après lequel les rapports sont actualisés.

   Par exemple, **15** actualise le rapport toutes les 15 secondes.

Affichage des rapports

Dans la visionneuse Report Viewer (Visionneuse de rapports), vous pouvez afficher les rapports dans deux formats :

- Graphical View (Vue graphique) (page 413): vue par défaut du portail Reporting Portal (Portail de rapports). Tous les rapports sont affichés sous la forme de graphiques.
- Grid View (Vue de grille) (page 429): vous pouvez sélectionner la vue de grille pour organiser les données dans un format de grille.

Informations complémentaires :

Rapports standard (page 431)
Interprétation des rapports (page 441)

Vue graphique des rapports

Par défaut, les rapports s'affichent sous la forme de graphiques.

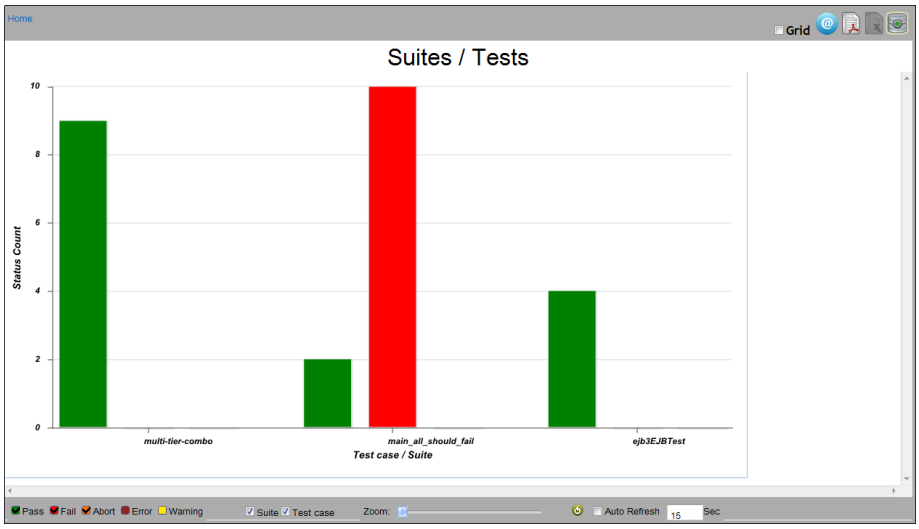
L'exemple de rapport suivant contient trois scénarios de test :

- multi-tier-combo
- main_all_should_fail
- ejb3EJBTest

Le rapport affiche uniquement les scénarios de test réussis, ayant échoué ou interrompus, car toutes les cases à cocher de critères de filtrage ne sont pas sélectionnées.

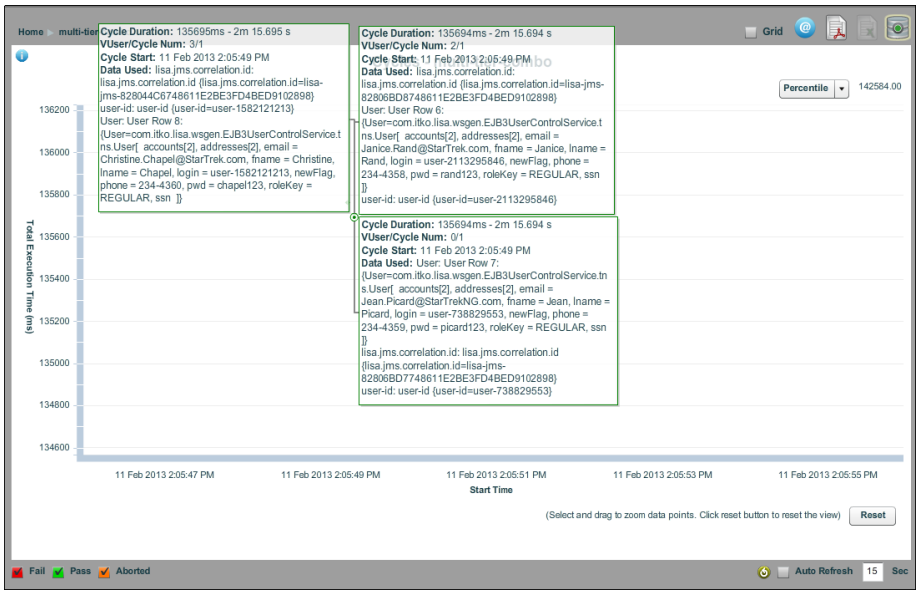
Les fonctionnalités suivantes sont disponibles dans la vue graphique :

- Passer la souris sur un scénario de test ouvre une zone d'informations qui indique le nom du scénario de test, la date d'exécution et les détails d'exécution du test.



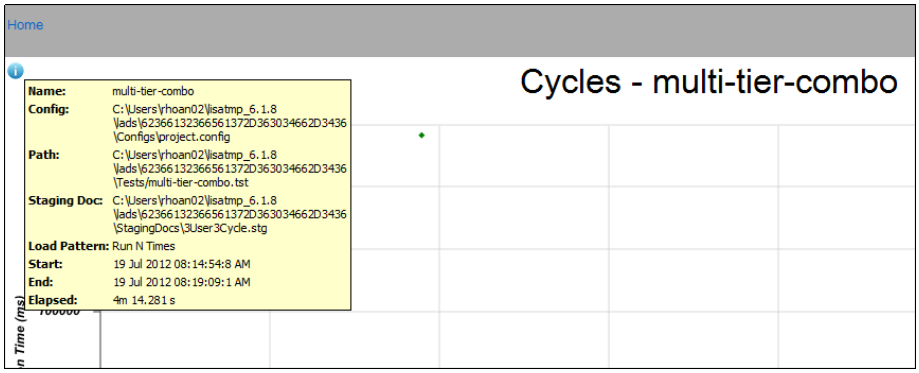
- Double-cliquer sur un scénario de test réussi dans le graphique génère un diagramme de cycles pour ce scénario.

  Lorsque vous affichez le diagramme de cycles en vue graphique, vous pouvez sélectionner des options dans la liste déroulante Centile afin d'indiquer la valeur de centile à afficher sous forme graphique (une ligne dans le diagramme). La valeur de centile indique le nombre maximum de millisecondes pendant lesquelles un certain pourcentage d'exécution d'étape a été effectué. Par exemple, si le 75ème cycle le plus long de l'étape A s'est terminé en 7,9 secondes, le 75ème centile serait 7 900 ou une valeur supérieure.

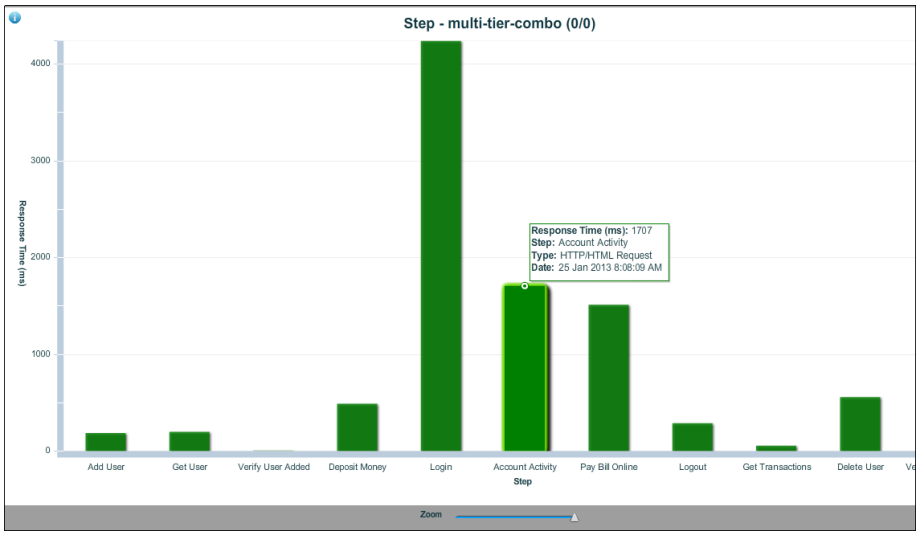


Lorsque vous placez votre souris sur ces points, vous constatez que chaque point représente un cycle du scénario de test, et les détails de chaque cycle sont affichés : temps de réponse en millisecondes, nombre de cycle/instance, date du cycle et données utilisées. Un cycle représente un scénario de test complet, du début à la fin.

- Cliquer sur Information dans la partie gauche supérieure de la fenêtre vous donne accès à des informations complémentaires sur l'environnement de scénario de test.



- Pour examiner un sous-ensemble de cycles de plus près, sélectionnez un groupe de points et zoomez de façon à afficher uniquement ce sous-ensemble.
- Pour retourner à la vue de cycle complète, cliquez sur Reset (Réinitialiser).
- Double-cliquer sur un scénario de test affiche une vue de chaque étape du scénario, avec les informations sur les temps de réponse.



- Les rapports affichés sous la forme de graphiques linéaires peuvent ajuster automatiquement leur échelle. Le paramètre par défaut pour l'option Current Scale (Echelle actuelle) est un ajustement automatique de l'échelle à 100. Le nombre entre parenthèses indique la valeur requise par la mise à l'échelle automatique pour ajuster toutes les données sur le même graphique 0 - 100. Dans le graphique : valeur * échelle = coordonnée y. Si vous modifiez l'échelle, la valeur actuelle ne change pas. Toutefois, le calcul permettant de déterminer la coordonnée y utilisée dans le graphique peut renvoyer une coordonnée y différente. Si la coordonnée est supérieure à 100, les limites de l'axe Y doivent être étendues, afin de prendre en compte de plus grandes valeurs.

Menu Reporting (Rapports) des rapports

Lorsque vous cliquez avec le bouton droit de la souris sur un scénario de test ou une suite, les options suivantes sont disponibles. Pour afficher des exemples de rapports, consultez la rubrique Exemples de vue graphique de rapports (page 417).

Analyze (Analyser)	■ **Top Ten Longest Transactions (10 transactions les plus longues) :** les dix transactions en cours d'exécution le plus longtemps dans un graphique à secteurs ■ **Average Transaction Response (Réponse de transaction moyenne) :** temps de réponse de transaction dans un graphique à courbes ■ **Metrics (Mesures) :** mesures d'événement DevTest dans un graphique à courbes par heure ■ **Requests/Second (Demandes par seconde) :** nombre de demandes par seconde dans un graphique à courbes par heure ■ **Performance Summary (Récapitulatif des performances) :** temps de réponse moyen et écart type par étape dans un graphique à barres ■ **Cycle Performance Summary (Récapitulatif des performances du cycle) :** durée du cycle et temps d'exécution du cycle en millisecondes dans un graphique à barres ■ **HTTP Details (Détails HTTP) :** détails du trafic HTTP sous forme de grille par nom ■ **HTTP Summary (Récapitulatif HTTP) :** récapitulatif du trafic HTTP cumulatif dans un graphique à courbes
View Error Reports (Afficher les rapports d'erreur)	■ Detailed Failures (Echecs détaillés) ■ Detailed Errors (Erreurs détaillées) ■ Detailed Warnings (Avertissements détaillés) ■ Detailed Aborts (Interruptions détaillées)
View Launch Properties (Afficher les propriétés de lancement)	Permet d'afficher toutes les propriétés de lancement avec un horodatage, le nom de propriété et la valeur de la propriété sous forme de grille.
View History (Afficher l'historique)	Permet de réduire le graphique actuel à la moitié supérieure de la fenêtre et d'afficher deux rapports supplémentaires : ■ Une liste de grille de scénarios de test exécutés et leurs résultats ■ Un graphique à courbes indiquant un historique des heures d'exécution Le rapport d'historique affiche des informations sur les exécutions précédentes du même scénario de test.
Supprimer	Permet de supprimer du rapport la suite ou le scénario de test sélectionné.
Pin (Epingler)	Permet d'éviter la suppression automatique du scénario de test ou de la suite après 30 jours. Les tests désépinglés antérieurs à 30 jours sont supprimés automatiquement. Pour plus d'informations, consultez la rubrique Maintenance de la génération de rapports automatique.

Import (Importer)	Permet d'importer les informations du scénario de test ou de la suite à partir d'un fichier XML.
Export (Exporter)	Permet d'exporter les informations du scénario de test ou de la suite dans un fichier XML.
Save Image (Enregistrer l'image)	Permet d'enregistrer l'image en tant que fichier .png. Cliquez avec le bouton droit de la souris dans la barre d'étape pour afficher un menu d'étape.

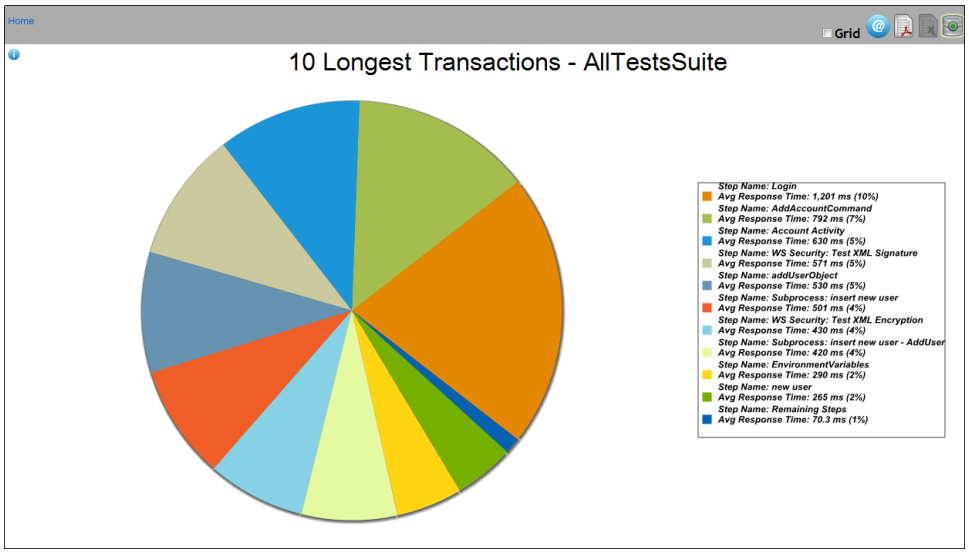
Exemples de vue graphique des rapports

Cette section fournit des exemples pour les options de menu de génération de rapports suivantes :

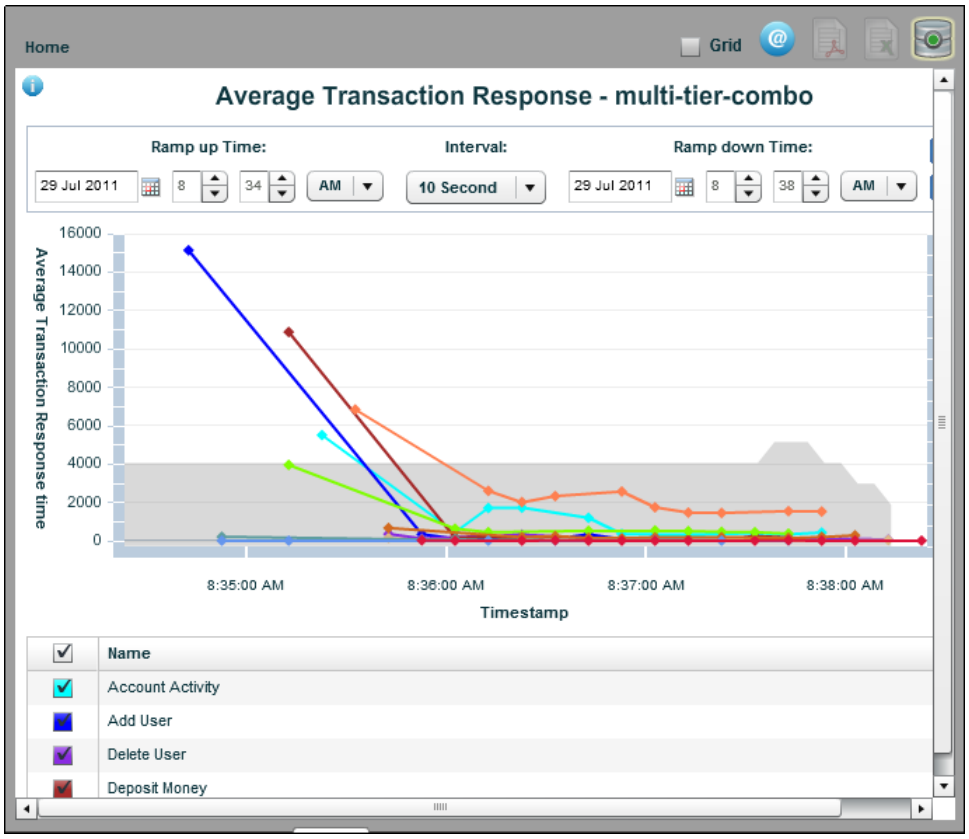
- Analyze (Analyser) (page 418)
- View Error Reports (Afficher les rapports d'erreur) (page 424)
- View History (Afficher l'historique) (page 428)

Exemples d'analyse de rapport

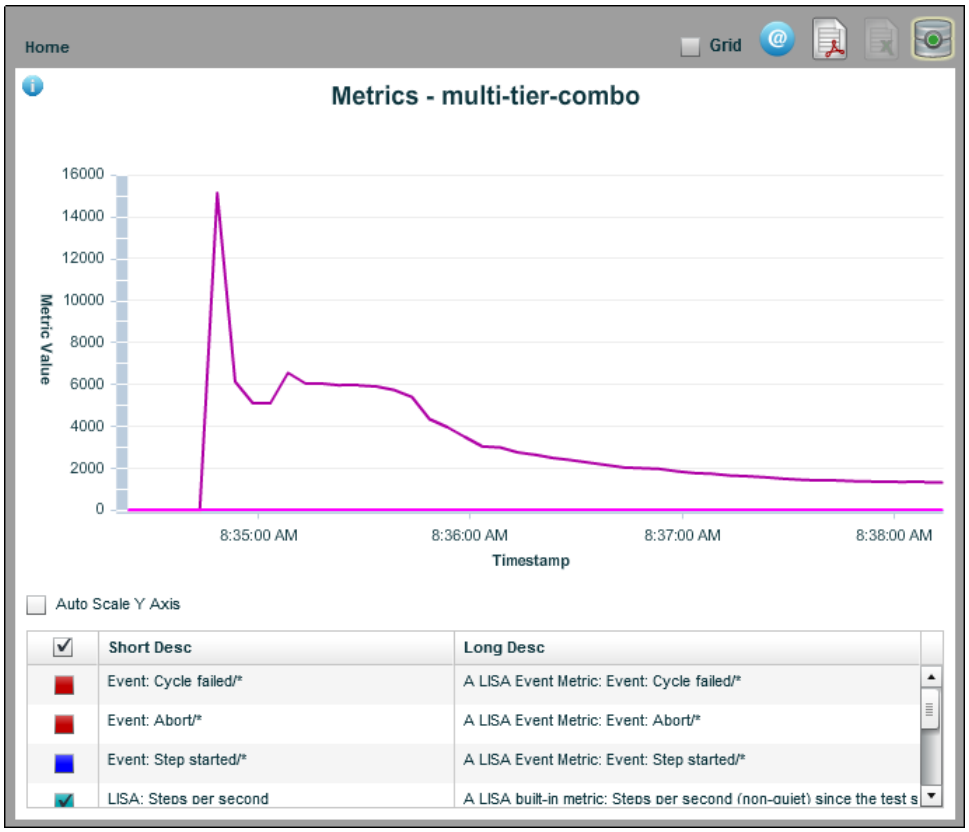
Top Ten Longest Transactions (10 transactions les plus longues)



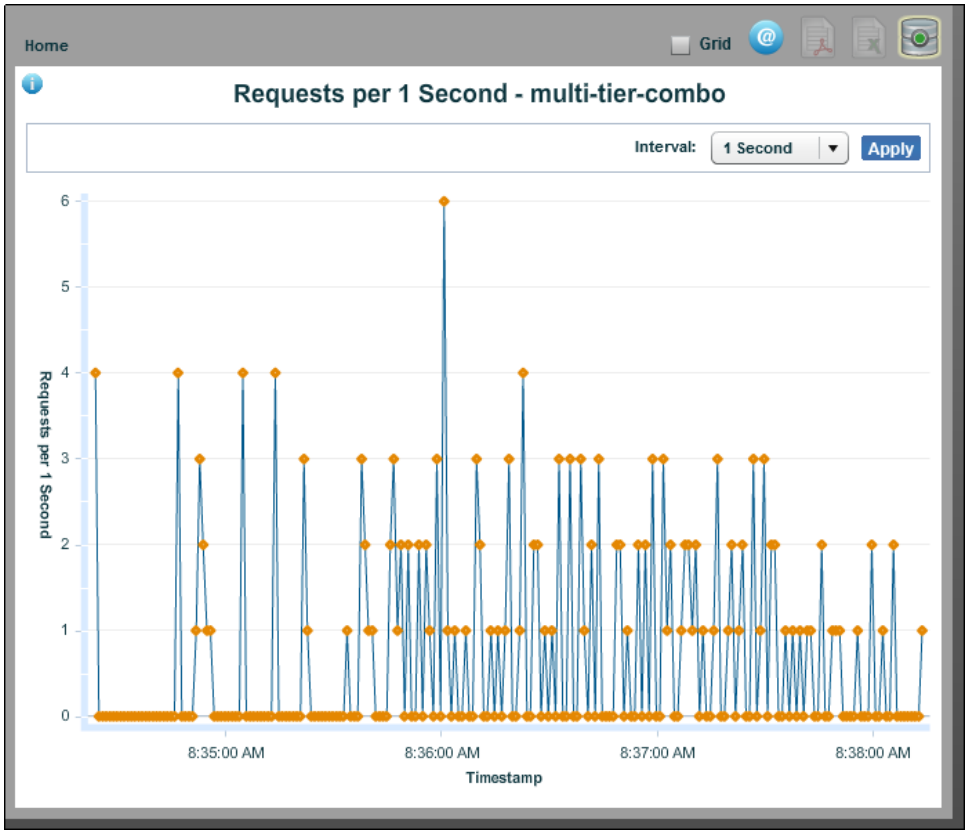
Average Transaction Response (Réponse de transaction moyenne)



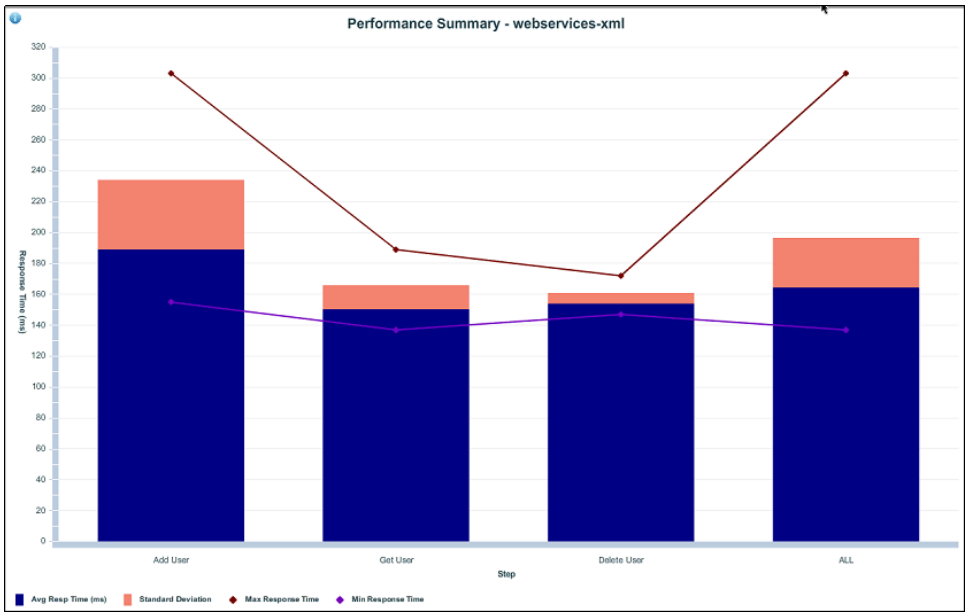
Metrics (Mesures)



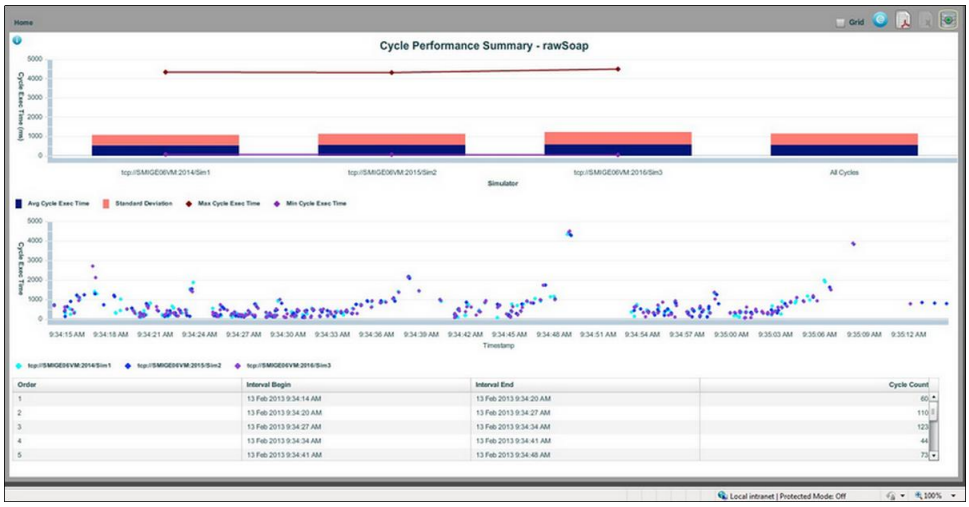
Requests/Second (Demandes par seconde)



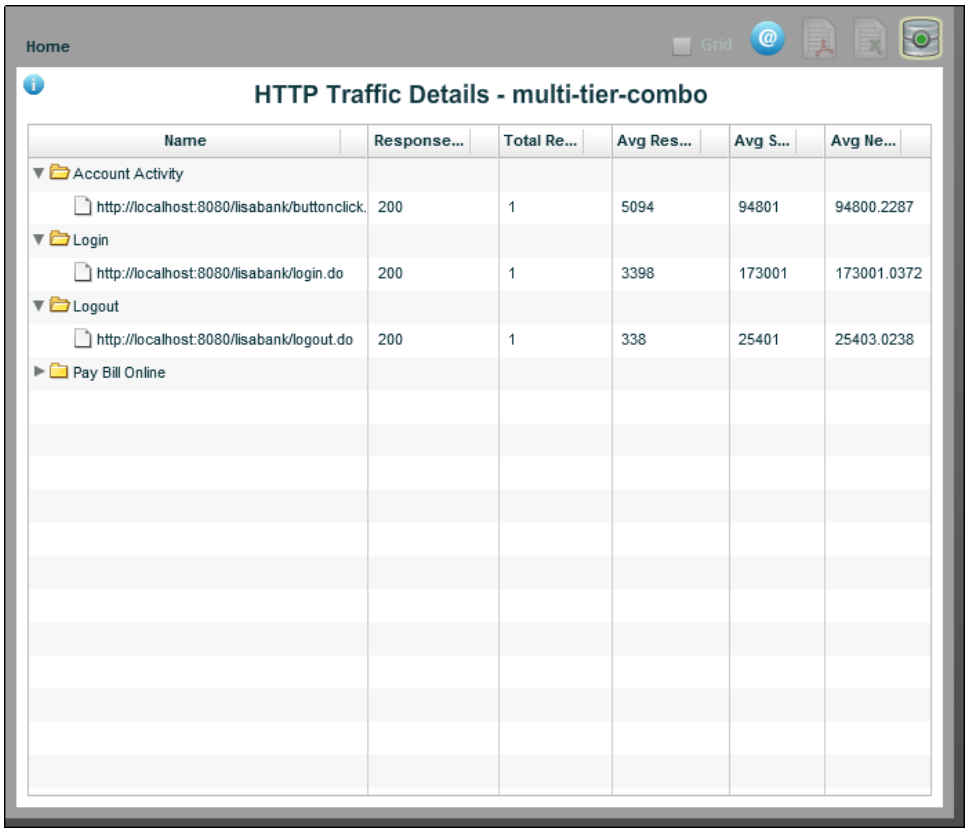
Performance Summary (Récapitulatif des performances)



Cycle Performance Summary (Récapitulatif des performances du cycle)

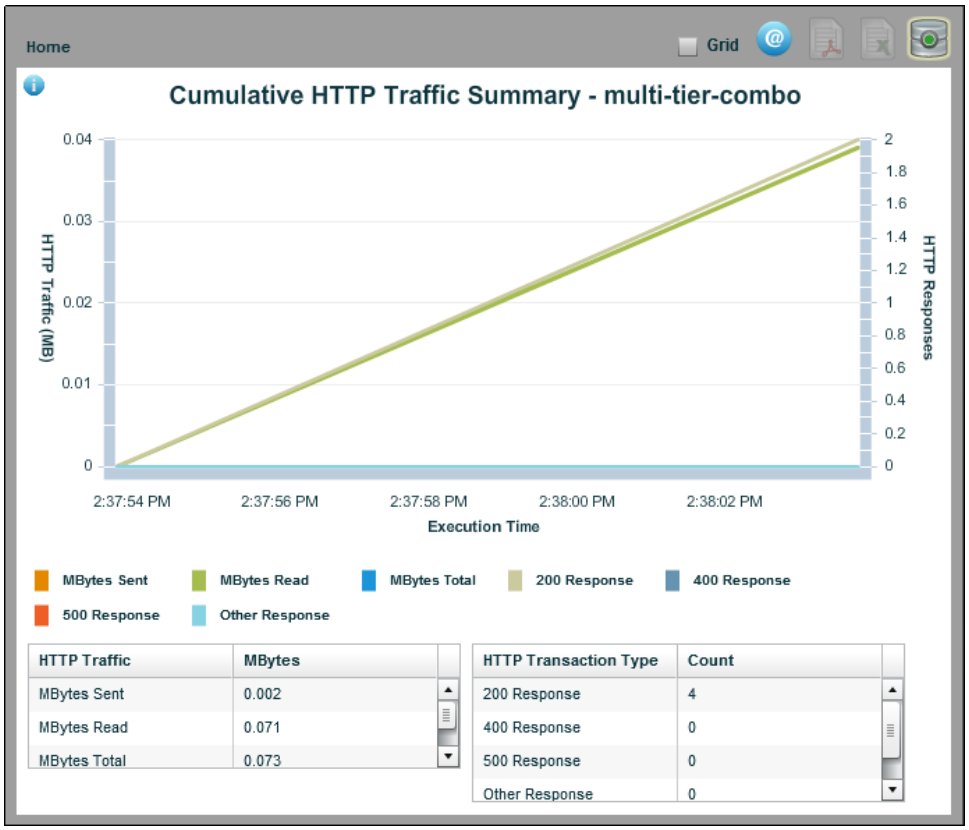


HTTP Details (Détails HTTP)



HTTP Summary (Récapitulatif HTTP)

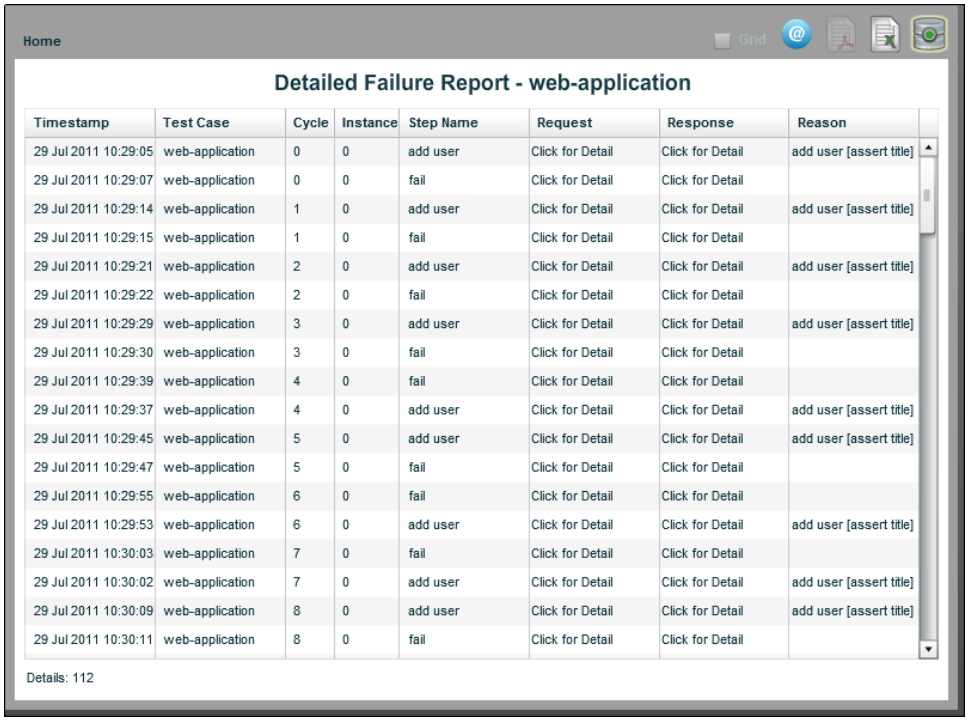
Pour afficher le rapport Cumulative HTTP Traffic Summary (Récapitulatif du trafic HTTP cumulatif), la propriété suivante doit être définie dans le fichier local.properties ou site.properties : **lisa.commtrans.ctstats=true**.



Exemples de rapport d'erreur

Les rapports d'erreur répertorient les erreurs et les avertissements provenant d'étapes dont l'exécution s'est terminée avec des erreurs. Les images suivantes représentent les rapports d'erreur suivants :

Detailed Failures (Echecs détaillés)


Home

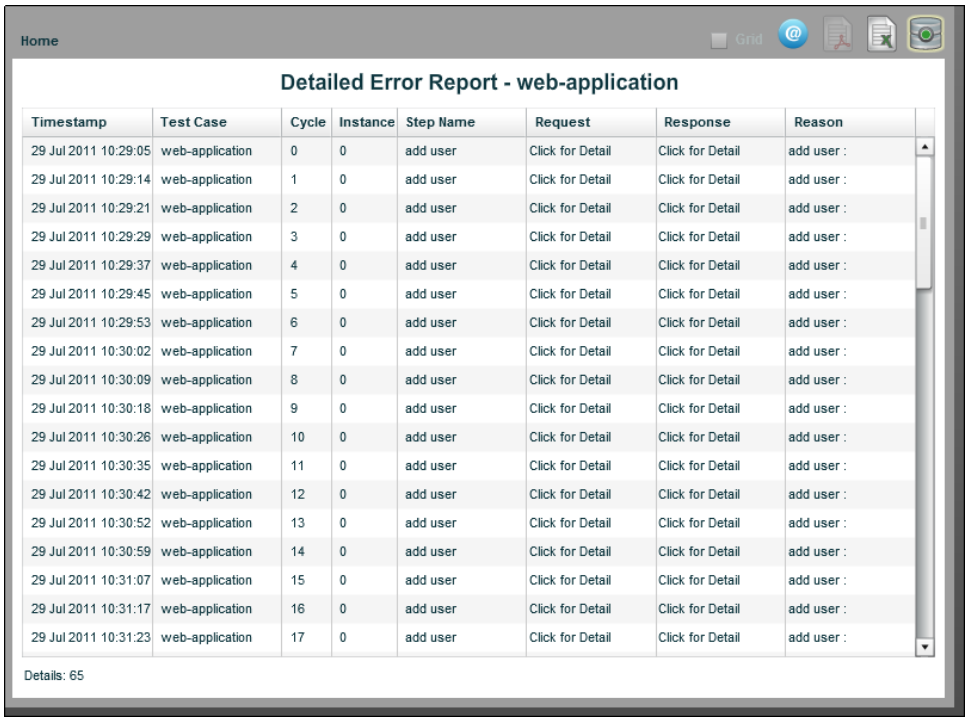
Grid

Detailed Failure Report - web-application

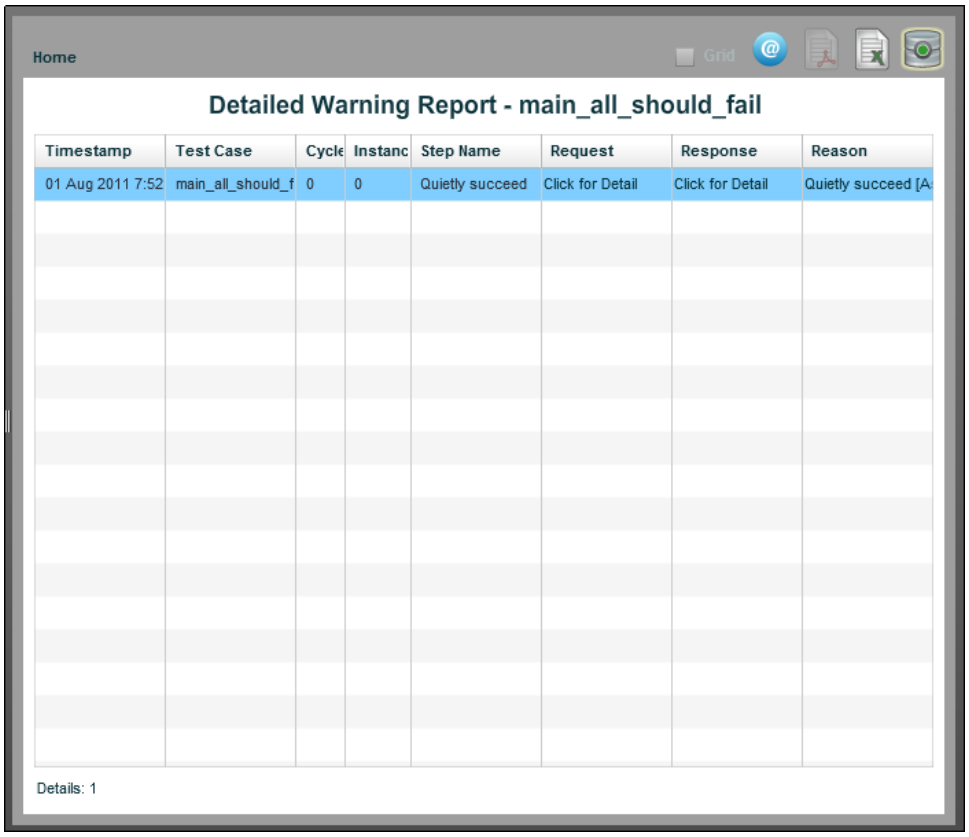
Timestamp	Test Case	Cycle	Instance	Step Name	Request	Response	Reason
29 Jul 2011 10:29:05	web-application	0	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:29:07	web-application	0	0	fail	Click for Detail	Click for Detail	
29 Jul 2011 10:29:14	web-application	1	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:29:15	web-application	1	0	fail	Click for Detail	Click for Detail	
29 Jul 2011 10:29:21	web-application	2	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:29:22	web-application	2	0	fail	Click for Detail	Click for Detail	
29 Jul 2011 10:29:29	web-application	3	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:29:30	web-application	3	0	fail	Click for Detail	Click for Detail	
29 Jul 2011 10:29:39	web-application	4	0	fail	Click for Detail	Click for Detail	
29 Jul 2011 10:29:37	web-application	4	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:29:45	web-application	5	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:29:47	web-application	5	0	fail	Click for Detail	Click for Detail	
29 Jul 2011 10:29:55	web-application	6	0	fail	Click for Detail	Click for Detail	
29 Jul 2011 10:29:53	web-application	6	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:30:03	web-application	7	0	fail	Click for Detail	Click for Detail	
29 Jul 2011 10:30:02	web-application	7	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:30:09	web-application	8	0	add user	Click for Detail	Click for Detail	add user [assert title]
29 Jul 2011 10:30:11	web-application	8	0	fail	Click for Detail	Click for Detail	

Details: 112

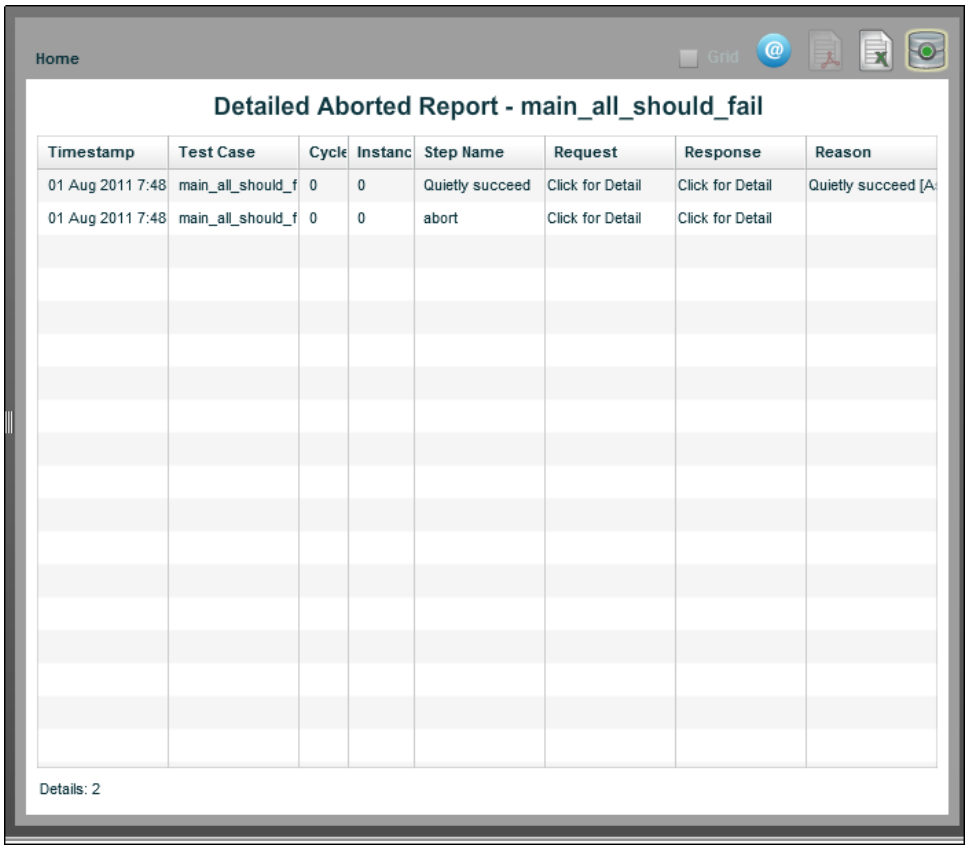
Detailed Errors (Erreurs détaillées)



Detailed Warnings (Avertissements détaillés)



Detailed Aborts (Interruptions détaillées)

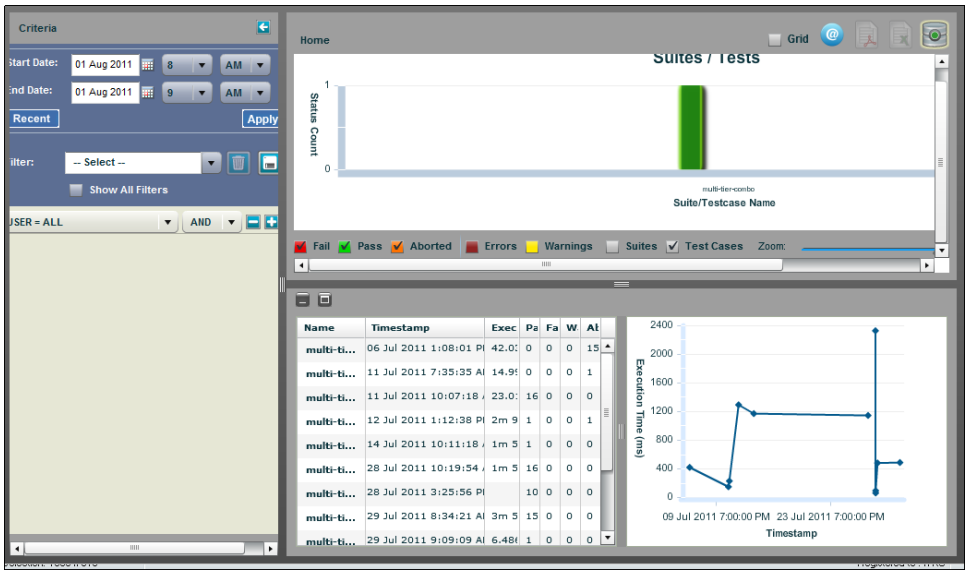

Home
Grid
Detailed Aborted Report - main_all_should_fail
Timestamp
Test Case
Cycle
Instanc
Step Name
Request
Response
Reason
01 Aug 2011 7:48
main_all_should_f
0
0
Quietly succeed
Click for Detail
Click for Detail
Quietly succeed [A
01 Aug 2011 7:48
main_all_should_f
0
0
abort
Click for Detail
Click for Detail
Details: 2

Exemple de rapport View History (Afficher l'historique)

Le rapport View History (Afficher l'historique) réduit le graphique actuel à la moitié supérieure de la fenêtre et affiche deux rapports supplémentaires :

- Une liste de grille de scénarios de test exécutés et leurs résultats
- Un graphique à courbes indiquant un historique des heures d'exécution

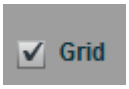
Ce rapport affiche des informations sur les exécutions précédentes du même scénario de test.



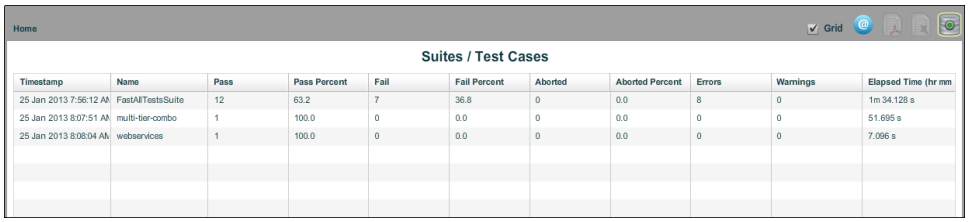
Vue de grille des rapports

La vue de grille affiche les mêmes informations que la vue graphique, mais sous la forme d'une grille. Tous les filtres d'informations de rapport fonctionnent de la même façon.

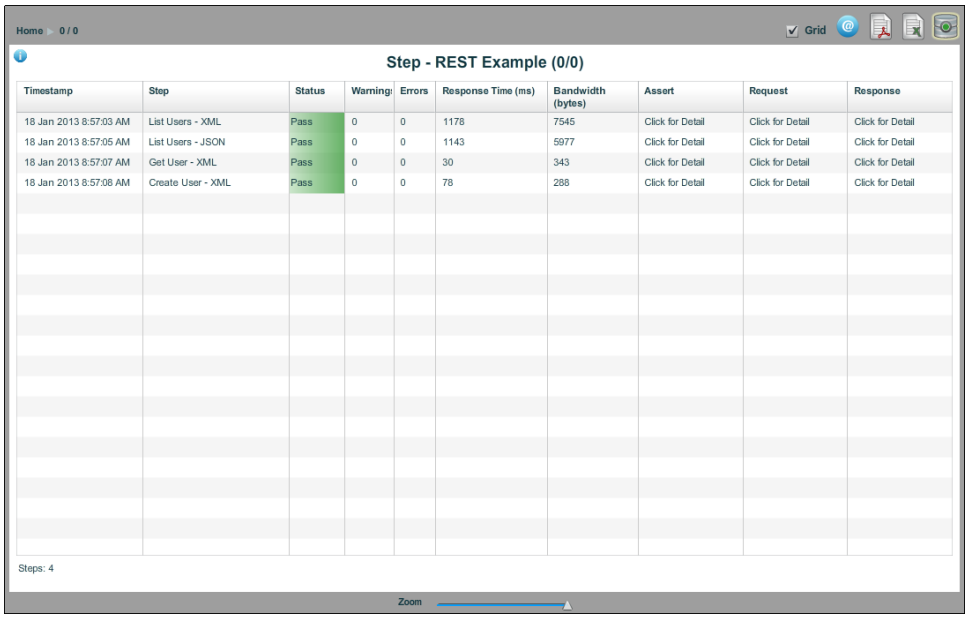
Remarque : Pour pouvoir exporter des rapports vers Excel, la vue de grille doit être active.

Pour afficher les résultats dans une grille, sélectionnez Grid (Grille) 

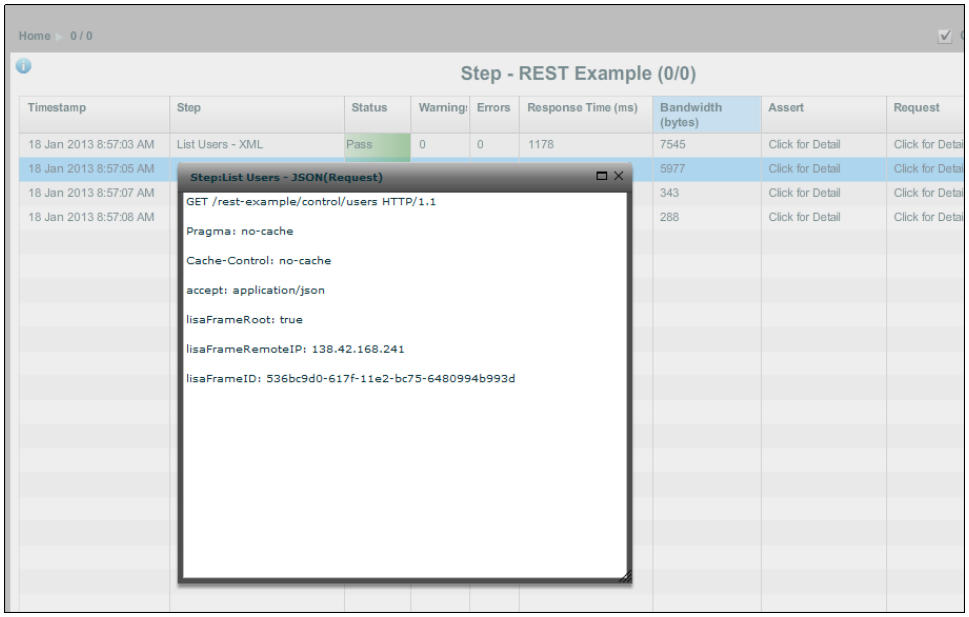
en haut de la fenêtre.



Vous pouvez sélectionner chaque scénario de test figurant dans le rapport afin d'obtenir plus de détails pour le scénario de test sélectionné.



Si vous cliquez sur Click for Detail (Cliquez pour obtenir des détails.) dans les colonnes Assert (Assertion), Request (Demande) ou Response (Réponse), des informations détaillées s'affichent pour chaque composant de l'étape.



Cliquez sur cette option pour afficher les données de demande et de réponse. Pour les données XML, vous pouvez sélectionner l'onglet Formatted XML (Contenu XML formaté) pour afficher le texte formaté. Pour les données non XML, l'onglet Formatted XML indique que le texte n'est pas un XML valide.

Lorsque le rapport Performance Summary (Récapitulatif des performances) est affichée dans la vue de grille, vous pouvez sélectionner des options à partir de la liste déroulante Customize (Personnaliser) pour indiquer la valeur de centile à afficher dans la dernière colonne du rapport. La valeur de centile représente le nombre maximum de millisecondes pendant lesquelles un certain pourcentage d'exécution d'étape a été effectué. Par exemple, si le 75ème cycle le plus long de l'étape A s'est terminé en 7,9 secondes, le 75ème centile serait 7 900 ou une valeur supérieure.

Rapports standard

Vous pouvez générer quatre rapports préformatés et les exporter au format PDF.

- Functional Test Report (Rapport de test fonctionnel) (page 432)
- Performance Test Report (Rapport de test de performances) (page 433)
- Suite Summary Report (Rapport récapitulatif de suite) (page 435)
- Metrics Report (Rapports de mesures) (page 437)

Définitions importantes :

- La taille **Sample Size** (Taille de l'échantillon) dépend du document de simulation. Par défaut, un échantillonnage est effectué à chaque seconde, puis une moyenne est calculée toutes les 10 secondes. Vous pouvez changer la taille de l'échantillon en déplaçant les curseurs.
- Un **cycle** est une exécution complète d'un scénario de test entier.
- La valeur de **Avg Cycle Exec Time** (Temps d'exécution du cycle moyen) correspond à la moyenne de la durée d'exécution du test du début à la fin (cycle).

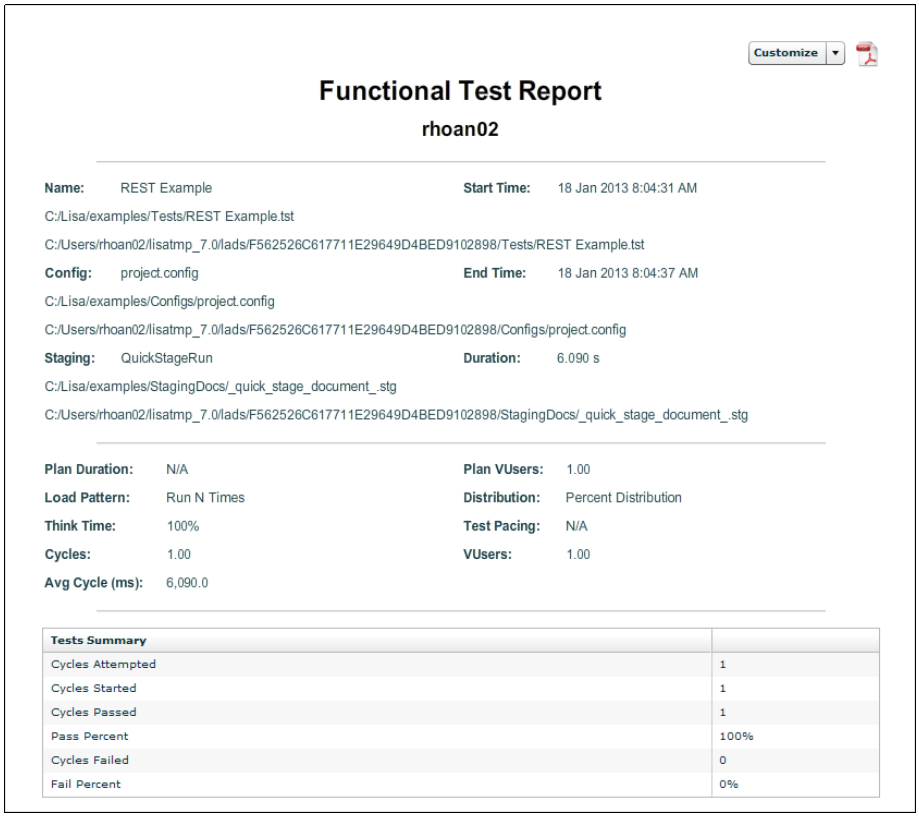
Rapport Functional Test Report (Rapport de test fonctionnel)

Pour produire un rapport Functional Test Report (Rapport de test fonctionnel) :

1. Pour afficher les étapes du rapport, double-cliquez sur un test.
2. Sélectionnez une étape, puis Export to PDF (Exporter dans un fichier PDF)  .

   La fenêtre Select a Report (Sélectionner un rapport) s'ouvre.
3. Dans la liste de rapports disponibles, sélectionnez Functional Test Report (Rapport de test fonctionnel) pour le test, puis cliquez sur OK.

   La liste déroulante Customize (Personnaliser) vous permet de générer un récapitulatif détaillé de ce rapport.

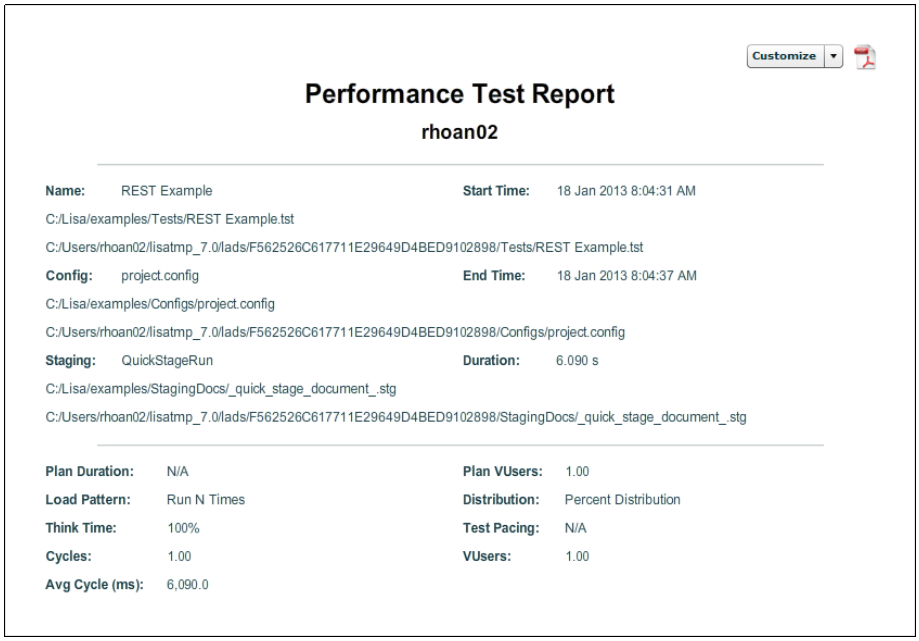


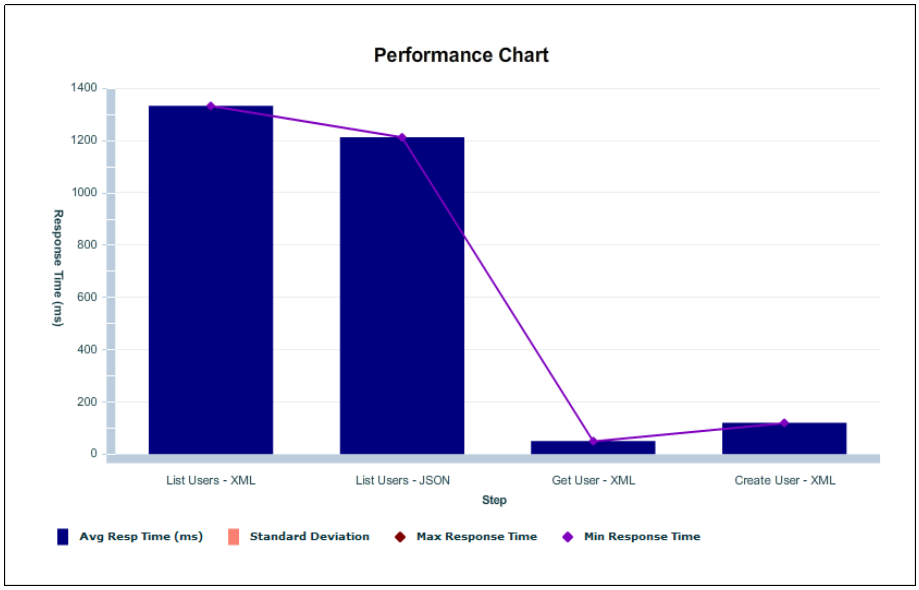
Rapport Performance Test Report (Rapport de test de performances)

Pour produire un rapport Performance Test Report (Rapport de test de performances) :

1. Cliquez avec le bouton droit de la souris sur une étape de test.
2. Cliquez sur Export to PDF (Exporter vers un fichier PDF)  .

   La fenêtre Select a Report (Sélectionner un rapport) s'ouvre.
3. Dans la liste de rapports disponibles, sélectionnez Performance Report et cliquez sur OK.





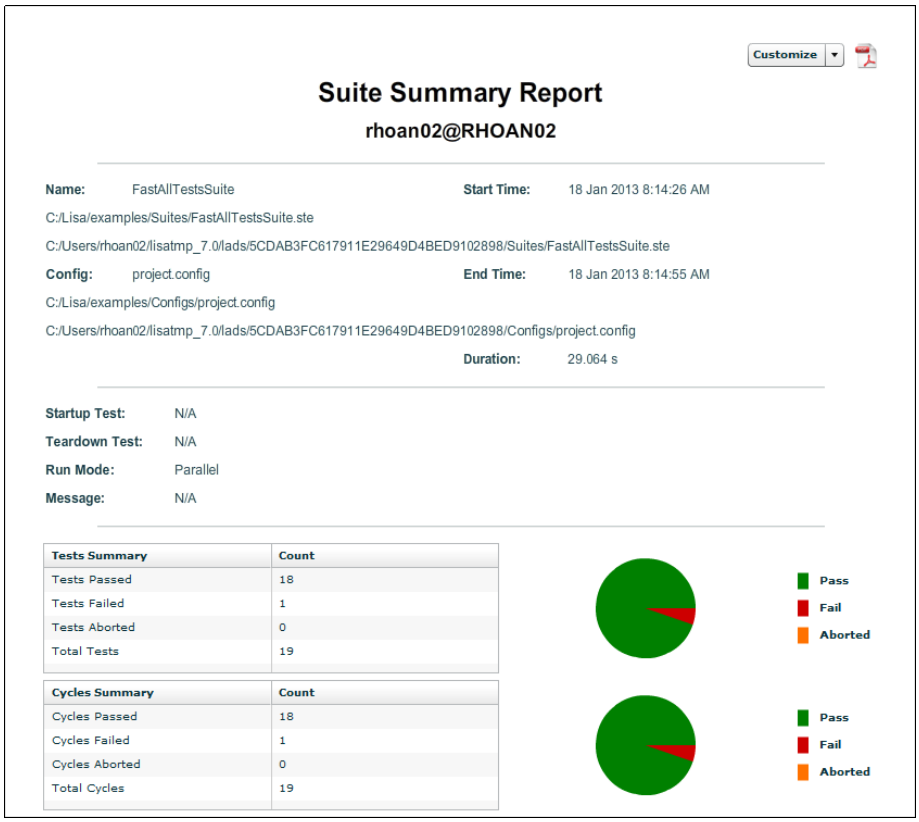
Step	Total Time (ms)	Sample Size	Min (ms)	Max (ms)	Avg (ms)	Median (ms)	Std Dev	90th percentile
List Users - XML	1,332	1	1,332	1,332	1,332.00	1,332.00	0.0	1,332.0
List Users - JSON	1,212	1	1,212	1,212	1,212.00	1,212.00	0.0	1,212.0
Get User - XML	50	1	50	50	50.00	50.00	0.0	50.0
Create User - XML	120	1	120	120	120.00	120.00	0.0	120.0

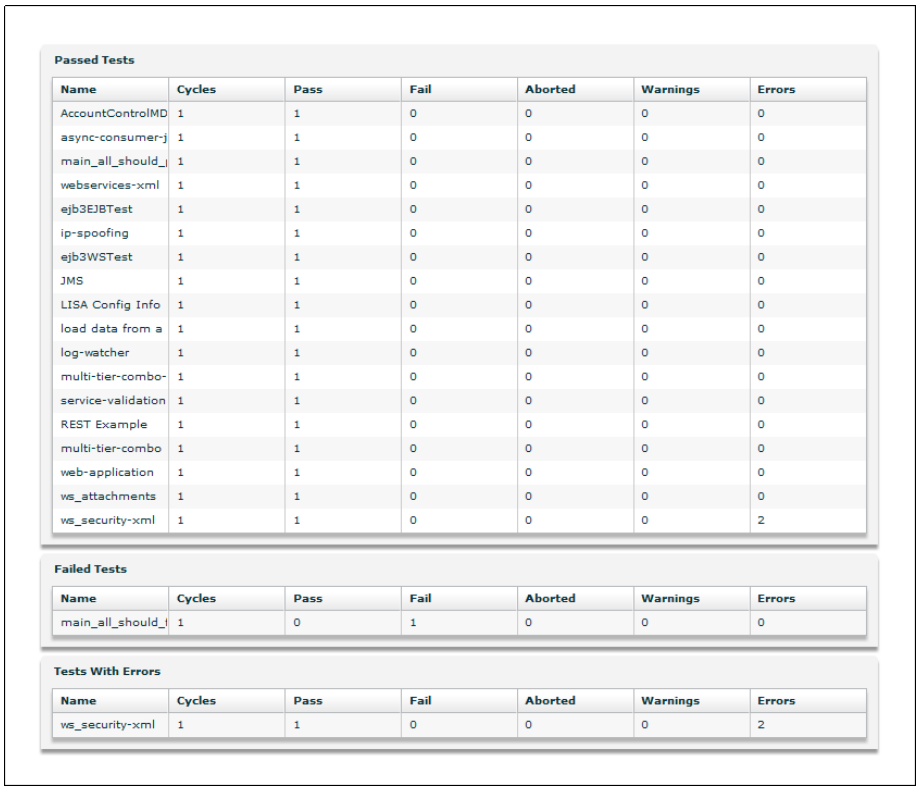
Rapport Suite Summary Report (Rapport récapitulatif de suite)

Pour produire un rapport Suite Summary Report (Rapport récapitulatif de suite) :

1. Ouvrez un rapport de suite de tests.
2. Cliquez sur Export to PDF (Exporter vers un fichier PDF)  .
3. La fenêtre Select a Report (Sélectionner un rapport) s'ouvre.
4. Dans la liste de rapports disponibles, sélectionnez Suite Summary Report et cliquez sur OK.

   La liste déroulante Customize (Personnaliser) vous permet d'afficher des détails sur les types de tests ayant échoué. Si tous les tests ont réussi, la vue Details (Détails) est identique à la vue Summary (Récapitulatif).





Passed Tests

Name	Cycles	Pass	Fail	Aborted	Warnings	Errors
AccountControlMD	1	1	0	0	0	0
async-consumer-j	1	1	0	0	0	0
main_all_should_	1	1	0	0	0	0
webservices-xml	1	1	0	0	0	0
ejb3EJBTest	1	1	0	0	0	0
ip-spoofing	1	1	0	0	0	0
ejb3WSTest	1	1	0	0	0	0
JMS	1	1	0	0	0	0
LISA Config Info	1	1	0	0	0	0
load data from a	1	1	0	0	0	0
log-watcher	1	1	0	0	0	0
multi-tier-combo-	1	1	0	0	0	0
service-validation	1	1	0	0	0	0
REST Example	1	1	0	0	0	0
multi-tier-combo	1	1	0	0	0	0
web-application	1	1	0	0	0	0
ws_attachments	1	1	0	0	0	0
ws_security-xml	1	1	0	0	0	2

Failed Tests

Name	Cycles	Pass	Fail	Aborted	Warnings	Errors
main_all_should_	1	0	1	0	0	0

Tests With Errors

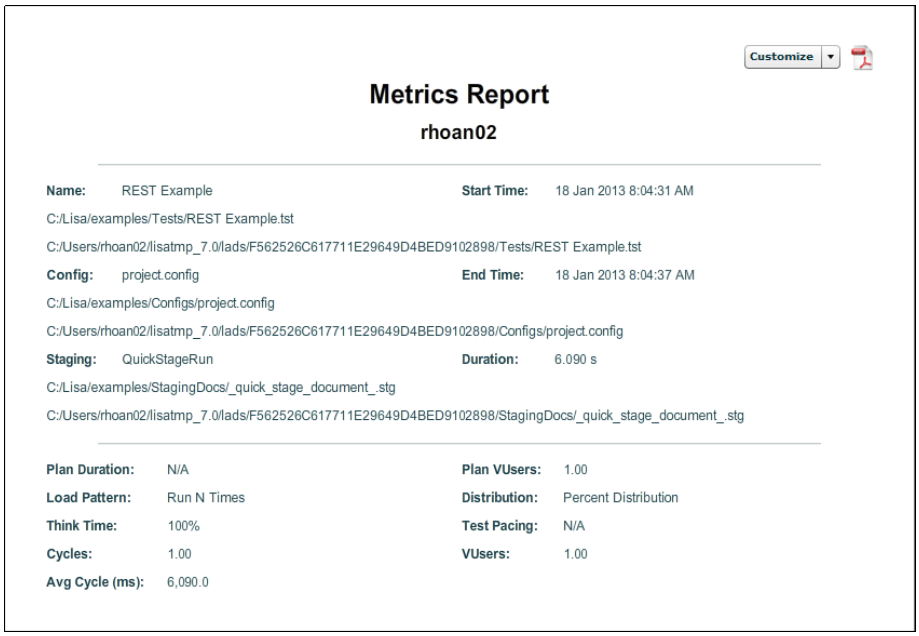
Name	Cycles	Pass	Fail	Aborted	Warnings	Errors
ws_security-xml	1	1	0	0	0	2

Rapport Metrics Report (Rapports de mesures)

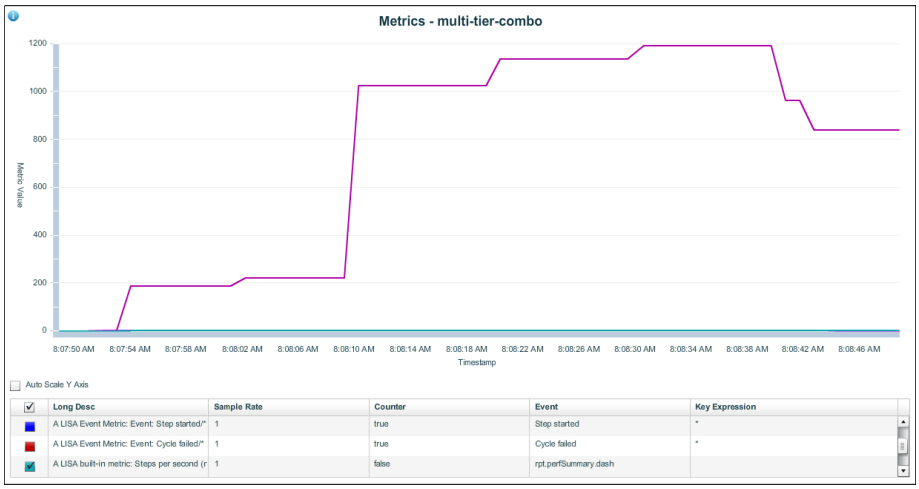
Pour produire un rapport Metrics Report :

1. A partir d'un rapport de test/suite, cliquez avec le bouton droit de la souris sur un test ou une suite.
2. Sélectionnez Analyze (Analyser), Metrics (Mesures) pour afficher un graphique ou une grille de mesures.
3. A partir de ce rapport, cliquez sur Export to PDF (Exporter dans un fichier PDF)  .

   La fenêtre Select a Report (Sélectionner un rapport) s'ouvre.
4. Dans la liste de rapports disponibles, sélectionnez Metrics Report et cliquez sur OK.



Cliquez avec le bouton droit de la souris sur la fenêtre pour afficher le graphique de mesures. La légende est interactive et vous pouvez sélectionner des événements et le paramètre de mise à l'échelle automatique. Dans le rapport au format PDF, utilisez la liste déroulante Customize (Personnaliser) pour effectuer des sélections.

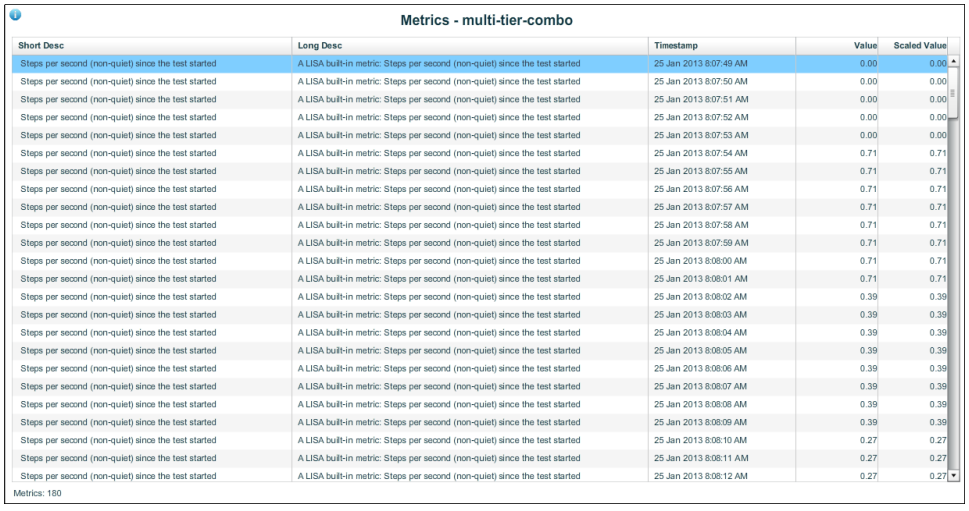


L'option Customize vous permet de sélectionner les options suivantes :

- Detailed Summary (Récapitulatif détaillé)
- Auto Scale Y Axis (Ajuster automatiquement l'échelle de l'axe Y)
- All Metrics (Toutes les mesures)
- No Metrics (Aucune mesure)
- Metrics (Mesures, une sélection)

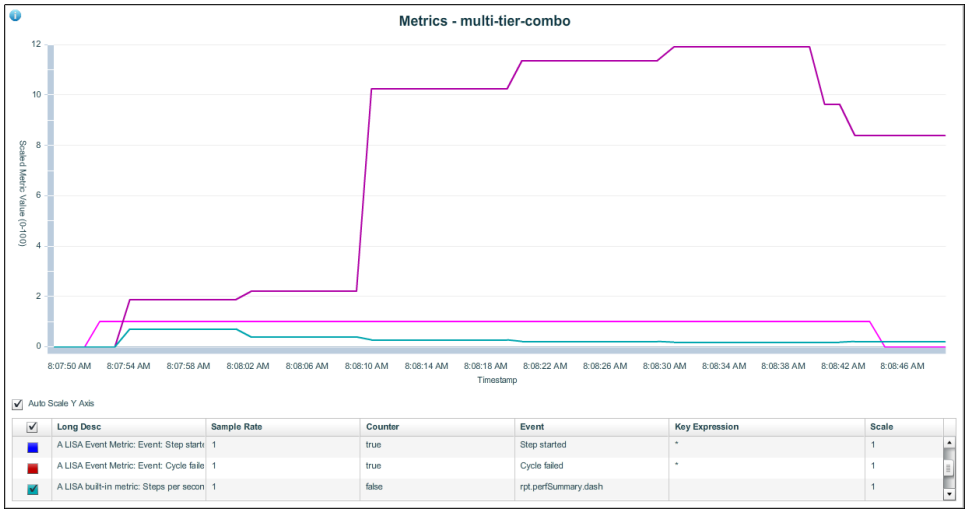
Chaque rapport personnalisé est représenté dans les exemples suivant.

Detailed Summary (Récapitulatif détaillé)



Auto Scale Y Axis (Ajuster automatiquement l'échelle de l'axe Y)

Comparez ce rapport au même rapport affiché précédemment pour voir la mise à l'échelle automatique de l'axe Y.



All Metrics (Toutes les mesures)

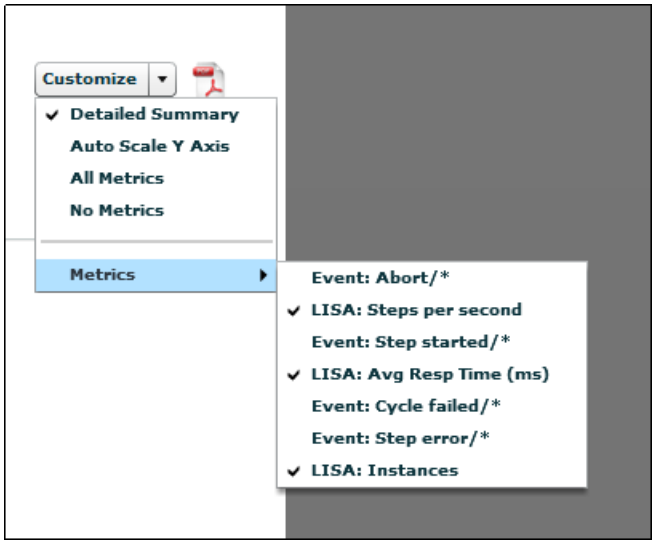
All Metrics est la sélection par défaut pour les rapports de mesures. Si vous avez sélectionné No Metrics ou Metrics (Mesures) dans la liste déroulante Customize (Personnaliser), vous pouvez sélectionner All Metrics pour afficher toutes les mesures disponibles pour le rapport.

No Metrics (Aucune mesure)

Lorsque vous sélectionnez No Metrics dans la liste déroulante Customize, toutes les mesures sont effacées du graphique de mesures.

Metrics (Mesures)

Vous pouvez sélectionner les mesures à afficher dans le graphique de mesures en sélectionnant des mesures disponibles dans la liste déroulante Metrics.


Customize
Detailed Summary
Auto Scale Y Axis
All Metrics
No Metrics
Metrics
Event: Abort/*
LISA: Steps per second
Event: Step started/*
LISA: Avg Resp Time (ms)
Event: Cycle failed/*
Event: Step error/*
LISA: Instances

Interprétation des rapports

Parfois, les résultats des rapports ne sont pas conformes à vos attentes et peuvent contenir des données de rapport incorrectes.

Exécution de tests en boucle

Inclure des boucles dans les scénarios de test peut entraîner des résultats inattendus dans le moteur de génération de rapports.

Le moteur de flux de travaux est conçu pour traiter une tâche de l'étape de début à l'étape de fin, sans boucles. L'exécution de cette chaîne d'étapes est censée renvoyer une réussite, un échec ou une interruption. L'exécution en boucle suppose une initialisation externe au scénario de test à l'aide d'un document de simulation. L'utilisation d'un document de simulation permet de spécifier 10 utilisateurs virtuels pour une exécution unique d'un scénario de test. Vous obtenez donc 10 événements indiquant une réussite, un échec ou une interruption. Lorsque l'exécution en boucle est effectuée dans le scénario de test, aucune réussite, interruption ni aucun échec ne se produit tant que l'étape de fin n'est pas atteinte, créant ainsi un événement unique indiquant une réussite, un échec ou une interruption.

Rapports et délai de réflexion

Le délai de réflexion indique la durée pendant laquelle DevTest patiente avant de démarrer l'exécution d'une étape de test. Le but du délai de réflexion est de simuler un utilisateur réel interagissant avec un système. Vous pouvez définir un délai de réflexion dans l'éditeur d'étapes ou dans un document de simulation.

Le panneau de rapports principal affiche la durée d'exécution totale, qui correspond à la période du début à la fin du test. Comme il s'agit d'une durée, les délais de réflexion sont inclus de manière prédéfinie, afin de comptabiliser la durée totale de l'exécution du test. Pour exclure le délai de réflexion, définissez-le sur 0 dans le document de simulation ou l'étape de test.

Si vous recherchez des données de performances, générez un rapport de performances qui indique les temps de réponse pour chaque étape qui n'inclut aucun délai de réflexion. Vous obtenez un rapport plus significatif pour les tests de performances. Le délai de réflexion n'est pas comptabilisé dans le temps moyen de performances de l'étape.

Exportation de rapports

Dans la visionneuse Report Viewer (Visionneuse de rapports), vous pouvez exporter les rapports dans trois formats :

- XML
- Microsoft Office Excel
- Adobe Acrobat PDF

Exportation de rapports dans un fichier XML

Si le rapport au format XML est spécifié dans la suite de tests ou le document de simulation, vous pouvez exporter les données de rapport directement dans un fichier XML. Pour procéder à l'exportation, cliquez avec le bouton droit de la souris sur le test ou la suite, puis sélectionnez Export (Exporter). Pour plus d'informations, consultez la rubrique Vue graphique des rapports (page 413). Une fois que les données exportées sont enregistrées, vous pouvez les formater ou les manipuler.

L'exportation des données de rapport au format XML vous permet de déplacer les données de rapport d'un registre à un autre.

Exportation de rapports dans un fichier Excel

Vous pouvez exporter toutes les données des rapports dans une feuille de calcul Excel.

Procédez comme suit:

1. Ouvrez le rapport à exporter.
2. Cochez la case Grid (Grille).

   Le rapport doit être affiché dans la vue de grille avant de pouvoir l'exporter dans un fichier Excel.
3. Cliquez sur Export to Excel (Exporter vers un fichier Excel).
4. Spécifiez le nom du fichier Excel.

   Les données de rapports sont stockées dans le fichier Excel enregistré.

Exportation de rapports dans un fichier PDF

Vous pouvez également exporter les données de rapport dans un fichier PDF.

Procédez comme suit:

1. Ouvrez la vue graphique du rapport.
2. Cliquez sur l'icône Export to PDF (Exporter dans un fichier PDF).

   La fenêtre Select a Report (Sélectionner un rapport) s'ouvre.
3. Sélectionnez le rapport à exporter.

Changement de bases de données de rapports

Vous pouvez définir le mode de configuration de DevTest de sorte à le connecter à d'autres bases de données dans **lisa.properties** ou **site.properties**. Définissez les composants de base de données de DevTest en assignant chacun d'eux à une configuration de pool définie comme suit :

```
lisadb.reporting.poolName=common
lisadb.vse.poolName=common
lisadb.legacy.poolName=common
lisadb.acl.poolName=common
```

Par défaut toutes les bases de données DevTest partagent un pool commun de connexions à la base de données Derby, tel que défini dans les paramètres suivants.

```
lisadb.pool.common.driverClass=org.apache.derby.jdbc.ClientDriv
er
lisadb.pool.common.url=jdbc:derby://localhost:1528/database/lis
a.db;create=true
lisadb.pool.common.user=rpt
lisadb.pool.common.password=rpt
```

Par exemple, si vous voulez changer la base de données de rapports et vous connecter à une base de données Oracle, définissez d'abord une nouvelle configuration de pool de base de données,

```
lisadb.pool.mypool.driverClass=oracle.jdbc.OracleDriver
lisadb.pool.mypool.url=jdbc:oracle:thin:@//myhost:1521/orcl
lisadb.pool.mypool.user=rpt
lisadb.pool.mypool.password=rpt
```

Puis, configurez le composant de rapports de sorte à utiliser ce pool.

```
lisadb.reporting.poolName=mypool
```

Lorsque vous entrez dans le portail Reporting Portal (Portail de rapports), vous pouvez passer la souris sur l'icône de base de données dans le coin supérieur droit de la fenêtre pour obtenir des informations détaillées sur la base de données que vous utilisez.

Dépannage des performances de génération de rapports

Si vous exécutez des tests de charge et qu'un goulot d'étranglement dans les scénarios de test écrit les événements dans la base de données de rapports, supprimez toutes les données sauf les mesures du générateur de rapports.

En général, le problème survient lorsque DevTest tente d'enregistrer un nombre excessif de données lors d'un test de charge. Dans l'onglet Reports (Rapports) de l'éditeur Test Suite Editor (Editeur de suites de tests), désactivez les événements dans l'ordre suivant :

Properties Set/Referenced (Propriétés définies/référencées)

Cet événement génère le plus grand nombre de lignes dans la base de données et n'est presque jamais utile pour un test de charge. Notez qu'il peut facilement y avoir plus de 10 propriétés Get et Set pour chaque étape exécutée. Dans un test de charge, cela se traduit rapidement par des millions de lignes dans la base de données. Par exemple, si vous avez un scénario de test comprenant 5 étapes (avec 5 propriétés GET et 5 propriétés SET pour chaque étape) et vous exécutez 100 utilisateurs pendant 10 000 cycles, vous obtiendrez 50 millions de lignes dans la base de données.

Record All Events (Enregistrer tous les événements)

Cet événement est essentiellement un enregistrement du moteur de flux de travaux. Il peut également générer plus de 5 lignes pour toutes les étapes.

Request / Response (Demande/Réponse)

Cet événement ne crée aucune ligne supplémentaire, mais les charges utiles sont stockées en tant qu'objets CLOB, car leur taille peut être importante. Cela crée probablement la plus grande quantité de données, sans toutefois trop affecter la base de données, car la taille des tables et des index n'augmentent pas aussi rapidement.

Si le moteur de génération de rapports continue à être un goulot d'étranglement, activez la propriété **lisa.reporting.useAsync**=false. Cette propriété indique à DevTest d'utiliser JMS pour envoyer l'événement de rapport. Les threads d'arrière-plan dans les simulateurs et le coordinateur enregistrent les événements dans la base de données de façon asynchrone. Votre test de charge se termine donc avant que tous les événements soient écrits dans la base de données et le rapport ne s'affiche pas tout de suite. Le délai dépend de vos scénarios de test et du nombre d'événements générés. La file d'attente de simulateur tarde généralement plus longtemps à se vider. Vous pouvez consulter le pourcentage effectué dans un message au niveau INFO du journal de simulateur.

La fonctionnalité de génération de rapports asynchrone permet au simulateur de s'exécuter plus rapidement, car il ne ralentit pas pour enregistrer des données dans la base de données. A la place, il place les données dans une file d'attente JMS pour un enregistrement ultérieur.

Chapitre 18: Enregistreurs et générateurs de tests

L'environnement de test sans code DevTest Workstation permet aux équipes de QA, de développement et autres de concevoir et d'exécuter rapidement des tests fonctionnels, d'unité, de régression et de charge pour des sites Web dynamiques (RIA).

Vous pouvez utiliser le produit pour tester des navigateurs riches, des interfaces utilisateur Web, les nombreux éléments structuraux et les données derrière l'interface utilisateur. DevTest permet aux équipes d'analyser, d'appeler et de vérifier toutes les couches de données et d'implémentation qui doivent être testées fonctionnellement, de manière à garantir que toutes les conditions requises soient remplies.

Ce chapitre traite des sujets suivants :

Génération d'un service Web

Vous pouvez créer un scénario de test de service Web (XML). Utilisez l'étape Web Service Execution (XML) (Exécution de service Web) pour appeler des opérations de service Web dans un scénario de test et tester la réponse et la demande. Ces opérations de service Web fournissent la même fonctionnalité que les appels de méthode équivalents dans l'objet EJB utilisé dans le didacticiel 7. Pour plus d'informations sur l'étape Web Service Execution (Exécution de service Web), consultez la section Etape Web Service Execution (XML) (Exécution de service Web) de la rubrique *Utilisation CA Application Test.*

Pour utiliser cette étape, vérifiez que le serveur de démonstration est en cours d'exécution.

Pour générer un service Web, effectuez les trois étapes suivantes :

1. Création de l'étape de service Web (XML) (page 446)
2. Créez le client de service Web. (page 446)
3. Exécutez le scénario de test. (page 446)

Création de l'étape de service Web (XML)

Procédez comme suit:

1. Ouvrez l'éditeur de modèles dans DevTest Workstation.
2. Cliquez avec le bouton droit de la souris dans la fenêtre de l'éditeur de modèles et sélectionnez Add Step (Ajouter une étape), Web/Web Services (Web/Services Web), Web Service Execution (XML) (Exécution de service Web).

   Une étape de services Web est ajoutée.
3. Renommez l'étape **addUser** dans la zone Step Information (Informations sur l'étape).
4. Double-cliquez sur l'étape addUser.
5. L'éditeur Web Service Execution s'ouvre.
6. Pour créer un document XML, cliquez sur New Document (Nouveau document).

Création du client de service Web

Procédez comme suit:

1. Entrez l'emplacement du code WSDL dans le champ WSDL URL (URL du document WSDL).

   `http://WSSERVER:WSPORT/itko-examples/services/UserControlService?wsdl`
2. Dans le champ Service Name (Nom du service), saisissez **UserControlServiceService**.

   N'utilisez pas d'espaces dans le nom du service Web.
3. Dans le champ Port, saisissez **UserControlServiceService**.
4. Dans la liste Operation (Opération), sélectionnez l'opération à tester.
5. Dans la liste On Error (En cas d'erreur), sélectionnez l'action à effectuer en cas d'erreur du test : Abort the Test (Interrompre le test).

Exécution du scénario de test

Procédez comme suit:

1. Cliquez sur Execute (Exécuter)  .

   Le test s'exécute et affiche la demande et la réponse.
2. Pour afficher la demande lors de l'exécution, cliquez sur l'onglet Request (Demande).
3. Pour afficher la réponse lors de l'exécution, cliquez sur l'onglet Response (Réponse).

Enregistrement d'un site Web

DevTest Workstation fournit un enregistreur HTTP permettant d'exécuter un scénario de test de site Web à l'aide d'un enregistreur de proxy.

Cet outil vous permet de suivre votre progression dans le site Web et de créer automatiquement des étapes de test pour chaque requête HTTP générée pendant l'enregistrement.

Remarque : Avant de commencer l'enregistrement, désactivez la mise en cache ou videz le cache de votre navigateur pour que l'enregistreur Web puisse récupérer le codage du paramètre de demande client à partir de paquets de réponse de serveur pour les systèmes de codage multi-octets. Si l'enregistreur Web ne peut pas capturer les paquets de réponse, il décode ou code le paramètre selon la norme ISO-8859-1 par défaut et les paramètres capturés peuvent être altérés.

Pour démarrer l'enregistrement :

1. Cliquez sur Recording (Enregistrement)  .
2. Cliquez sur Record Test Case for User Interface (Enregistrer le scénario de test pour l'interface utilisateur), Web Recorder (HTTP proxy) (Enregistreur Web (proxy HTTP)).

   Dans le menu principal, sélectionnez Actions, Record Test Case for User Interface, Web Recorder (HTTP proxy).

   Cela vous permet de lancer le navigateur utilisé pour enregistrer et lire les tests HTTP.

   - Si un scénario de test est déjà ouvert dans l'onglet actif, vous êtes invité à spécifier si vous voulez relire les tests dans le navigateur.
   - Si aucun scénario de test n'est ouvert dans DevTest Workstation, la fenêtre Test Recorder (Enregistreur de test) s'ouvre et vous permet d'entrer le nom du site Web à enregistrer.
3. Entrez l'URL pour la page Web à tester.

   Remarque : La variable localhost (par exemple, **http://localhost:8080/lisabank/)** ne fonctionne pas avec l'enregistrement de proxy HTTP. Pour l'URL, utilisez le nom d'hôte ou l'adresse IP au lieu de la variable localhost.
4. Sélectionnez vos préférences :

   - HTML Responses Only (Réponses HTML uniquement) **:** capture uniquement les réponses HTML.
   - Use External Browser (Utiliser un navigateur externe) **:** ouvre une fenêtre de navigateur externe.

Utilisation de ports

Par défaut, l'enregistreur de proxy DevTest utilise le port 8010 pour l'enregistrement.

Si vous ne voulez pas utiliser ce port, vous pouvez remplacer le paramètre dans le fichier **lisa.properties**, avec la propriété suivante : **lisa.editor.http.recorderPort=8010**.

Pour commencer l'enregistrement, cliquez sur Start Recording (Lancer l'enregistrement) pour démarrer l'enregistreur ou cliquez sur Proxy Settings (Paramètres de proxy) pour configurer le proxy.

Les rubriques suivantes sont disponibles.

- Configuration des paramètres de proxy (page 449)
- Démarrage de l'enregistrement (page 450)
- Affichage des transactions enregistrées (page 451)
- Affichage dans l'ITR (page 451)

Configuration des paramètres de proxy

Pour ouvrir la fenêtre Proxy Settings (Paramètre de proxy), cliquez sur Proxy Settings dans la fenêtre Test Recorder (Enregistreur de test). Vous pouvez configurer les paramètres de proxy suivants :

Use Proxy (Utiliser un proxy)

Pour utiliser des paramètres de proxy, sélectionnez cette option. Elle vous permet d'entrer les paramètres de proxy lorsque vous voulez l'utiliser de façon intermittente.

Web Proxy Server (Serveur proxy Web)

Entrez le nom d'hôte du serveur proxy (adresse IP du serveur) et le numéro de port respectifs.

Bypass Web Proxy for these hosts and domains (Omettre le proxy Web pour ces hôtes et domaines)

Entrez le nom d'hôte et les domaines des serveurs pour lesquels vous voulez omettre le proxy. Entrez une liste d'hôtes, séparés par une barre verticale (|), auxquels se connecter directement sans passer par un serveur proxy. Vous pouvez utiliser un astérisque * comme caractère générique pour la mise en correspondance. Par exemple, *.foo.com|localhost.

Secure Web Proxy Server (Serveur proxy Web sécurisé)

Entrez les paramètres de proxy sécurisé.

Bypass Web Proxy Server for these hosts and domains (Omettre le serveur proxy Web pour ces hôtes et domaines)

Entrez le nom d'hôte et les domaines pour lesquels vous voulez omettre le proxy.

Exclude simple host names (Exclure les noms d'hôte simples)

Sélectionnez cette option si vous voulez exclure les noms d'hôte simples. Par exemple, **localhost** et **servername** par rapport à **192.168.1.1** ou **server1.company.com**.)

Proxy server Authentication (Authentification du serveur proxy)

Saisissez les informations d'authentification :

- Domain (Domaine)
- User name (Nom d'utilisateur)
- Password (Mot de passe)

Send Preemptively (Envoyer à titre préventif)

Sélectionnez Wait for Challenge (Attendre la stimulation), Send Basic (Envoyer les paramètres de base) ou Send NTLM (Envoyer les paramètres NTLM).

Remarque : En général, le serveur proxy stimule les demandes lorsque l'authentification est requise. Si le serveur proxy n'envoie pas de stimulation, l'activation de cette option forcera la définition de l'en-tête d'authentification à la première demande.

Cliquez sur OK pour enregistrer les modifications et définir les paramètres de proxy.

Démarrage de l'enregistrement

Procédez comme suit:

1. Pour lancer l'enregistrement du test, cliquez sur Start Recording (Lancer l'enregistrement) dans l'enregistreur Test Recorder (Enregistreur de test).

   La fenêtre Test Recorder s'ouvre et affiche l'URL de la page Web chargée.

   Remarque : Pour enregistrer des caractères Unicode, utilisez un navigateur externe.

2. Vous pouvez tester la page Web en entrant des informations qu'un utilisateur normal saisirait.

3. Une fois le test terminé, cliquez sur Stop Recording (Arrêter l'enregistrement) au bas de la fenêtre Test Recorder pour arrêter l'enregistrement de l'activité du navigateur.

   La fenêtre Recorded Elements (Eléments enregistrés) s'affiche.

4. Cliquez sur Commit Edits (Valider les modifications) pour terminer l'enregistrement.

Affichage des transactions enregistrées

Une fois que vous avez arrêté l'enregistrement, toutes les transactions enregistrées sont affichées dans l'onglet Recorded Elements (Eléments enregistrés).

Toutes les transactions sont répertoriées dans le panneau gauche. L'onglet de droite contient les détails de l'étape et la réponse.

Procédez comme suit:

1. Sélectionnez l'onglet Response (Réponse) pour afficher la réponse HTML enregistrée.
2. Cliquez sur Commit Edits (Valider les modifications) pour valider les transactions dans l'enregistreur Test Recorder (Enregistreur de test).

   Le panneau Parameters In Web Recording (Paramètres dans l'enregistrement Web) s'ouvre.
3. Entrez les paramètres requis pour votre test.
4. Pour ajouter des transactions en tant qu'étapes de test, cliquez sur Add to Test and Close (Ajouter au test et fermer) au bas de la fenêtre Test Recorder.

   Un scénario de test est créé selon vos demandes HTTP, dans le flux de travaux de DevTest. Chaque étape de test du scénario de test représente une requête HTTP enregistrée.

Affichage dans l'ITR

Vous pouvez afficher toutes ces transactions à nouveau lorsque vous exécutez ce scénario de test dans l'ITR.

Pour afficher des informations supplémentaires sur l'étape enregistrée, reportez-vous aux onglets View (Afficher), Source et DOM Tree (Arborescence DOM).

Enregistrement d'un scénario de test d'application mobile

Procédez comme suit:

1. Vérifiez que le fichier de configuration définissant l'actif mobile souhaité est actif.
2. Créez un scénario de test (page 219).
3. Cliquez sur Create steps by recording or templating (Créer des étapes à l'aide d'un enregistrement ou d'un modèle)  .
4. Cliquez sur Record Test Case for User Interface (Enregistrer le scénario de test pour l'interface utilisateur), Mobile Recorder (Enregistreur mobile).

   La fenêtre Test Recorder (Enregistreur de test) s'ouvre.

   Remarque : Si vous utiliser un simulateur mobile pour l'enregistrement, la fenêtre de simulateur mobile s'ouvre également.
5. Si vous avez défini plusieurs actifs mobiles, sélectionnez un actif auquel se connecter dans la liste déroulante Choose Mobile Asset (Sélectionner un actif mobile), puis cliquez sur OK.
6. Cliquez sur Start Recording (Lancer l'enregistrement) au bas de la fenêtre d'enregistreur.
7. Dans la fenêtre Test recorder (Enregistreur de test), effectuez les actions à enregistrer pour votre scénario de test.

   Important : N'effectuez pas ces actions directement dans le simulateur mobile. DevTest Workstation envoie toutes les actions que vous effectuez dans l'enregistreur de test au simulateur automatiquement.

   L'enregistreur de test met en surbrillance chaque élément de la fenêtre avec lequel vous interagissez. Un contour rouge indique une position de fenêtre. Un contour vert indique l'élément de fenêtre plus spécifique. Lorsque vous placez le pointeur sur un élément de la fenêtre, une info-bulle s'affiche et l'identifie. Lorsqu'il y a plusieurs comportements pour le même bouton ou composant, l'info-bulle vous indique quel est l'objet sur lequel vous avez cliqué.
8. Pour ajouter manuellement des actions ou des gestes supplémentaires lors de l'enregistrement, cliquez avec le bouton droit de la souris sur la fenêtre ou l'élément dans l'enregistreur de test, puis sélectionnez l'action appropriée.

   Par exemple, vous pouvez manuellement insérer un "secouer", un changement d'orientation ou une assertion au cours du processus d'enregistrement. Pour plus d'informations sur les actions disponibles, consultez la rubrique Modification d'étapes de test d'applications mobiles.

   Remarque : Une étape de test d'application mobile contient autant de gestes (appuyer, glisser) que nécessaires pour terminer l'étape. Ces sous-étapes, ou actions, s'affiche dans la table Actions lorsque vous double-cliquez sur une étape de test.
9. Cliquez sur Stop Recording lorsque vous avez capturé toutes les actions pour le test.

La fenêtre d'enregistreur et le simulateur mobile se ferment. Le nouveau scénario de test est rempli avec des étapes de test qui représentent les actions d'applications mobiles capturées pendant l'enregistrement.

10. Pour afficher les détails de chaque étape :

    a. Cliquez sur l'étape de test à réviser. Chaque étape de test contient une capture d'écran de l'action effectuée.

    b. Pour afficher les détails, cliquez sur l'étape Mobile Testing Step (Etape de test d'application mobile) dans l'arborescence d'élément à droite.

       L'onglet Mobile Testing Step (Etape de test d'application mobile) s'ouvre et affiche une capture d'écran de l'application de test. La section Actions, en haut de l'onglet, indique les actions effectuées dans l'étape de test. Pour sélectionner l'élément associé dans la capture d'écran, cliquez sur une action.

    c. Pour afficher la capture d'écran associée à une action, cliquez sur l'action dans la section Actions.

       Pour plus d'informations sur la modification d'une étape de test enregistrée, consultez la rubrique Modification d'étapes de test d'applications mobiles.

       Pour plus d'informations sur l'ajout d'assertions à une étape de test, consultez la rubrique Ajout d'une assertion à une étape de test d'applications mobiles (page 158).

Exécution d'un scénario de test d'application mobile dans l'ITR

Procédez comme suit:

1. Vérifiez que le fichier de configuration contenant l'actif mobile souhaité est actif.
2. Ouvrez le scénario de test (page 219) à exécuter.
3. Dans la barre d'outils, cliquez sur Start a new ITR (Démarrer une nouvelle ITR)  .

   Remarque : Indiquez si démarrer une nouvelle exécution d'un test interactif doit être démarrée ou ouvrir une exécution d'un test interactif précédente (si vous avez exécuté l'ITR précédemment).

   La fenêtre Interactive Test Run (Exécution d'un test interactif) s'ouvre.
4. Cliquez sur Execute (Exécuter)  au bas de la fenêtre de l'ITR.

   La fenêtre de simulateur mobile s'ouvre et DevTest exécute les étapes de test. L'application mobile s'affiche dans le simulateur lors de l'exécution du test.

   Un message s'affiche indiquant que le test est terminé. La fenêtre du simulateur se ferme.
5. Cliquez sur OK.

   Pour plus d'informations sur l'utilisation de l'utilitaire d'exécution d'un test interactif, consultez la rubrique Exécution de scénarios de test et de suites (page 291).

Exécution d'un scénario de test d'application mobile à distance

Procédez comme suit:

1. Vérifiez que le fichier de configuration qui définit l'actif mobile attendu est actif.
2. Ouvrez le scénario de test (page 219)à exécuter.
3. Dans la barre d'outils, cliquez sur l'icône ITR  .

   Remarque : Indiquez si démarrer une nouvelle exécution d'un test interactif doit être démarrée ou ouvrir une exécution d'un test interactif précédente (si vous avez exécuté l'ITR précédemment).

   La fenêtre Interactive Test Run (Exécution d'un test interactif) s'ouvre.
4. Cliquez sur Execute (Exécuter)  au bas de la fenêtre de l'ITR.

   La fenêtre de simulateur mobile s'ouvre et DevTest exécute les étapes de test.

   Pour afficher les informations du scénario de test, connectez-vous à votre fournisseur de services cloud (Par exemple, SauceLabs.com). Vous pouvez afficher le test sous la forme d'une capture vidéo en direct lors de son exécution sur l'émulateur SauceLabs.

   Un message dans DevTest indique que le test est terminé.
5. Cliquez sur OK.

   Pour plus d'informations sur l'utilisation de l'utilitaire d'exécution d'un test interactif, consultez la rubrique Exécution de scénarios de test et de suites (page 291).

Chapitre 19: Test d'application mobile

Cette section comprend les rubriques suivantes :

Mise en route de tests d'application mobile

Cette section comprend les rubriques suivantes :

Présentation des tests d'application mobile

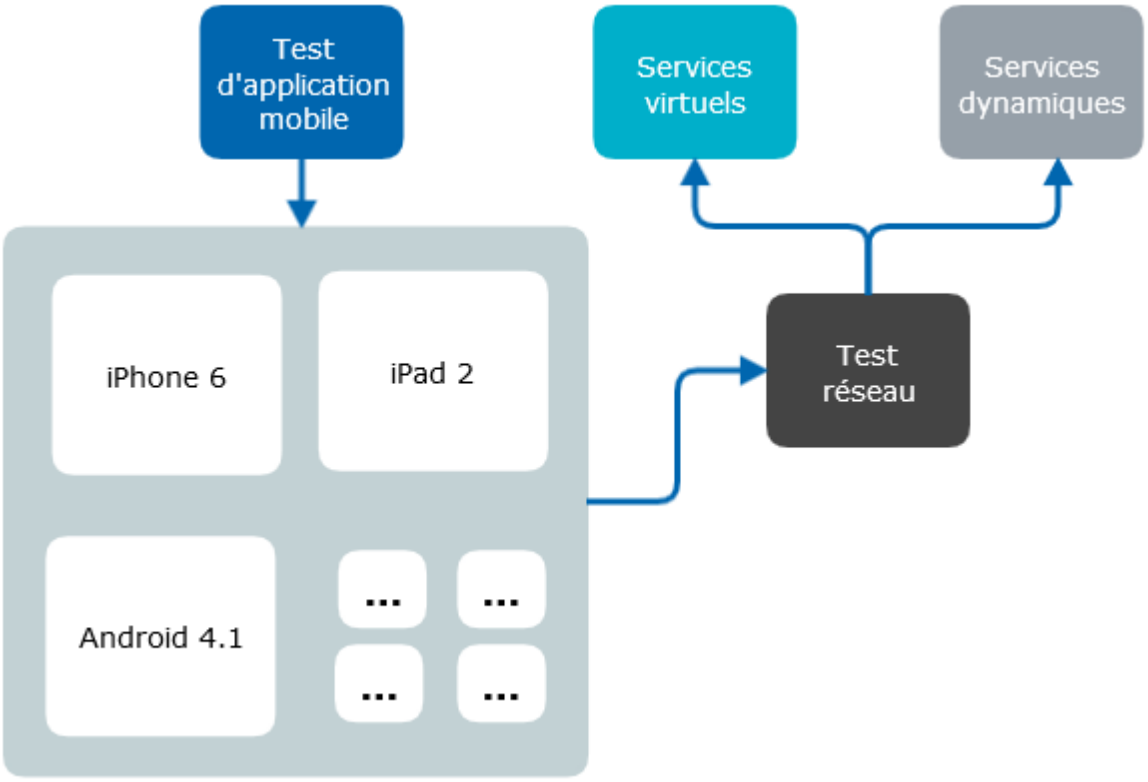
Les tests d'application mobile permettent d'étendre les fonctionnalités de base de DevTest pour le développement, la simulation et la surveillance des scénarios de test aux tests d'application mobile iOS, Android et hybride. Vous pouvez créer des scénarios de test en enregistrant les opérations effectuées avec vos applications mobiles. Vous pouvez ensuite adapter ces tests, comme pour les autres scénarios de test DevTest, en ajoutant des actions, des assertions et des filtres à chaque étape.

L'enregistreur de test d'application mobile prend en charge les méthodes d'enregistrement suivantes :

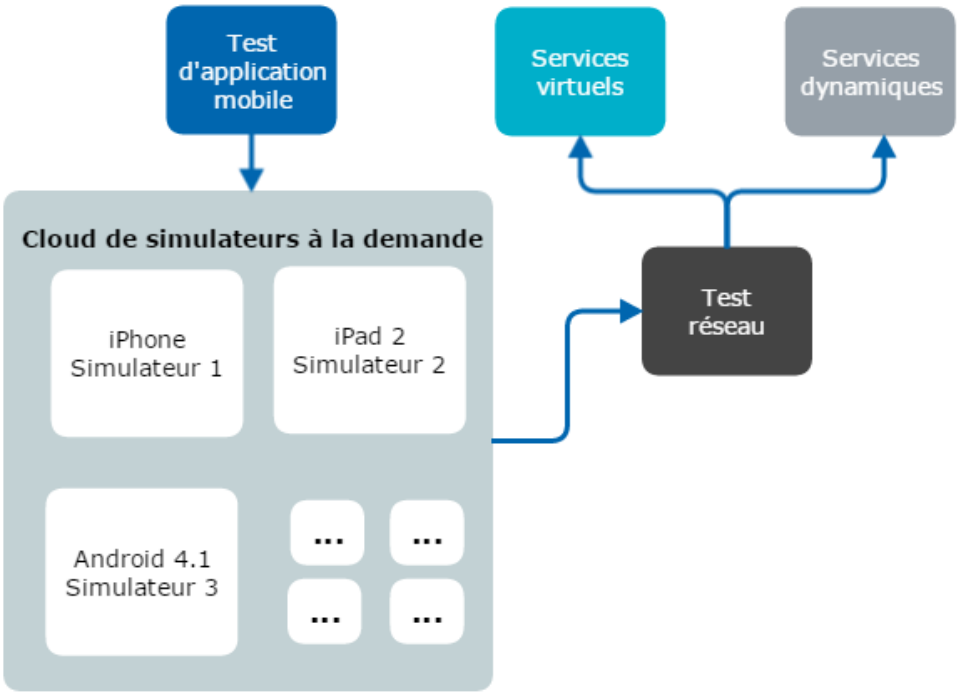
- Enregistrement direct à partir d'une unité connectée à votre ordinateur via le port USB.
- Enregistrement à l'aide des simulateurs Android et iOS installés.
- Enregistrement à partir de simulateurs Android et iOS cloud pour une solution évolutive

La fonctionnalité de test d'application mobile vous permet de créer un scénario de test unique que vous pouvez appliquer à plusieurs unités. Cette capacité associée à l'ensemble de simulateurs disponibles simplifie le processus de développement et de test des applications pour différents dispositifs mobiles.

L'illustration suivante présente un environnement de test d'application mobile avec des dispositifs connectés et des simulateurs cloud.



L'illustration suivante présente un environnement de test d'application mobile avec des simulateurs cloud installés.



Remarque : Pour plus d'informations sur la configuration de tests d'application mobile, reportez-vous à la section Configuration de l'environnement de tests d'application mobile.

Gestes et actions prises en charge pour les applications mobiles

DevTest Solutions prend en charge les comportements mobiles suivants :

Geste	Application native iOS	Application native Android	Application hybride iOS	Application hybride Android
Appuyer	Oui	Oui	Oui	Oui
Appui long	Oui	Oui	Oui	Oui
Faire glisser un élément	Oui	Oui	Oui	Oui
Balayer l'écran	Oui	Oui	Oui	No
Pincement	Oui	Oui	Oui	No
Zoom	Oui	Oui	Oui	N/D
Rotation	Oui	Oui	Oui	N/D
Défilement	Oui	Oui	Oui	N/D
Modifier l'orientation	Oui	Oui	Oui	Oui

Secouer	Oui	Oui	Oui	N/D
Précédent	No	Oui	No	Oui
Accéder à l'arrière-plan	Oui	N/D	N/D	N/D

Procédure de création et réexécution d'un scénario de test d'application mobile

Pour créer et réexécuter un scénario de test d'application mobile, effectuez les tâches suivantes.

Procédez comme suit:

1. Créez l'un des actifs suivants :

   Actif SauceLabs Session (Session SauceLabs)

   Définit les informations de compte SauceLabs pour les tests cloud d'applications mobiles.

   Actif Simulator Session (Session de simulateur)

   Définit le simulateur d'application mobile utilisé pour le test.

   Actif Attached Device Session (Session de dispositif connecté)

   Définit le dispositif mobile utilisé pour le test. Cet actif est utilisé pour des dispositifs mobiles physiques connectés à votre réseau.

   Pour plus d'informations sur chacun de ces actifs, reportez-vous à la section Actifs mobiles.

   Pour des informations générales sur la création d'actifs, consultez la section Création d'actifs (page 122).

2. Enregistrez un scénario de test d'application mobile (page 452).
3. Modifiez les étapes de test selon vos besoins.
4. Exécutez un scénario de test d'application mobile.
   - Pour obtenir des informations générales sur l'exécution d'un scénario de test, consultez la rubrique Exécution de scénarios et de suites (page 291).
   - Pour des informations supplémentaires sur l'exécution d'un scénario de test d'application mobile dans l'utilitaire d'exécution de test interactif, consultez la rubrique Exécution d'un scénario de test d'application mobile dans l'ITR (page 454).
   - Pour des informations spécifiques sur l'exécution d'un scénario de test d'application mobile à distance via SauceLabs, consultez la section Exécution d'un scénario de test d'application mobile à distance (page 455).

Test à distance

Utilisez la fonctionnalité de test à distance pour tester des dispositifs mobiles Android et iOS physiques reliés à un ordinateur d'un côté, auquel se connecte un autre ordinateur à partir d'un emplacement différent. La disponibilité de dispositifs distants vous permet de créer une stratégie d'analyse qui peut utiliser différentes combinaisons de systèmes d'exploitation et de dispositifs.

L'analyse à distance vous permet d'exécuter des tests de dispositifs mobiles avec différentes combinaisons de systèmes d'exploitation :

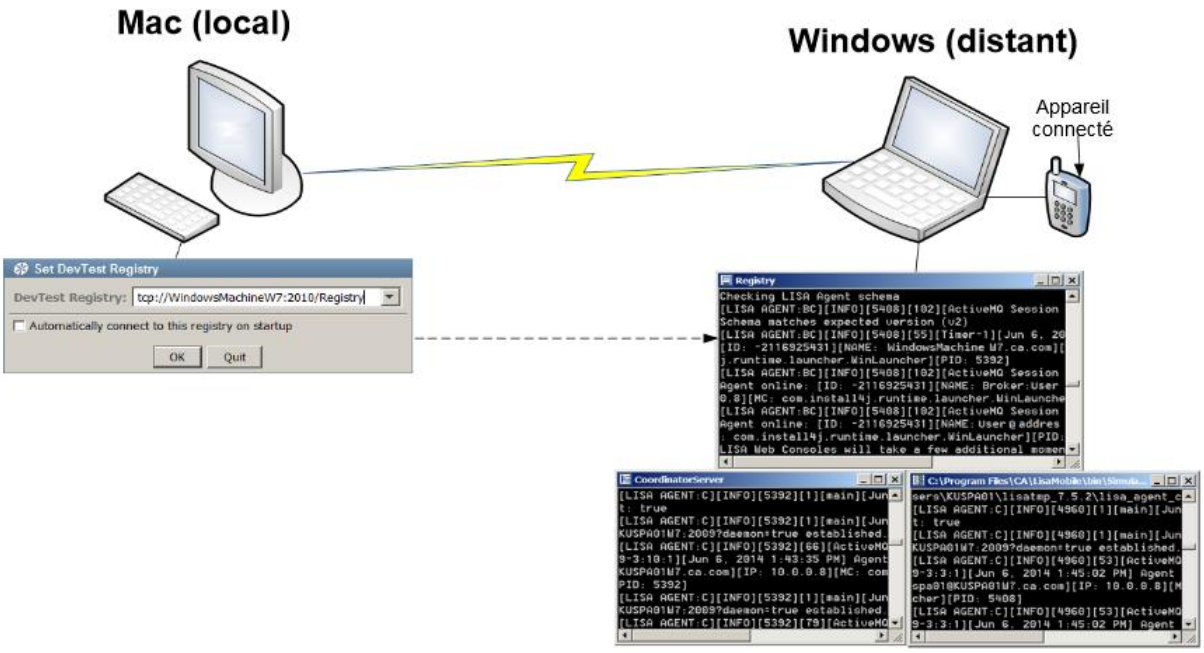
- Mac vers Windows
- Windows vers Mac
- Windows vers Windows
- Mac vers Mac

Le processus suivant s'applique pour le test à distance avec DevTest Workstation :

1. Connection de DevTest Workstation à partir d'un ordinateur local au registre, au simulateur et au coordinateur en cours d'exécution sur un ordinateur distant auquel un dispositif mobile physique est connecté.
2. Création et vérification d'un actif mobile sur l'ordinateur local avant d'exécuter les tests à distance.
3. Création d'un scénario de test sur votre ordinateur local.
4. Simulation du scénario de test sur l'ordinateur local, basé sur le registre, le simulateur et le coordinateur de l'ordinateur distant auquel vous êtes connecté.

Remarque : Vous pouvez uniquement exécuter des tests sur des dispositifs Android à partir d'un ordinateur Windows. Vous pouvez exécuter des tests Android et iOS à partir d'un Mac.

La procédure suivante explique la procédure d'exécution d'un test à distance pour une application native en cours d'exécution sur un dispositif Android. Dans cet exemple, le dispositif Android est connecté à un ordinateur Windows exécutant le registre, le simulateur et le coordinateur. DevTest Workstation sur le Mac se connecte au registre, au simulateur et au coordinateur sur l'ordinateur Windows, ou "ordinateur distant", comme dans l'illustration suivante :



Procédez comme suit:

1. Démarrez le registre, le simulateur et le coordinateur sur l'ordinateur Windows.
2. Démarrez DevTest Workstation sur le Mac.
3. Connectez-vous au registre en cours d'exécution sur l'ordinateur Windows. Entrez l'adresse du registre de l'ordinateur Windows dans la boîte de dialogue Set LISA Registry (Définir le registre LISA).
4. Créez un nouveau projet dans DevTest Workstation sur l'ordinateur Mac.
5. Copiez un fichier .apk (application Android native) dans le dossier Data (Données) de votre projet LISA sur l'ordinateur Mac. Pointez le champ Application de l'actif vers cet emplacement pour le fichier .apk. Par exemple :

   `Application : <emplacement_projet_LISA>/Data/ApiDemos.apk`
6. Vérifiez que votre ordinateur local contient un scénario de test enregistré et un actif vérifié. Si votre ordinateur contient déjà ces éléments, poursuivez à l'étape 10.
7. Sur l'ordinateur Mac, créez un actif de dispositif natif Mac Android pour ApiDemos.

8. Vérifiez l'actif à 100 pour cent.
9. Enregistrez et lisez un test localement, pour vérifier que les deux actions sont effectuées sur l'ordinateur local avec l'actif de dispositif que vous avez créé à l'étape 7.
10. Cliquez avec le bouton droit de la souris sur le scénario de test, puis cliquez sur Stage Test (Simuler le test).
11. Pour simuler le test, cliquez sur Play (Lire).

Vous pouvez également utiliser la fonctionnalité de suite de tests AllTestSuite pour effectuer une simulation à distance.

Procédez comme suit:

1. Cliquez avec le bouton droit de la souris sur AllTestSuite, puis sur Run with DCM tcp://<*registre_connecté*> (Exécuter avec DCM) pour simuler le test à distance.

   La boîte de dialogue Run Suite via tcp://<*registre_connecté*>/ (Exécuter la suite via) s'ouvre.
2. Cliquez sur Stage (Simuler).

   Les tests commencent sur l'ordinateur Mac et les résultats sont affichés. DevTest Workstation sur le Mac se connecte à un registre, un simulateur et un coordinateur distants en cours d'exécution sur l'ordinateur Windows.

   Remarque : Vérifiez que votre fichier AllTestSuite contient les éléments à simuler (tests, fichiers MAR, suites).

Test de navigateur Web

Pour enregistrer un test Web, DevTest fournit une application avec un navigateur Web intégré.

Procédez comme suit:

1. Créez un actif qui contient la combinaison de plate-forme et d'appareil pour laquelle vous voulez tester votre contenu Web.
2. Chargez le fichier .app ou .apk qui contient un navigateur Web intégré dans le champ Application de la boîte de dialogue Mobile Session (Session mobile).

   Remarque : DevTest installe une application avec un navigateur Web intégré pour iOS comme pour Android (Mobile Browser.app et MobileBrowser.apk). Dans le champ Application, accédez à :
   `LISA_HOME\examples\Data\mobile`
3. Enregistrez un scénario de test (page 452) pour l'application pour vérifier que le contenu est correct.

Laboratoire de tests d'application mobile

Le laboratoire de tests d'application mobile DevTest permet mettre un groupe d'unités et des simulateurs/émulateurs de votre réseau local à la disposition de CA Application Test à des fins d'interaction.

Plusieurs appareils peuvent être connectés à un seul ordinateur exécuté dans votre environnement. Le laboratoire de tests d'application mobile met ces appareils à la disposition de vos testeurs. Vous pouvez tester et gérer jusqu'à 50 appareils physiques connectés via USB ou des simulateurs. iOS, Android et plusieurs versions d'appareils sont pris en charge.

Le laboratoire de tests d'application mobile inclut la même fonctionnalité que DevTest Cloud Manager (page 368), mais également une fonction permettant d'identifier des appareils mobiles connectés et des émulateurs/simulateurs.

Démarrage de tests d'application mobile

Lorsque vous démarrer un laboratoire de tests d'application mobile virtuel, vous démarrez une instance de ce laboratoire.

Les laboratoires de tests d'application mobile virtuels permettent aux utilisateurs de stocker des tests d'application mobile dans un pool de matériel dédié (par exemple, des serveurs OSX dédiés pour des tests d'application iOS). Pour lancer un laboratoire de tests d'application mobile virtuel, DevTest requiert au moins un processus de serveur de coordination (par instance de laboratoire virtuel) associé à un ou plusieurs processus de serveur de simulation qui réaliseront les tâches de test. Il est recommandé d'utiliser une instance de simulateur par serveur. Lorsque le processus de serveur de coordination n'effectue aucun test réel, il sera requis pour la planification de tests et la génération de rapports de résultats correctes.

Pour plus d'informations, consultez la rubrique Démarrage d'un laboratoire (page 378).

Vous pouvez démarrer un laboratoire en appelant l'un des fichiers exécutables de DevTest Server et en spécifiant un nom de serveur et un nom de laboratoire.

Procédez comme suit:

1. Dans la ligne de commande, lancez le Mobile Lab Coordinator Server (serveur de coordination du laboratoire de tests d'application mobile).

   `CoordinatorServer -l LabName`

   où *LabName* identifie le nom du laboratoire défini par l'utilisateur. Il est recommandé d'utiliser un processus de coordinateur par laboratoire.

2. Lancez le Mobile Lab Simulator (Simulateur du laboratoire de tests d'application mobile).

   `Simulator -l LabName -n LabServerName`

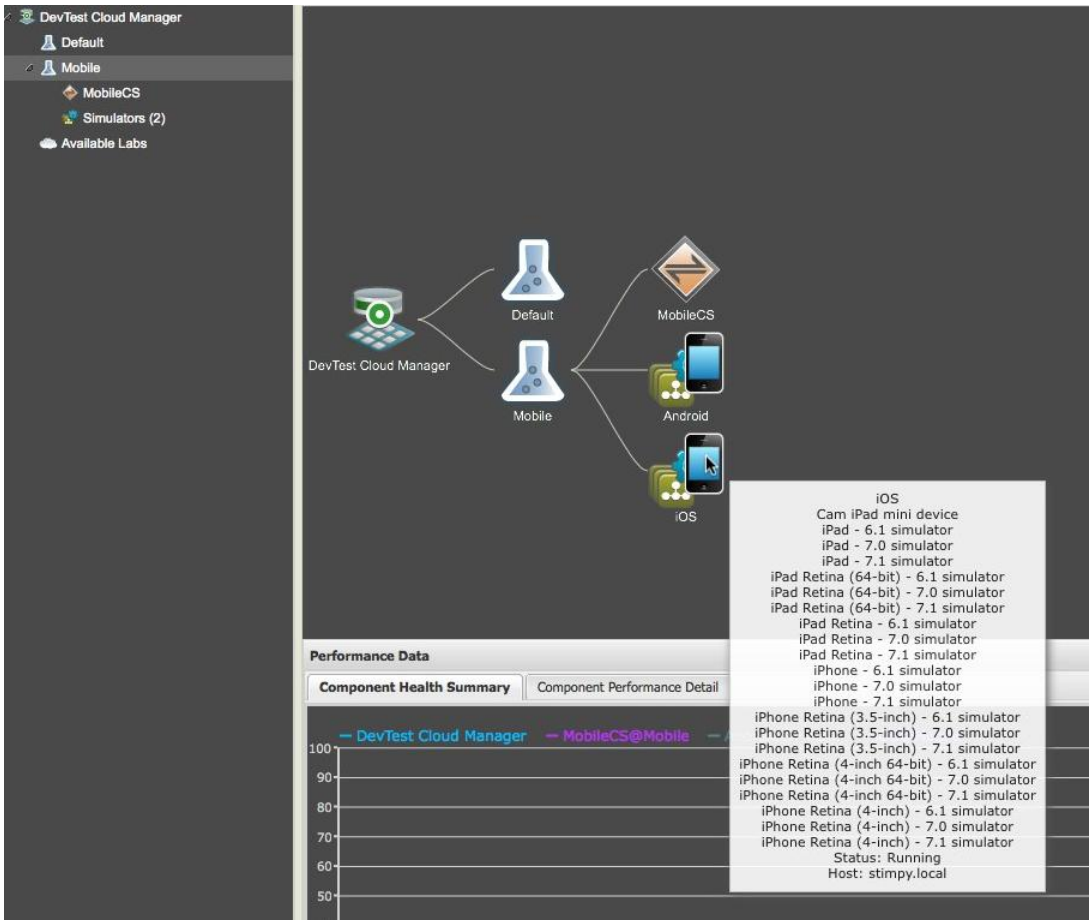
   où *Labname* identifie le nom du laboratoire défini par l'utilisateur et *LabServerName* est le nom du simulateur. Il est recommandé d'utiliser un simulateur par serveur de laboratoire.

   Remarque : Ces fichiers exécutables sont disponibles dans le répertoire LISA_HOME\bin.

Vérification du laboratoire de tests d'application mobile

Vous pouvez vérifier l'existence du laboratoire de tests d'application mobile via la console de serveur.

Validez les fonctionnalités mobiles de chaque ordinateur dans le laboratoire en passant le curseur de la souris sur l'icône Simulator (Simulateur). L'ensemble de fonctionnalités répertorie les appareils connectés, les images de simulateur pour iPhone et les unités virtuelles Android.



Simulation d'un test dans le laboratoire de tests d'application mobile

Procédez comme suit:

1. Recherchez le test dans le panneau Project.
2. Cliquez avec le bouton droit de la souris sur le test, alors sélectionnez StageTest (Simuler le test).

   Remarque : Veillez à sélectionner le serveur de coordination du laboratoire pour simuler le test dans le laboratoire.

Seuls les coordinateurs des laboratoires disposant de la fonctionnalité mobile sont disponibles pour les tests dans la liste prédéfinie de coordinateurs.

Simulation d'une suite de tests dans le laboratoire de tests d'application mobile

Procédez comme suit:

1. Avant de simuler une suite de tests dans le laboratoire de tests d'application mobile, veillez à ce que la suite dispose d'un coordinateur par défaut assigné. Si ce n'est pas le cas, vous serez invité à en placer un lors de la simulation de la suite.
   a. Ouvrez la suite de tests dans un éditeur.
   b. Cliquez sur l'onglet Defaults (Valeurs par défaut).
   c. Sélectionnez le coordinateur de laboratoire dans la liste des coordinateurs.
2. Pour simuler une suite, cliquez dessus avec le bouton droit de la souris dans le panneau Project.
3. Sélectionnez Run with DCM (Exécuter avec DCM).
4. Cliquez sur Stage (Simuler).

Tests automatisés par Voyager

Voyager est un explorateur d'applications, ou robot, qui examine vos applications mobiles et crée des scénarios de test. Voyager examine votre fichier d'application, puis répartit ce fichier vers tous les vecteurs - pages, liens, gestes et entrée - dans toute l'application. Ces vecteurs sont alors explorés et mis en pratique, puis les données sont générées à partir de ces vecteurs.

Voyager permet d'effectuer les actions suivantes :

- **Créer un ensemble de tests automatisés**

  Voyager vous permet d'économiser le temps dédié à l'enregistrement des tests dans l'enregistreur, de modifier des tests et de les intégrer à une suite de teste. Voyager génère tous les vecteurs dans l'application, puis crée et automatise les scénarios de test pour vous.

- **Créer un scénario de test pour une page spécifique de votre application**

  Vous pouvez identifier une zone de données spécifique à partir de Voyager et générer ensuite un scénario de test pour cette zone.

- **Ajouter des étapes à un scénario de test existant à l'aide des données de Voyager**

  Vous pouvez utiliser des données de Voyager pour ajouter une étape à un scénario de test déjà existant.

Génération de tests

Vous pouvez générer automatiquement des scénarios de test à l'aide de Voyager.

Procédez comme suit:

1. Dans le menu Actions, cliquez sur Launch Voyager (Lancer Voyager).

   La boîte de dialogue Voyager Configuration (Configuration de Voyager) s'affiche.

2. Dans le champ App (Application), saisissez ou sélectionnez l'application à laquelle Voyager doit accéder.

3. Dans le champ App Graph (Graphique d'application), entrez ou sélectionnez un nom pour le fichier de données qui contient les données de Voyager.

   Remarque : Le fichier de données pour le test est disponible sous le dossier Data (Données) du panneau Project.

4. Cliquez sur Auto Generate Tests (Générer automatiquement des tests) pour activer le Mobile Test Generator (Générateur de tests d'application mobile).

   Remarque : Les tests sont créés lors de chaque détection d'une nouvelle page et la base de données est enregistrée après l'arrêt de Voyager.

5. Dans le champ Output Folder (Dossier de sortie), entrez un emplacement pour votre fichier .appgraph, ou acceptez l'emplacement par défaut.

6. Cliquez sur Reconfigure SauceLabs (Reconfigurer SauceLabs) si le compte SauceLabs ou la clé d'accès ont été changés. Si SauceLabs n'est pas configuré, la boîte de dialogue SauceLabs s'ouvre avant le démarrage de Voyager.

7. Cliquez sur OK pour lancer Voyager.

   Le tableau de bord Voyager Dashboard s'ouvre. Comme les pages sont indexées, la page en cours s'ouvre. Le nombre de pages uniques s'affiche dans la partie supérieure.

8. Cliquez sur Stop (Arrêter) pour arrêter Voyager.

Création d'un scénario de test spécifique à une page

Une fois que Voyager est lancé et qu'un fichier de données est créé pour capturer les données de l'application, vous pouvez identifier une zone spécifique des données et générer un scénario de test pour celles-ci.

Procédez comme suit:

1. Créez un scénario de test d'application mobile (page 461).
2. Dans l'éditeur de tests, cliquez sur Create steps by recording or templating (Créer des étapes à l'aide d'un enregistrement ou d'un modèle)  .
3. Cliquez sur Record Test Case for User Interface (Enregistrer le scénario de test pour l'interface utilisateur), Generate Mobile Steps (Générer des étapes d'application mobile).

   La boîte de dialogue Page Selection (Sélection de page) s'affiche.

   Remarque : Si vous utilisez un simulateur mobile pour l'enregistrement, la fenêtre de simulateur mobile s'ouvre également. Cela se produit uniquement lorsque Voyager est en cours d'exécution. Dans ce cas, les étapes sont générées à partir de données déjà enregistrées.

   Pour remplir cette boîte de dialogue, Voyager doit être exécuté au moins une fois.
4. Sélectionnez UICatalog comme application dans la liste déroulante.
5. Sélectionnez Data/Voyager/UICatalog.appgraph comme base de données de Voyager.
6. Sélectionnez une ou plusieurs pages à partir de Voyager à inclure dans votre test.

Ajout d'étapes de test d'applications mobiles à un fichier Voyager à partir d'un scénario de test existant

Vous pouvez ajouter des étapes de test d'application mobile à un fichier Voyager à partir d'un scénario de test mobile existant.

Procédez comme suit:

1. Dans l'arborescence du projet dans le panneau Project, cliquez avec le bouton droit de la souris sur le fichier .appgraph qui contient vos données.
2. Sélectionnez Merge Test Case. (Fusionner le scénario de test).
3. Sélectionnez le scénario de test auquel vous voulez ajouter la nouvelle étape.
4. Cliquez sur OK.

   Les données de test sont fusionnées avec le fichier Voyager .appgraph.

Ajout d'étapes de test d'applications mobiles à un scénario de test à partir d'un fichier Voyager existant

Vous pouvez ajouter une étape à un scénario de test d'application mobile existant à l'aide de données de Voyager.

Procédez comme suit:

1. Ouvrez un scénario de test d'application mobile.
2. Cliquez sur Record Test Case for User Interface (Enregistrer le scénario de test pour l'interface utilisateur), Generate Mobile Steps (Générer des étapes d'application mobile).

   La boîte de dialogue Page Selection (Sélection de page) s'affiche.

   Remarque : Pour remplir cette boîte de dialogue, Voyager doit être exécuté au moins une fois.
3. Sélectionnez le fichier d'application enregistré avec Voyager.
4. Sélectionnez le fichier .appgraph qui contient l'étape que vous voulez ajouter à votre test.
5. Cliquez sur une page enregistrée.
6. Vérifiez que les pages ont été ajoutées à la base de données.

Dépannage des scénarios de test d'application mobile

- Pour consulter davantage de messages d'erreur, cliquez sur System Messages (Messages système) dans le coin inférieur gauche de DevTest Workstation. Ce volet fournit souvent des informations supplémentaires qui peuvent résoudre les problèmes.
- Pour des informations de débogage supplémentaires, définissez la propriété suivante dans le fichier LISA_HOME\logging.properties :

  log4j.logger.com.itko.lisa.mobile=DEBUG.

Chapitre 20: Fonctionnalités avancées

Ce chapitre traite des sujets suivants :

Utilisation de BeanShell dans DevTest

BeanShell (http://www.beanshell.org/) est un langage de script Java open source, léger et libre. BeanShell est une application Java qui utilise l'API Reflection pour exécuter des instructions et des expressions Java de façon dynamique. BeanShell vous évite de devoir compiler les fichiers de classe.

BeanShell vous permet de saisir de la syntaxe Java standard (instructions et expressions) dans une ligne de commande et d'afficher les résultats immédiatement. Une interface utilisateur graphique Swing est également disponible. Vous pouvez également appeler BeanShell à partir d'une classe Java (méthode utilisée dans le produit).

BeanShell est utilisé en différentes occasions :

- Pour interpréter des expressions de propriété
- Comme structure d'interprétation pour l'étape de test Java Script (Script Java)
- Comme structure d'interprétation pour l'assertion Assert by Script Execution (Affirmer par exécution de script)

Informations complémentaires :

Utilisation du langage de script BeanShell

La différence majeure entre du code Java BeanShell et du code Java compilé réside dans le système de saisie. Java est fortement typé, tandis que BeanShell prend en charge une saisie plus flexible dans son environnement de scripts. Vous pouvez toutefois imposer une saisie stricte dans BeanShell si vous le voulez.

BeanShell prend en charge une saisie flexible qui vous permet d'écrire des scripts BeanShell similaire à du code Java standard. Vous pouvez également écrire des scripts plus proches du langage de script traditionnel, comme Perl ou JavaScript, en maintenant la structure de syntaxe Java.

Si une variable est typée, BeanShell respecte le type et le vérifie. Dans le cas contraire, BeanShell signale uniquement une erreur si vous tentez d'utiliser le type de la variable de manière incorrecte.

Les fragments Java suivants sont tous valides dans BeanShell :

```
foo = "Foo";
four = (2+2) * 2 / 2.0;
print(foo + " = " + four);
.
.
hash = new Hashtable();
date = new Date();
hash.put("today", date);
.
.
```

BeanShell vous permet de déclarer, puis d'utiliser des méthodes. Les arguments et les types de retour peuvent également être typés de manière flexible :

Typé

```
int addTwoNumbers(int a, int b){
return a + b;
}
```

Typé de manière flexible

```
add (a,b){
return a + b;
}
```

Dans le deuxième exemple, l'expression suivante est correctement prise en charge :

```
sumI = add (5,7);
sumS = add("DevTest " , "Rocks");
sumM = add ("version ", 2);
```

BeanShell fournit également une bibliothèque de commandes qui facilitent son utilisation.

Exemples de commandes :

- **source() :** permet de lire un script BeanShell (bsh).
- **run() :** permet d'exécuter un script bsh.
- **exec() :** permet d'exécuter une application native.
- **cd()**, **copy()**, etc. : commandes de shell similaires aux commandes UNIX.
- **print() :** permet d'imprimer un argument en tant que chaîne.
- **eval()** : permet d'évaluer l'argument de chaîne en tant que code.

Pour plus d'informations, consultez le manuel BeanShell User Guide sur le site http://www.beanshell.org/.

Vous pouvez également y obtenir BeanShell, le code source et l'outil Javadoc complet.

Utilisation de BeanShell en tant qu'interpréteur autonome

Vous pouvez utiliser BeanShell en tant qu'interpréteur autonome avec d'autres applications que DevTest. Vous pouvez télécharger BeanShell à partir du site www.beanshell.org. Le fichier téléchargé consiste en un petit fichier JAR unique nommé bsh-xx.jar (xx est le numéro de version, actuellement version 2.0). Ajoutez le fichier JAR à la variable classpath.

Vous pouvez utiliser BeanShell dans les configurations suivantes :

- **A partir d'une ligne de commande :** java bsh.Interpreter [script name] [args]
- **A partir de l'interface utilisateur graphique BeanShell :** java bsh.Console
- **A partir d'une classe Java :**

```
Import bsh.Interpreter;
.
.
Interpreter I = new Interpreter();
i.set ("x",5);
i.set("today", new Date());
Date d = (Date)i.get("date");
i.eval("myX = x * 10");
System.out.println(i.get("myX"));
```

Utilisation de BeanShell dans DevTest

L'interpréteur BeanShell est utilisé dans l'étape Java Script (Script Java) et dans l'assertion Assert by Script Execution (Affirmer par exécution de script). Ces deux éléments affichent également les objets Java de DevTest et l'état actuel de DevTest (propriétés). Vous disposez donc d'un environnement très pratique pour ajouter des fonctionnalités personnalisées. Vous pouvez utiliser les objets Java exposés pour interroger et modifier l'état actuel du test. Par exemple, vous pouvez lire, modifier et créer des propriétés DevTest dans vos scripts.

Pour commencer, familiarisez-vous avec la classe **TestExec** dans DevTest. Des informations sur la classe TestExec et d'autres classes sont disponibles à la section *Utilisation du kit SDK.*

DevTest utilise également BeanShell dans la notation de propriété lorsqu'un signe égal est présent. Par exemple :

```
{{= new Date()}}
```

Cette expression de propriété est interprétée à l'aide de BeanShell.

Utilisation d'utilitaires de date

La classe **com.itko.util.DateUtils** inclut un nombre de fonctions d'utilitaire de date en tant que méthodes statiques. Ces fonctions renvoient toutes la date sous la forme d'une chaîne. Vous pouvez utiliser ces fonctions dans des expressions de paramètre ou dans l'étape Java Script (Script Java).

```
com.itko.util.DateUtils.formatDate(Date date, String format)
com.itko.util.DateUtils.formatCurrentDate(String format)
com.itko.util.DateUtils.formatCurrentDate(int offsetInSec, String format)
com.itko.util.DateUtils.rfc3339(Date date)
com.itko.util.DateUtils.rfc3339()
com.itko.util.DateUtils.rfc3339(int offsetInSec)
com.itko.util.DateUtils.samlDate(Date date)
com.itko.util.DateUtils.samlDate()
com.itko.util.DateUtils.samlDate(int offsetInSec)
```

Par exemple, si un appel de service Web utilise une chaîne de date formatée et le serveur a 2 minutes de retard, vous pouvez utiliser :

```
=com.itko.util.DateUtils.formatCurrentDate(-120,"yyyy-MM-dd'T'HH:mm:ss.SSSZ")
```

La chaîne 2007-11-22T13:30:37.545-0500 est générée, ce qui correspond à l'heure actuelle moins 120 secondes formatées selon les paramètres spécifiés.

Avec l'utilitaire RFC 3339, la date est légèrement différente de la date générée par le formateur de date Java par défaut. Si vous avez besoin d'une date RFC 3339 stricte, vous pouvez utiliser les fonctions rcf3339 :

```
=com.itko.util.DateUtils.rfc3339()
```

La chaîne 2007-11-22T13:30:37.545-05:00 est générée.

Les dates SAML sont formatées au format **yyyy-MM-dd'T'HH:mm:ss'Z'**. Les fonctions samlDate sont simplement des aides qui vous évitent de devoir vous rappeler de ce format de chaîne lorsque vous utilisez les API formatDate.

Pour plus d'informations, consultez les ressources suivantes :

http://download.oracle.com/javase/1.5.0/docs/api/java/text/SimpleDateFormat.html

http://tools.ietf.org/html/rfc3339#section-5.6

Exemple de bac à sable du chargeur de classes

La classe Java suivante est un exemple simple d'une classe que vous ne pouvez pas exécuter en mode multithread ou multi-utilisateurs, car elle accède et modifie une variable statique.

```
public class NeedsASandbox \{

static \{

System.out.println("This is my static initializer. You will see
this many times.");

static String s;

public NeedsASandbox() \{\};

public void setS(String s)\{

this.s = s;

public String getS() \{

return s;
```

Cette classe doit être exécutée dans un bac à sable de chargeur de classes.

Supposons, par exemple, que vous devez exécuter cette classe dans DevTest et créer un test de charge de dix utilisateurs. Si vous n'utilisez pas le bac à sable de chargeur de classes, l'expression System.out.println s'affiche une seule fois et la valeur de s est incorrecte. Ce résultat est obtenu du fait que tous les utilisateurs sont exécutés dans un même chargeur de classes. Dans ces circonstances, un échec se produit pour cette classe. Si s'agit de la fonction de l'application appropriée, utilisez le bac à sable du chargeur de classes pour obtenir un résultat correct.

Lorsque vous créez le compagnon Class Loader Sandbox (Bac à sable du chargeur de classes) et que DevTest simule les dix utilisateurs, l'expression de texte dans le code s'affiche dix fois. DevTest crée dix chargeurs de classes distincts et instancie cette classe dix fois. Il existe donc dix instances distinctes de la variable de classe static string s. Cette fonctionnalité permet d'exécuter la logique d'application, non sécurisée, dans des tests d'utilisateur simultanés.

Le compagnon Class Loader Sandbox est utile uniquement si les trois conditions suivantes sont réunies :

- Vous testez un objet POJO avec DevTest.
- L'objet POJO comprend des membres statiques.
- Vous effectuez le test avec plusieurs utilisateurs virtuels.

Fonctionnalité ICT (In-Container testing)

La fonctionnalité ICT (In-Container Testing, test dans un conteneur), lorsqu'elle utilise des proxys distants dans DevTest, est disponible dans les étapes de test Enterprise JavaBean Execution (Exécution d'un objet EJB) et Dynamic Java Execution (Exécution Java dynamique).

Cette fonctionnalité permet de tester dans un conteneur les objets EJB locaux et les objets Java arbitraires, c'est-à-dire, tous les objets disponibles dans la variable classpath du conteneur pour l'application testée.

Les objets RMI et EJB sont pris en charge en tant que protocoles d'objet distants.

La fonctionnalité ICT utilise deux modes de connexion : EJB et RMI.

Cette section comprend les rubriques suivantes :

Accès en mode EJB

Pour tester des objets dans un conteneur J2EE standard, une session Enterprise JavaBean (EJB) avec état est utilisée. L'objet EJB est livré avec le programme d'installation en tant que répertoire EAR éclaté nommé **lisa-remote-object-manager.ear** et fichier JAR autonome **LISARemoteObjectManagerEJB.jar** dans le répertoire LISA_HOME/incontainer/ejb, où LISA_HOME représente le répertoire d'installation de DevTest.

Cet objet EJB doit être déployé avec l'application J2EE dans le conteneur J2EE et être accessible via le JNDI sous le nom **LISARemoteObjectManagerEJB**. Le déploiement d'un objet EJB varie selon le conteneur J2EE utilisé et la documentation du fournisseur peut vous aider en cas de problème.

Si votre application J2EE utilise un chargeur de classes isolé, vous devez inclure l'objet EJB pour le test ICT dans votre application en modifiant les descripteurs de déploiement XML pour inclure l'objet EJB et ses dépendances.

JBoss

1. Vérifiez que vous pouvez déployer le fichier EAR éclaté pour le test ICT dans JBoss en copiant le répertoire **$LISA\incontainer\ejb\lisa-remote-object-manager.ear to $JBOSS\server\default\deploy** ou dans un autre répertoire de déploiement approprié.
2. JBoss identifiera le nouveau fichier EAR et le déploiera sans erreurs. Vérifiez que vous pouvez vous connecter à l'objet EJB à partir de DevTest Workstation.
3. Une fois le fichier EAR déployé dans une configuration autonome, essayez d'intégrer l'objet EJB du test ICT à votre application J2EE. Pour cela, copiez le contenu de **$LISA\incontainer\ejb\lisa-remote-object-manager.ear** et incluez-le dans le fichier EAR existant de votre application. Vous pouvez également créer un fichier EAR qui inclut l'application J2EE combinée avec ces fichiers.

Pour en savoir plus sur la procédure de modification de vos descripteurs de déploiement d'application afin d'inclure le test ICT et ses dépendances, consultez le fichier **application.xml** du descripteur XML de déploiement d'application.

Les éléments XML <module> sont utilisés pour indiquer la présence de l'objet EJB du test ITC et les dépendances de fichier JAR Java.

WebLogic

1. Le premier test à effectuer sur WebLogic consiste à déployer le fichier EAR éclaté du test ITC dans WebLogic, par exemple, à l'aide de l'interface utilisateur de la console d'administration WebLogic. Si le serveur WebLogic est en mode développement, vous pouvez également copier le répertoire **$LISA\incontainer\ejb\lisa-remote-object-manager.ear** dans le répertoire **autodeploy** du serveur et WebLogic procède automatiquement au déploiement du fichier EAR. Le fichier EAR est déployé sans erreurs.
2. Vérifiez que vous pouvez vous connecter à l'objet EJB à partir de DevTest Workstation.
3. Une fois le fichier EAR déployé dans une configuration autonome, essayez d'intégrer l'objet EJB du test ICT à votre application J2EE. Pour cela, copiez le contenu de **$LISA\incontainer\ejb\lisa-remote-object-manager.ear** et incluez-le dans le fichier EAR de votre application. Vous pouvez également créer un nouveau fichier EAR qui inclut l'application J2EE combinée avec ces fichiers.

Accès en mode RMI

Pour les applications Java personnalisées, vous pouvez modifier le code source de votre application afin de permettre l'intégration de la fonctionnalité ICT. Vous pouvez effectuer un test ICT en associant l'objet de serveur distant **LISARemoteObjectManagerRMIServer** au registre RMI. Cet objet accepte les connexions à partir de DevTest Workstation pour le test ICT.

Par exemple, vous pouvez inclure le code suivant dans votre application personnalisée afin de lier le serveur d'objet distant du test ICT via RMI :

```
LISARemoteObjectManagerRMIServer remoteObjectManagerServer = new
LISARemoteObjectManagerRMIServer();Registry registry =
LocateRegistry.createRegistry(port);registry.bind("LISARemoteObjectManager",
remoteObjectManagerServer);
```

Remarque : Le nom RMI **LISARemoteObjectManager** doit être entré tel quel pour que le test ICT fonctionne.

DevTest Workstation tente de se connecter à votre application via l'URL RMI **rmi://***hostname***:***port***/LISARemoteObjectManager**.

Le nom d'hôte et le port sont des variables. Vous pouvez les modifier dans votre application de test et les configurer dans DevTest Workstation.

Consultez l'exemple d'application de serveur sous $LISA/incontainer/rmi/example. A l'aide d'une fenêtre de console, vous pouvez exécuter l'exemple de serveur à l'aide de la commande java -jar ExampleServer.jar.

Remarque : Pour que cette commande fonctionne, vous devez l'exécuter à partir du répertoire **$LISA/incontainer/rmi/example**. Consultez l'exemple de code source du répertoire **$LISA/incontainer/rmi/example/src**.

Test de l'installation de l'ICT

Une fois que la partie serveur du test ICT est en cours d'exécution, testez la connexion à partir de DevTest Workstation.

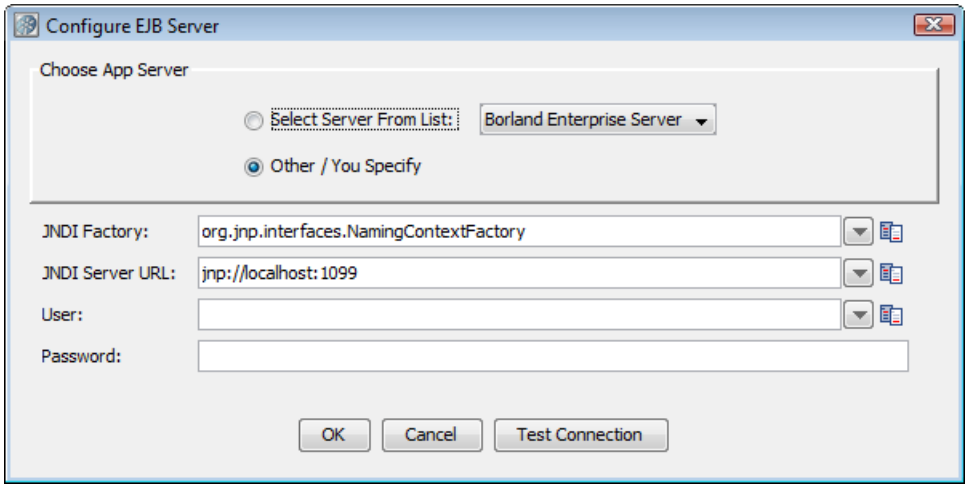
Pour cela, procédez comme suit :

1. Ouvrez DevTest Workstation.
2. Créez un scénario de test nommé **Test de connexion ICT**.
3. Dans ce scénario de test, créez une étape de test Dynamic Java Execution (Exécution Java dynamique).
4. Dans l'éditeur de la nouvelle étape de test, sélectionnez l'option Remote (Distant).
5. Dans la liste déroulante Remote Container Type (Type de conteneur distant), définissez votre protocole de connexion, EJB ou RMI.

Type de conteneur distant : EJB

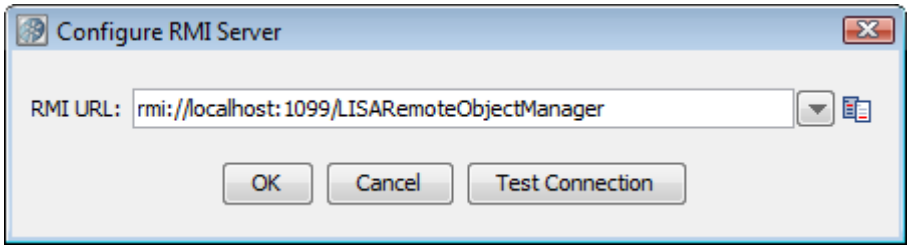
Si vous utilisez EJB comme protocole de connexion, cliquez sur Configure (Configurer) pour entrer les paramètres de configuration du protocole.

Par exemple, dans la boîte de dialogue Configure EJB Server (Configurer le serveur EJB), saisissez le nom d'hôte ou l'adresse IP, et le numéro de port du conteneur J2EE exécutant le test ICT.



Type de conteneur distant : RMI

Si vous utilisez RMI comme protocole de connexion, examinez l'exemple de boîte de dialogue Configure RMI Server (Configurer un serveur RMI) que vous pouvez utiliser pour spécifier les paramètres RMI.



Une fois que la connexion au serveur ICT a été testée, entrez **java.util.Date** dans le champ Make New Object of Class (Créer un objet de classe), puis cliquez sur Construct/Load Object (Générer/charger un objet) pour créer une nouvelle instance de la classe java.util.Date dans un conteneur.

L'installation est terminée et vous pouvez utiliser la fonctionnalité ICT.

Dépendances

Les classes de serveurs ICT dépendent des bibliothèques tierces suivantes :

- BeanShell 2.0b4 (bsh-2.0b4-lisa-remote-object-manager.jar).

  Remarque : Le fichier JAR d'origine a été modifié pour que les scripts .bsh soient supprimés. Ces scripts sont connus pour causer des problèmes avec les conteneurs J2EE qui vérifient les fichiers JAR et les exécutent automatiquement.
- XStream 1.1.2 (xstream-1.1.2.jar)
- Log4j 1.2.13 (log4j-1.2.13.jar)
- Jakarta Commons Logging 1.0.2 (commons-logging.jar)

Ces dépendances sont intentionnellement distribuées sous la forme de fichiers distincts des fichiers JAR ICT de DevTest dans le but de faciliter l'intégration de la fonctionnalité ICT. L'application que vous testez pouvant déjà inclure une partie ou la totalité de ces bibliothèques dans la variable classpath, ajouter ces dépendances à la variable classpath peut ne pas être nécessaire.

Génération de fichiers DDL

Définir les propriétés suivantes dans **local.properties** permet de créer des fichiers DDL pour la génération de rapports et le VSE :

- **eclipselink.ddl-generation=create-tables**
- **eclipselink.ddl-generation.output-mode=sql-script**
- **eclipselink.target-database=Oracle**

Pour créer des fichiers DDL pour l'agent, CAI et le service de validation en continu, utilisez les commandes suivantes :

- **java -jar LisaAgent.jar -ddl oracle** : permet de générer le fichier DDL Oracle pour CAI.
- **java -jar LisaAgent.jar -ddl mysql** : permet de générer le fichier DDL MySQL pour CAI.

Annexe A: Annexe A : fichier de propriété DevTest (lisa.properties)

Le fichier de propriété **lisa.properties** stocke les informations d'initialisation et de configuration.

Les fichiers de propriété sont stockés dans le répertoire d'installation de DevTest.

Vous pouvez afficher le contenu de ce fichier dans DevTest Workstation.

Remarque : N'ajoutez pas de propriétés personnalisées à ce fichier, car CA Technologies se réserve le droit de remplacer ce fichier à tout moment.

Pour ouvrir le fichier **lisa.properties**, sélectionnez System (Système), Edit Properties (Modifier les propriétés) dans le menu principal.

Ce chapitre traite des sujets suivants :

Liste de chemins séparés par des virgules pour l'outil Javadoc et le code source

Ces chemins sont utilisés pour afficher la documentation de classe et de paramètre. Le chemin docpath accepte les répertoires et les URL, qui correspondent à des chemins de base de l'outil Javadoc. L'exemple suivant inclut des documents de kit de développement Java provenant d'un site Web. Toutefois, notez que les sites Web ne sont pas recommandés, car ils supposent des délais.
lisa.java.docPath=LISA_HOME\examples\javadoc,[http://java.sun.com/j2se/1.3/docs/api/

lisa.java.docPath=LISA_HOME\examples\javadoc

lisa.java.sourcePath=LISA_HOME\examples\src

Le chemin d'accès sourcepath accepte les répertoires en tant que chemins de base et les fichiers JAR ou ZIP source.

lisa.axis.compiler.version=1.4

Version 1.4 du compilateur lisa.axis.compiler.

Propriétés du système

file.encoding

Codage des fichiers écrits ou lus par DevTest

Par défaut : UTF-8

lisa.supported.html.request.encodings

Pour inclure les codages que vous voulez prendre en charge dans l'étape HTTP/HTML Request (Demande HTTP/HTML), modifiez la liste des encodages séparés par des virgules. La machine virtuelle Java sous-jacente doit également prendre en charge tous les codages de cette liste. Si une page Web utilise un codage non pris en charge dans la liste, le codage sera remplacé par le codage par défaut de DevTest (clé file.encoding dans **lisa.properties**). De plus, si un codage n'est pas sélectionné lors de la création d'une étape HTTP/HTML Request (Demande HTTP/HTML), le codage par défaut de DevTest est utilisé.

Valeur par défaut : ISO-8859-1, UTF-8, Shift_JIS, EUC-JP

lisa.supported.jres

Valeur par défaut : 1.7

org.jfree.report.LogLevel

Valeur par défaut : Error

javax.xml.parsers.DocumentBuilderFactory

Valeur par défaut : org.apache.xerces.jaxp.DocumentBuilderFactoryImpl

javax.xml.parsers.SAXParserFactory

Valeur par défaut : org.apache.xerces.jaxp.SAXParserFactoryImpl

javax.xml.transform.TransformerFactory

Valeur par défaut : org.apache.xalan.processor.TransformerFactoryImpl

Propriétés du serveur

lisa.net.bindToAddress

Adresse IP sur laquelle les écoutes sont effectuées. Par défaut, les écoutes sont effectuées sur toutes les adresses IP de l'application. Vous pouvez limiter cette valeur à une adresse IP particulière. Pour empêcher que d'autres ordinateurs se connectent, mais permettre des connexions locales, définissez la valeur sur *127.0.0.1* ou sur *localhost.*

Propriétés OS X

apple.awt.brushMetalLook

Valeur par défaut : false

apple.awt.brushMetalRounded

Valeur par défaut : false

apple.laf.useScreenMenuBar

Valeur par défaut : true

apple.awt.showGrowBox

Valeur par défaut : true

com.apple.mrj.application.growbox.intrudes

Valeur par défaut : true

com.apple.macos.smallTabs

Valeur par défaut : true

com.apple.mrj.application.apple.menu.about.name

Par défaut : LISA

com.apple.mrj.application.live-resize

Valeur par défaut : true

Notifications de mise à jour

lisa.update.every=1

Contrôle la fréquence à laquelle DevTest vérifie la disponibilité d'une version plus récente à télécharger. Pour désactiver la vérification, laissez la valeur vide. Les autres valeurs valides sont les suivantes :

- 0 (vérification à chaque démarrage)
- 1 (vérification une fois par jour)
- 2 (vérification tous les deux jours), etc.

lisa.update.URL=http://www.itko.com/download/ga/

Valeurs par défaut de base

lisa.testcase=test.xml

Valeur par défaut lors de l'exécution directe d'un test à partir de la classe com.itko.lisa.test.TestCase.

lisa.registry=registry.xml

Valeur par défaut lors de l'exécution directe d'un test à partir de la classe com.itko.lisa.test.TestCase.

lisa.runName=Ad-hoc Run

Valeur par défaut lors de l'exécution directe d'un test à partir de la classe com.itko.lisa.test.TestCase.

lisa.registryName=registry

Nom par défaut du registre à associer et nom par défaut du registre lorsque vous le démarrer sans spécifier de nom.

lisa.coordName=coordinator

Nom par défaut du serveur de coordination lorsque vous le démarrez sans spécifier de nom explicite et lorsque l'utilitaire Test Runner (Exécuteur de tests) requiert un serveur de coordination, mais qu'aucun n'est spécifié dans la ligne de commande.

lisa.simulatorName=simulator

Nom par défaut d'un démon de simulateur si vous n'en fournissez aucun dans la ligne de commande.

lisa.vseName=VSE

Nom par défaut du serveur d'environnement virtuel lorsqu'il est démarré sans un nom explicite.

lisa.defaultRegistry.pulseInterval=30

Intervalle de journal de statut pour le registre. La valeur par défaut est 30 secondes.

lisa.coordinator.pulseInterval=30

Intervalle de journal de statut pour le coordinateur. La valeur par défaut est 30 secondes.

lisa.simulator.pulseInterval=30

Intervalle de journal de statut pour le simulateur. La valeur par défaut est 30 secondes.

lisa.vse.pulseInterval=30

Intervalle de journal de statut pour VSE. La valeur par défaut est 30 secondes.

lisa.server.projectmap.refresh.pulseInterval=600

Intervalle après lequel les serveurs DevTest (coordinateur et simulateur) actualisent le mappage des noms de projet vers les chemins d'accès au fichier.

lisa.defaultRegistryConnectionTimeoutSeconds=90

Valeur du délai d'expiration en secondes que les coordinateurs et les simulateurs utilisent lors de la connexion à un registre DevTest. La valeur 0 indique un délai d'expiration infini, c'est-à-dire que la tentative de connexion est effectuée indéfiniment.

lisa.regex.helper.tutorial.url=http://download.oracle.com/javase/tutorial/essential/regex/

URL à afficher pour le didacticiel d'expression régulière, dans la fenêtre de l'assistant pour les expressions régulières.

lisa.hooks=com.itko.lisa.files.SampleHook

Permet d'enregistrer des hooks dans DevTest. Ces valeurs sont séparées par des virgules.

lisa.project.ignore=CVS|SCCS|RCS|rcs|\.DS_Store|\.svn|vssver\.scc|vssver2\.scc|\.sbas|\.IJI\..*|.*\.pyc|.*\.pyo|\.git|.*\.hprof|_svn|\.hg|.*\.lib|.*~|__pycache__|\.bundle|.*\.rbc

Cette propriété est une expression régulière Java qui vous permet de masquer les types de fichiers que vous ne voulez pas afficher dans l'arborescence Project (Projet). Par défaut, un ensemble de fichiers de contrôle connus pour des produits tiers sont masqués. Pour afficher des fichiers masqués par défaut, modifiez l'expression régulière.

Propriétés de clés d'en-tête HTTP

La liste suivante contient les valeurs par défaut pour les clés d'en-tête dans le cadre de la prise en charge HTTP. Pour tirer profit de la modification, supprimez ou modifiez-les pour tous les tests. Pour les modifier pour un test ou l'exécution d'une transaction HTTP, utilisez des directives d'en-tête TestNode.

ice.browser.http.agent

Valeurs : compatible, MSIE 6.0 et Windows NT 5.0

Valeur par défaut : Mozilla/4.0

lisa.http.header.0.key

Valeur par défaut : Pragma

lisa.http.header.0.value

Valeur par défaut : no-cache

lisa.http.header.1.key

Par défaut : Cache-Control

lisa.http.header.1.value

Valeur par défaut : no-cache

lisa.http.header.0.key

Valeur par défaut : Accept

lisa.http.header.0.value

Valeur par défaut : image/gif, image/x-xbitmap, image/jpeg

lisa.http.header.1.key

Par défaut : Accept-Language

lisa.http.header.1.value

Valeur par défaut : en

lisa.http.header.2.key

Par défaut : Accept-Charset

lisa.http.header.2.value

Valeur par défaut : iso-8859-1,*,utf-8

lisa.http.header.3.key

Par défaut : User-Agent

lisa.http.header.3.value

Valeur par défaut : Mozilla/4.0, MSIE 6.0, Windows NT 5.0

Propriétés d'éditeur de champs HTTP

La liste suivante contient des valeurs par défaut pour des champs qui apparaissent dans le HTTP Field Editor (Editeur de champs HTTP). N'incluez pas la valeur Authentication, car l'éditeur l'ajoute automatiquement.

lisa.gui.http.fieldNames

Valeurs : Accept, Accept-Language, User-Agent, Connection

lisa.webrecorder.textMIMEs

Valeurs : html, text, magnus-internal, application/pdf

lisa.webrecorder.notTextMIMEs

Valeurs : css, script

lisa.webrecorder.alwaysIgnore

Valeurs : .gif, .jpg, .jpeg, .css, .js, .ico

lisa.web.ntlm

Active l'authentification NTLM dans l'utilitaire Test Runner (Exécuteur de tests).

Valeur par défaut : true

Paramètres d'exécution de scénario de test

lisa.hotDeploy=C\Projects\Lisa\custom_classes

Ce paramètre indique au chargeur de classes intégré à DevTest l'emplacement dans lequel rechercher des classes personnalisées. Par défaut, cet emplacement est le répertoire $LISA_HOME\hotDeploy.

lisa.overloadThreshold=1000

TestNode tente de déterminer si le simulateur a subi un arrêt brutal en vérifiant la durée réelle de mise en veille durant le délai de réflexion par opposition à la durée du délai de réflexion supposée. Ce paramètre est la durée supplémentaire acceptable ajoutée au délai de réflexion avant d'envoyer un avertissement TestEvent indiquant que le simulateur est surchargé. La valeur par défaut de 1000 suppose que si le simulateur est mis en veille 1 seconde de plus que ce qu'il est supposé, (sans doute parce que l'UC est saturée), l'événement TestEvent est déclenché.

lisa.webservices.encode.empty.xmlns=true

Certaines piles de serveur de services Web requièrent des chaînes xmlns vides dans la demande SOAP (par exemple, jbossWs). D'autres, comme Amazon, ne fonctionnent pas avec des chaînes xmlns vides. Modifiez cette propriété en fonction de la pile à appeler.

lisa.webservices.encode.version=1.1

Définissez la version de codage à appliquer à la génération de stub client. La valeur par défaut est 1.1.

lisa.tm.sys.min.millis=0

lisa.tm.sys.max.millis=0

Délai de réflexion par défaut pour les nouvelles étapes système, en millisecondes. Les étapes système incluent les étapes de sous-processus, les étapes continues, les étapes continues (silencieuses), les étapes d'échec et les étape de fin.

lisa.numFilters.warning=100

lisa.numAsserts.warning=100

Parfois, les filtres et les assertions sont ajoutés de façon dynamique à des étapes de test. Dans un test de charge, cela peut signifier plusieurs milliers d'assertions ou de filtres. Les deux nombres précédents correspondent au seuil avant que le journal génère un message de niveau WARN.

lisa.exception.on.num.exceeded=true

Lorsque le seuil est dépassé, permet de déterminer si une exception TestDefException doit être renvoyée (et le test arrêté).

lisa.urltrans.encode.queryparams=false

Définissez cette propriété sur false pour que les paramètres de requête dans une URL HTTP/HTML ne soient pas décodés/codés lors de l'exécution de l'étape.

lisa.generic.url.decoder=true

Définissez cette propriété sur true lorsque vous ouvrez un scénario de test dans DevTest 7.1.1 ou version ultérieure comprenant une étape HTTP/HTML Request (Demande HTTP/HTML) créée avec la version 7.0.0 ou une version antérieure.

Personnalisations du traitement de l'événement TestEvent

lisa.perfmon.snmp.port=1161

L'outil StatKeeper (Outil de conservation des statistiques) peut charger une classe d'intégration PerfMon qui encapsule ou implémente un moniteur spécifique de plate-forme, comme Windows PerfMon, JMX, SNMP, etc. DevTest fournit une classe de DLL PerfMon et une classe SNMP pour prendre en charge la génération de statistiques sur le moniteur de performances Windows ou sur un agent SNMP, jamais les deux. Pour plus d'informations, reportez-vous à la documentation SNMP. Le port habituel pour le protocole SNMP est le port 161, mais vous devez être connecté avec le compte root pour pouvoir l'utiliser.

lisa.perfmon.class=com.itko.lisa.stats.snmp.SnmpPerfmon

Lorsqu'un outil est disponible pour l'envoi de données de système d'exploitation natives vers un moniteur de performances, implémentez une classe qui peut envoyer des données DevTest vers cet outil et placez-y le nom de classe.

lisa.perfmon.dll=/c:/Projects/Lisa/PerfmonJNI/LISAPerfmonJNI/Debug/LISAPerfmonJNI.dll

L'intégration de Windows PerfMon dans l'outil StatKeeper comprend un paramètre DLL pour l'implémentation Windows (natif). Entrez le chemin complet ou le fichier dans ce paramètre. Ce paramètre est requis uniquement si vous utilisez PerfmonStatKeeperWindows.

Les simulateurs utilisent un thread et une file d'attente distincts pour envoyer des événements TestEvent au coordinateur, afin de réduire le trafic RMI. Ce sont les seuils que le thread de démon observent pour envoyer des événements. La taille minimum ou le délai d'attente maximum est utilisé pour déclencher les événements.

lisa.eventPoolPoll=250

Fréquence de vérification de la taille de file d'attente ou de si la durée d'attente maximum a été dépassée (en millisecondes).

lisa.eventPoolSize=64

Taille maximum de la file d'attente avant l'envoi d'un événement.

lisa.eventPoolMaxWait=1000

Durée d'attente maximum d'un événement non envoyé.

Propriétés de l'éditeur/gestionnaire de tests

gui.show.memory.status

Valeur par défaut : false

lisa.screencap.delay.seconds

Valeur par défaut : 6

lisa.screencap.dir

Valeur par défaut : LISA_HOME\screens

lisa.screen.cap.prefix

Valeur par défaut : lisa-screencap-

lisa.earsubdir.endingnamepart

Valeur par défaut : -contents

lisa.model.editor.inspector.scale

Cette propriété définit l'échelle (la taille de police principalement) pour des éléments du modèle et les inspecteurs d'étape de l'éditeur de modèles principal. La valeur 1.0 correspond à 12 points. Déterminez une valeur pour cette propriété en divisant la taille appropriée en points par 12. Par exemple, 11 points correspond à 11/12 = 0.92, 10 points à 10/12 = 0.83 (la valeur par défaut), 14 points à 14/12 = 1.17, etc.

Valeur par défaut : 0.83

lisa.stats.decimalFormat

Certaines mesures intégrées (étapes par seconde) utilisent un format DecimalFormat Java pour afficher des valeurs à virgule flottante. Si vous ne voulez pas préciser de décimal, définissez cette valeur comme ###### ou sélectionnez une valeur à partir d'une plage d'affichages. Pour plus d'informations, consultez la page http://download.oracle.com/javase/6/docs/api/java/text/DecimalFormat.html.

Valeur par défaut : ###,###.#

lisa.editor.custJavaNodeEditor.classes

Liste séparée par des virgules de tous les noeuds de test Java personnalisés à inclure dans le scénario de test.

lisa.editor.combined.report.type

lisa.gui.log4jfmt

Ce paramètre régit le formatage des messages de la fenêtre System Messages (Messages système).

Valeur par défaut : %-5p - %m%n

lisa.editor.http.recorderPort

L'enregistreur HTTP est associé à ce port.

Valeur par défaut : 8010

lisa.editor.proxy.webProxySupport

Valeur par défaut : on

Paramètres du serveur J2EE

lisa.prefill.jndiNames=	JBOSS=org.jnp.interfaces.NamingContextFactory Weblogic=weblogic.jndi.WLInitialContextFactory Websphere=com.ibm.websphere.naming.WsnInitialContextFactory Borland Enterprise Server=com.inprise.j2ee.jndi.CtxFactory iPlanet/Sun AS=com.sun.jndi.cosnaming.CNCtxFactory
lisa.prefill.jndiUrlPrefix=	JBOSS=jnp:// Weblogic=t3:// Websphere=iiop:// Borland Enterprise Server=iiop:// iPlanet/Sun AS=iiop://
lisa.prefill.jndiDefPort=	JBOSS=1099 Weblogic=7001 Websphere=2809 Borland Enterprise Server=1099 iPlanet/Sun AS=1099
lisa.prefill.jndiNeedsClass=	JBOSS=false Weblogic=false Websphere=true Borland Enterprise Server=true iPlanet/Sun AS=true
lisa.prefill.jndiFactories=	org.jnp.interfaces.NamingContextFactory weblogic.jndi.WLInitialContextFactory com.ibm.websphere.naming.WsnInitialContextFactory com.webmethods.jms.naming.WmJmsNamingCtxFactory com.tibco.tibjms.naming.TibjmsInitialContextFactory com.inprise.j2ee.jndi.CtxFactory com.sun.jndi.cosnaming.CNCtxFactory fiorano.jms.runtime.naming.FioranoInitialContextFactory
lisa.prefill.jndiServerURLs=	jnp://SERVER:1099&t3://SERVER:7001 iiop://SERVER:PORT tibjmsnaming://SERVER:7222&iiop://SERVER:PORT iiop://localhost:9010&wmjmsnaming://Broker #1@SERVER:PORT/JmsAdminTest
lisa.editor.URLTransEditor.protos=	http,https
lisa.editor.URLTransEditor.hosts=	
lisa.editor.URLTransEditor.ports=	80,443

lisa.editor.URLTransEditor.files=	
lisa.prefill.jdbc.names=	Oracle SQL Server WLS Oracle JDataStore Sybase DB2 MySQL Derby
lisa.prefill.jdbc.jdbcDrivers=	oracle.jdbc.driver.OracleDriver com.microsoft.sqlserver.jdbc.SQLServerDriver com.microsoft.jdbc.sqlserver.SQLServerDriver weblogic.jdbc.oci.Driver com.borland.datastore.jdbc.DataStoreDriver com.sybase.jdbc.SybDriver com.ibm.db2.jcc.DB2Driver org.gjt.mm.mysql.Driver org.apache.derby.jdbc.ClientDriver
lisa.prefill.jdbc.jdbcConnectionURLs=	jdbc:oracle:thin:@SERVER:1521:SIDNAME jdbc:sqlserver://SERVER:PORT;databasename=DBNAME jdbc:microsoft:sqlserver://SERVER:PORT jdbc:weblogic:oracle:TNSNAME jdbc:borland:dslocal:DBNAME jdbc:sybase:Tds:SERVER:PORT/DBNAME jdbc:db2://SERVER:PORT/DBNAME jdbc:mysql://SERVER:PORT/DBNAME jdbc:derby://DBSERVER:DBPORT/DBNAME

Informations de navigateur natives à utiliser pour l'interprétation interne

lisa.internal.browser.on=yes

lisa.internal.browser.win=com.itko.lisa.web.ie.IEUtils

lisa.internal.browser.osx=com.itko.lisa.web.jxbrowser.JxBrowserUtils

lisa.internal.browser.linux=com.itko.lisa.web.jxbrowser.JxBrowserUtils

lisa.internal.browser.sol-sparc=null

lisa.internal.browser.imgs=false

lisa.internal.browser=msie

WR type: mozilla, safari, msie

lisa.internal.browser.swing.heavy=false

lisa.internal.browser.usejsinviews=yes

lisa.example.wsdls=http://localhost:8080/itko-examples/services/UserControlService?wsdl

Propriétés du moniteur/gestionnaire de tests

monitor.events.maxrows

Nombre maximum de lignes d'événement conservées dans le gestionnaire de tests lors de l'exécution du test. Le nombre d'objets conservés correspond à deux fois cette valeur.

Valeur par défaut : 500

lisa.tm.sysmess.size

Taille maximum de la fenêtre de messages système. La mémoire consommée correspond à deux fois cette valeur.

Valeur par défaut : 10240

Modèles de générateur de chaîne intégrés

lisa.patterns.stringgenerator.types=&Phone=(DDD)DDD-DDDD, &SSN=DDD-DD-DDDD, &Date=Lll-DD-DDDD, &Zip=D*(5)

Informations sur JMX

lisa.jmx.types	com.itko.lisa.stats.jmx.JSE5Connection com.itko.lisa.stats.jmx.TomcatConnection com.itko.lisa.stats.jmx.JBossConnection com.itko.lisa.stats.jmx.JSR160RMIConnection com.itko.lisa.stats.jmx.WeblogicConnector com.itko.lisa.stats.jmx.Weblogic9Connector com.itko.lisa.stats.jmx.WebsphereSOAPConnection com.itko.lisa.stats.jmx.ITKOAgentConnection com.itko.lisa.stats.jmx.OracleASConnector
lisa.jmx.typeprops	com.itko.lisa.stats.jmx.JSE5Connection=LISA_JMX_JSE5 com.itko.lisa.stats.jmx.TomcatConnection=LISA_JMX_TOMCAT5 com.itko.lisa.stats.jmx.JBossConnection=LISA_JMX_JBOSS3240 com.itko.lisa.stats.jmx.JSR160RMIConnection=LISA_JMX_JSR160RMI com.itko.lisa.stats.jmx.Weblogic9Connector=LISA_JMX_WLS9 com.itko.lisa.stats.jmx.WeblogicConnector=LISA_JMX_WLS6781 com.itko.lisa.stats.jmx.OracleASConnector=LISA_JMX_OC4J com.itko.lisa.stats.jmx.WebsphereSOAPConnection=LISA_JMX_WASSOAP5X com.itko.lisa.stats.jmx.ITKOAgentConnection=LISA_JMX_ITKOAGENT

Vous n'avez généralement rien besoin d'ajouter à ces paramètres.

LISA_JMX_JSR160RMI

Valeur par défaut : LISA_HOME/lib/mx4j.lib

LISA_JMX_ITKOAGENT

Valeur par défaut : LISA_HOME/lib/mx4j.lib

LISA_JMX_JSE5

Si vous exécutez JSE 5, vous ne devez pas modifier cette propriété. Un client jbossall est fourni et devrez couvrir vos besoins, en supposant que la version est appropriée.

LISA_JMX_JBOSS3240

Valeur par défaut :
LISA_HOME/lib/mx4j.lib{{path.separator}}LISA_HOME/hotDeploy/jbossall-client.jar

LISA_JMX_TOMCAT5

Ce paramètre est spécifique à Tomcat.

Valeur par défaut :
LISA_HOME/lib/mx4j.lib{{path.separator}}LISA_HOME/hotDeploy/mx4j-tools.jar

LISA_JMX_WLS9

Valeur par défaut :
LISA_HOME/hotDeploy/weblogic.jar{{path.separator}}LISA_HOME/hotDeploy/wljmxclient.jar

LISA_JMX_WLS6781

Ce paramètre doit indiquer l'emplacement du fichier weblogic.jar. Aucun fichier wlclient.jar ne fonctionnera.

Valeur par défaut : LISA_HOME/hotDeploy/weblogic.jar

Oracle AS

LISA_JMX_OC4J

Valeur par défaut :
LISA_HOME/lib/oc4jclient.jar{{path.separator}}LISA_HOME/lib/adminclient.jar

IBM WebSphere

LISA_JMX_WASSOAP5X

Il est plus facile d'utiliser la variable CLASSPATH d'IBM/WAS avant de démarrer DevTest et de la laisser vide.

Valeur par défaut : LISA_HOME/lib/mx4j.lib

lisa.alert.email.emailAddr

Si vous utilisez des alertes de surveillance de performances, ce paramètre est l'adresse électronique de l'expéditeur de ces alertes.

Format : lisa@itko.com

lisa.alert.email.defHosts

Ce paramètre correspond au serveur de messagerie utilisé pour le routage des courriels (serveur SMTP).

Valeur par défaut : localhost

lisa.rundoc.builtins

Valeurs :

com.itko.lisa.files.1user1cycle.stg

Exécute le scénario de test une fois avec un utilisateur simulé.

com.itko.lisa.files.1user1cycle0think.stg

Exécute le scénario de test une fois avec un utilisateur simulé et aucun délai de réflexion.

com.itko.lisa.files.1user1min.stg

Exécute le scénario de test avec un utilisateur simulé pendant une minute et redémarre le test si nécessaire.

com.itko.lisa.files.1user5min.stg

Simulez le test pendant 5 minutes, puis redémarre si nécessaire.

com.itko.lisa.files.1usernonstop.stg

Exécute le test jusqu'à l'arrêt manuel.

com.itko.lisa.files.5user1min.stg

Exécute le test avec 5 utilisateurs virtuels pendant 1 minute.

lisa.auditdoc.builtins

Valeur par défaut : com.itko.lisa.files.DefaultAudit.aud

Propriétés de l'ITR/du gestionnaire de tests

lisa.tm.itr.max.delay.seconds

Par défaut : 5

Shells de commande externes

test.cmde.win.shell=cmd /c

test.cmde.unix.shell=sh -c

test.cmde.Windows.NT.(unknown).shell=cmd /c

Paramètres de test

lisa.props.blankOnMissing=true

Propriété à utiliser pour que DevTest remplace la clé par une chaîne vide si l'état de la clé n'est pas approprié.

lisa.test.custevents=&101="Custom Event 101"; &102="Custom Event 102"

Evénements personnalisés : le premier numéro d'événement permis est 101. Deux exemples sont donc présentés ici.

lisa.SimpleWebFilter.responseCodeRegEx=[45] d d

lisa.fsss.dateformat=MM/dd/yyyy hh:mm:ss a

Propriétés utilisées par StdSchedulerFactory pour créer une instance de planificateur Quartz

Configuration des propriétés du planificateur principal

org.quartz.jobStore.class

Valeur par défaut : org.quartz.simpl.RAMJobStore

org.quartz.threadPool.class

Valeur par défaut : org.quartz.simpl.SimpleThreadPool

org.quartz.threadPool.threadCount

Par défaut : 5

lisa.meta-refresh.max.delay

Par défaut : 5

Paramètres spécifiques à une plate-forme dans le gestionnaire de tests

lisa.tm.exec.unix

Valeur par défaut : xterm -e {0}

lisa.tm.exec.win

Valeur par défaut : cmd /c start {0}

lisa.tm.exec.osx

Valeur par défaut : open -a /Applications/Utilities/Terminal.app {0}

Propriétés utilisées pour la prise en charge des tests Swing

lisa.swingtest.client.logging.properties.file

Pour utiliser un fichier de propriétés de journalisation Log4J personnalisé dans SwingTestProgramStarter, annulez la mise en commentaire.

Valeur par défaut : C:/Lisa/swingtestclient-logging.properties

Paramètres de licence

laf.request

Valeur par défaut : laf/license.do

laf.default.url

Valeur par défaut : https://license.itko.com

laf.displaysetting

Valeur par défaut : true

Propriétés de rapports

lisa.reporting.defaultPageSize

Valeurs : A4 | letter

Propriétés JPA de rapports

rpt.eclipselink.ddl-generation

Valeur par défaut : create-tables

rpt.eclipselink.ddl-generation.output-mode

Valeur par défaut : database

rpt.eclipselink.validateschema

Valeur par défaut : false

perfmgr.rvwiz.whatrpt.autoExpire

Valeur par défaut : true

perfmgr.rvwiz.whatrpt.expireTimer

Pour rechercher des rapports expirés, définissez autoExpire sur true. Définissez la période d'expiration : un nombre entier suivi par m (mois), w (semaine), d (jour) ou h (heure). La période d'expiration par défaut est 30d (30 jours).

Valeur par défaut : 30d

rpt.hibernate.validateschema

Permet de valider le schéma de base de données de rapports. Par défaut, seul le registre valide le schéma.

Valeur par défaut : false

lisa.0.registry.local.autoshutdown

Valeur par défaut : true

lisa.8.registry.local.autoshutdown

Valeur par défaut : true

lisa.10.registry.local.autoshutdown

Le comportement par défaut est de ne pas arrêter automatiquement le registre local s'il a été démarré automatiquement. Des exceptions existent pour DevTest Workstation, VSE et VSE Workstation.

Valeur par défaut : true

lisa.4.registry.local.autoshutdown

Valeur par défaut : false

lisa.5.registry.local.autoshutdown

JUnit et TestRunner ne doivent jamais arrêter le registre automatiquement.

Valeur par défaut : false

Propriété utilisée pour le connecteur Eclipse

lisa.eclipse.connector.port

Valeur par défaut : 8546

Propriété utilisée pour les exemples de suites de tests

EXAMPLES_HOME

Valeur par défaut : LISA_HOME/examples

Propriétés de VSE

lisa.magic.string.min.length

Définit la longueur minimum de la valeur d'argument dans une demande de transaction VSE qui est requise pour considérer cet argument pour la construction d'une chaîne magique.

Valeur par défaut : 3

lisa.magic.string.word.boundary.type

Définit la relation des recherches de contenu de chaîne magique dans VSE avec les limites de mot.

Valeurs :

- **none** (aucune) : aucune limite de mot n'est prise en compte.
- **start** (début) : les candidats de chaîne magique doivent commencer par une limite de mot.
- **end** (fin) : les candidats de chaîne magique doivent se terminer par une limite de mot.
- **both** (les deux) : les candidats de chaîne magique doivent être identifiés comme des mots entiers, c'est-à-dire, qu'une limite doit se trouver au début et à la fin du mot.

Valeur par défaut : both

lisa.magic.string.exclusion

Spécifie si certaines chaînes doivent être exclues de la génération de chaînes magiques.

Valeurs : Yes, YES, yes, No, NO, no, true, True, TRUE, false, False, FALSE, __NULL

Valeur par défaut : Yes, YES, yes, No, NO, no, true, True, TRUE, false, False, FALSE, __NULL

lisa.magic.string.xml.tags

Spécifie si le texte placé entre des balises XML doit être converti en chaînes magiques.

Valeur par défaut : false

lisa.vse.server.dir.full.service.name=false

Spécifie si plusieurs instances de VSE s'exécutent sur différents ordinateurs à partir du même répertoire d'installation. Si c'est le cas, supprimez les marques de commentaire de cette propriété et définissez-la sur true.

Valeur par défaut : false

lisa.vse.response.xml.prettyprint

Spécifie si le code XML dans l'image de service doit être formaté si VSE enregistre une réponse XML. La valeur par défaut, false, indique que DevTest n'améliore l'impression, ne formate, ni n'ajoute de sauts de ligne au XML. Si le XML arrive sous forme d'une longue chaîne, DevTest l'affiche tel quel.

Valeur par défaut : false

lisa.vse.remember.execution.mode

Indique si VSE doit mémoriser les modes d'exécution que vous définissez dans le tableau de bord de VSE pour les services virtuels. Ajoutez des commentaires à cette propriété si vous ne voulez pas que VSE mémorise les modes d'exécution entre les arrêts.

Valeur par défaut : true

lisa.vse.match.event.buffer.size

Définit le nombre d'événements pour que chaque modèle de service virtuel définisse le nombre d'événements liés correspondants que le VSE doit mettre en tampon. La propriété définit le nombre d'événements pour chaque modèle de service virtuel. Par exemple, si deux modèles de service virtuel sont déployés avec une taille de tampon d'événement par défaut de 100, 200 événements sont mis en tampon. Ces événements sont utilisés comme source pour l'onglet Match (Correspondance) de la page d'inspection de modèle de service virtuel dans le tableau de bord de VSE.

Par défaut : 100

lisa.vse.request.event.set.buffer.size

Définit le nombre de demandes de VSE entrantes pour lesquelles un ensemble complet d'événements DevTest est mis en tampon. La propriété définit le nombre d'événements pour chaque modèle de service virtuel. Par exemple, si vous avez deux modèles de service virtuel déployés avec une taille de tampon de demande par défaut de 5, dix ensembles d'événements (groupés par demande entrante) seront mis en tampon. Ces ensembles d'événements sont utilisés comme source pour l'onglet Events (Evénements) de la page d'inspection de modèle de service virtuel dans le tableau de bord de VSE.

Pour des raisons de performances, VSE contient 50 événements à traiter. Cette limite entraîne parfois la suppression de certains événements de la file d'attente de traitement et empêche l'affichage des étapes dans la vue d'inspection. Pour éviter la troncation des événements, vous pouvez augmenter la taille de la file d'attente de traitement d'événements. Toutefois, cette action peut augmenter l'utilisation de la mémoire et affecter les performances.

Par défaut : 5

lisa.vse.si.text.editor.order

Définit l'ordre dans lequel les éditeurs de réponse de texte VSE enregistrés sont interrogés lorsque vous utilisez la détection automatique. Seule la partie droite du nom de classe est requis pour le rendre unique.

Valeurs : XMLTextEditor, JSONTextEditor

Valeur par défaut : l'éditeur de texte par défaut.

lisa.tm.def.min.millis

Définit le temps de réflexion minimum en millisecondes par défaut pour les nouvelles étapes standard.

Valeur par défaut : 500

lisa.tm.def.max.millis

Définit le temps de réflexion maximum en millisecondes par défaut pour les nouvelles étapes standard.

Valeur par défaut : 1000

lisa.vse.rest.max.optionalqueryparams

Spécifie le nombre maximum de paramètres de requête facultatifs à utiliser par méthode dans un fichier WADL ou RAML. Si le nombre de paramètres de requête facultatifs *lisa.vse.rest.max.optionalqueryparams* spécifiés est supérieur dans un fichier WADL ou RAML, seuls les premiers *lisa.vse.rest.max.optionalqueryparams* seront utilisés pour créer des transactions.

Valeur par défaut : 5

Propriétés de Date-Checker

Les utilitaires de date de VSE utilisent les propriétés suivantes pour déterminer les modèles de date valides pour les conversions de date. Chaque entrée suivante représente une expression régulière considérée comme valide par VSE au sein d'une date.

lisa.vse.datechecker.dayregex

((\[12\]\\d)\|(3\[01\])\|(0?\[1-9\]))

lisa.vse.datechecker.monthnumberregex

((1\[012\])\|(0 \\d)\|0\[1-9\]\|\[1-9\])

lisa.vse.datechecker.monthalpharegex

(\\bJAN\\b\|\\bFEB\\b\|\\bMAR\\b\|\\bAPR\\b\|\\bMAY\\b\|\\bJUN\\b\|\\bJUL\\b\|\\bAUG\\b\|\\bSEP\\b\|\\bOCT\\b\|\\bNOV\\b\|\\bDEC \\b)

lisa.vse.datechecker.yearlongregex

\\d\\d\\d\\d

lisa.vse.datechecker.yearshortregex

\\d\\d

lisa.vse.datechecker.timeregex

(\\s?((\[012\]? \\d)\|(2\[0123\]))\:((\[012345\] \\d)\|(60))\:((\[012345\] \\d)\|(60)))

lisa.vse.datechecker.time.hhmmssregex

((\[012\]? \\d)\|(2\[0123\]))\:((\[012345\] \\d)\|(60))\:((\[012345\] \\ d)\|(60))

lisa.vse.datechecker.time.millisregex

((\[012\]?\\d)\|(2\[0123\]))\:((\[012345\]\\d)\|(60))\:((\[012345\]\\d)\|(60))\\.((\\d \\d \\d)\|0)

lisa.vse.datechecker.time.millis.zoneregex

((\[012\]?\\d)\|(2\[0123\]))\:((\[012345\] \\d)\|(60))\:((\[012345\] \\ d)\|(60))\\.((\\d \\d \\d)\|0) \\s(\[A-Za-z\]\[A-Za-z\]\[A-Za-z\])

lisa.vse.datechecker.wstimestampregex

\\d\\d\\d\\d-((1\[012\])\|(0\\d)\|0\[1-9\]\|\[1-9\])-((\[12\]-\\-d)\|(3\[01\])\|(0?\[1-9\]))T((\[012\]?-\\-d)\|(2\[0123\]))\:((\[012345\]-\\-d)\|(60))\:((\[012345\]-\\-d)\|(60))-\\-.-\\-d-\\-d-\\-d\[-+\]\\d\\d\\d \\d

lisa.vse.datechecker.ddmmmyyyyregex

((\[12\]\\d)\|(3\[01\])\|(0?\[1-9\]))(JAN\|FEB\|MAR\|APR\|MAY\|JUN\|JUL\|AUG\|SEP\|OCT\|NOV\|DEC)\\d \\d \\d \\d

lisa.vse.datechecker.mmmddyyyyregex

(JAN\|FEB\|MAR\|APR\|MAY\|JUN\|JUL\|AUG\|SEP\|OCT\|NOV\|DEC)((\[12\]\\d)\|(3\[01\])\|(0?\[1-9\]))\\d \\d \\d \\d

lisa.vse.datechecker.yyyyddmmmregex

\\d\\d \\d \\d((\[12\]
\\d)\|(3\[01\])\|(0?\[1-9\]))(JAN\|FEB\|MAR\|APR\|MAY\|JUN\|JUL\|AUG\|SEP\|OCT\|NOV\|DEC)

lisa.vse.datechecker.yyyymmmddregex

\\d\\d\\d\\d(JAN\|FEB\|MAR\|APR\|MAY\|JUN\|JUL\|AUG\|SEP\|OCT\|NOV\|DEC)((\[12\] \\d)\|(3\[01\])\|(0?\[1-9\]))

lisa.vse.datechecker.ddmmmregex

((\[12\]\\d)\|(3\[01\])\|(0?\[1-9\]))(JAN\|FEB\|MAR\|APR\|MAY\|JUN\|JUL\|AUG\|SEP\|OCT\|NOV\|DEC)

lisa.vse.datechecker.mmmddregex

(JAN\|FEB\|MAR\|APR\|MAY\|JUN\|JUL\|AUG\|SEP\|OCT\|NOV\|DEC)((\[12\]\\d)\|(3\[01\])|(0?\[1-9\]))

lisa.vse.datechecker.ddmmmyyregex

((\[12\]\\d)\|(3\[01\])\|(0?\[1-9\]))(JAN\|FEB\|MAR\|APR\|MAY\|JUN\|JUL\|AUG\|SEP\|OCT\|NOV\|DEC)\\d\\d

Chaque entrée suivante représente un modèle valide d'une partie de la date dans VSE :

lisa.vse.datechecker.dayformat

Format : dd

lisa.vse.datechecker.monthnumberformat

Format : MM

lisa.vse.datechecker.monthalphaformat

Format : MMM

lisa.vse.datechecker.yearlongformat

Format : yyyy

lisa.vse.datechecker.yearshortformat

Format : yy

lisa.vse.datechecker.timeformat

Format : HH:mm:ss

lisa.vse.datechecker.time.hhmmssformat

Format : HH:mm:ss

lisa.vse.datechecker.time.millisformat

Format : HH:mm:ss.SSS

lisa.vse.datechecker.time.millis.zoneformat

Format : HH:mm:ss.SSS z

lisa.vse.datechecker.wstimestampformat

Format : yyyy-MM-dd'T'HH:mm:ss.SSSZ

Les modèles de date suivants ne peuvent pas être créés à l'aide d'une combinaison pour les modèles précédents impliquant au moins un jour, un mois et une année :

lisa.vse.datechecker.mmmddyyyy.separatorformat

Format : MMM*dd*yyyy

lisa.vse.datechecker.mmddyyyy.separatorformat

Format : MM*dd*yyyy

lisa.vse.datechecker.ddmmmyyyy.separatorformat

Format : dd*MMM*yyyy

lisa.vse.datechecker.ddmmyyyy.separatorformat

Format : dd*MM*yyyy

lisa.vse.datechecker.yyyymmmdd.separatorformat

Format : yyyy*MMM*dd

lisa.vse.datechecker.yyyymmdd.separatorformat

Format : yyyy*MM*dd

lisa.vse.datechecker.ddmmmyyyyformat

Format : ddMMMyyyy

lisa.vse.datechecker.mmmddyyyyformat

MMMddyyyy

lisa.vse.datechecker.yyyyddmmmformat

Format : yyyyddMMM

lisa.vse.datechecker.yyyymmmddformat

Format : yyyyMMMdd

lisa.vse.datechecker.ddmmmyyformat

Format : ddMMMyy

lisa.vse.datechecker.ddmmmformat

Format : ddMMM

lisa.vse.datechecker.mmmddformat

Format : MMMdd

lisa.vse.datechecker.separators

Définit les caractères de séparation valides à utiliser dans les formats de date.

Valeurs : -/.

Exemple : lisa.vse.datechecker.separators=/ définit la barre oblique / comme un séparateur dans la date 15/10/11.

lisa.vse.datechecker.top.priorityorder=lisa.vse.datechecker.wstimestampformat

Définit l'ordre dans lequel la correspondance au modèle de date doit être établie. Les caractères de séparation définis par **lisa.vse.datechecker.separators** remplacent les astérisques.

Exemple : MM*dd*yy génère quatre modèles de date : MM-dd-yyyy, MM dd yyyy, MM/dd/yyyy, MM.dd.yyyy.

lisa.vse.datechecker.date.priorityorder

lisa.vse.datechecker.mmmddyyyy.separatorformat&\
lisa.vse.datechecker.mmddyyyy.separatorformat&\
lisa.vse.datechecker.ddmmmyyyy.separatorformat&\
lisa.vse.datechecker.ddmmyyyy.separatorformat&\
lisa.vse.datechecker.yyyymmmdd.separatorformat&\
lisa.vse.datechecker.yyyymmdd.separatorformat&\
lisa.vse.datechecker.mmmddyyyyformat&\
lisa.vse.datechecker.ddmmmyyyyformat&\
lisa.vse.datechecker.yyyyddmmmformat&\
lisa.vse.datechecker.yyyymmmddformat&\
lisa.vse.datechecker.ddmmmyyformat

lisa.vse.datechecker.time.priorityorder

lisa.vse.datechecker.time.millis.zoneformat&\
lisa.vse.datechecker.time.millisformat&\
lisa.vse.datechecker.time.tenthsformat&\
lisa.vse.datechecker.time.hhmmssformat

lisa.vse.datechecker.bottom.priorityorder

lisa.vse.datechecker.mmmddformat&\
lisa.vse.datechecker.ddmmmformat

lisa.vse.datechecker.months

Valeurs : JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC

lisa.vse.datechecker.coarse.year.regex

Format :
(?ism).*((\\d[./-](19|20)[0-9]{2})|((JAN|FEB|MAR|APR|MAY|JUN|JUL|AUG|SEP|OCT|NOV|DEC)[. /-](19|20)[0-9]{2})|((19|20)[0-9]{2}[. /-][0-9]{2})).*

Définit un test de niveau rudimentaire pour déterminer si la charge utile contient un élément similaire à une année. Pour éviter un nombre excessif de correspondances, il recherche d'autres informations permettant une meilleure identification, telles que des informations relatives aux mois ou des séparateurs de date communs (exemple: -, / ou .).

Lorsque l'année ne peut pas être distinguée d'autres nombres à 4 chiffres, ce modèle peut ne pas renvoyer de correspondance. Dans ce cas, les dates magiques seront introuvables. Essayez l'une des solutions suivantes :

- Ajustez cette expression régulière.
- Essayez le format le moins restrictif suivant.

lisa.vse.datechecker.coarse.year.regex

Format :
(?ism).*((\\d[./-]?(19|20)[0-9]{2})|((JAN|FEB|MAR|APR|MAY|JUN|JUL|AUG|SEP|OCT|NOV|DEC)[. /-]?(19|20)[0-9]{2})|((19|20)[0-9]{2}[. /-]?[0-9]{2})).*

Définit un test de niveau rudimentaire moins restrictif que le test précédent pour déterminer si la charge utile contient un élément similaire à une année. Si les dates ne sont pas reconnues, cette expression régulière peut vous permettre d'obtenir des résultats. Pour éviter un nombre excessif de correspondances, il recherche d'autres informations permettant une meilleure identification, telles que des informations relatives aux mois ou des séparateurs de date communs (exemple: -, / ou .).

Toutefois, cette expression régulière peut affecter les performances. Si cette expression régulière détecte trop de dates, les modèles ci-dessus sont exécutés dans des cas non requis. Exécuter un grand nombre d'expressions régulières sur des charges utiles volumineuses peut également supposer d'importants délais.

Propriétés JMS personnalisées à ignorer pour la messagerie JMS de VSE

vse.jms.ignore.proplist

Certaines plates-formes JMS incluent des propriétés personnalisées supplémentaires dans les messages JMS qu'elles livrent. Ces propriétés peuvent affecter la création de VSE. **vse.jms.ignore.proplist** contient une liste de propriétés à ignorer lorsque vous enregistrez un service JMS avec l'enregistreur de messagerie de VSE. Le format consiste en une liste d'expressions régulières séparées par des virgules. Par défaut, les propriétés d'extension JMS standard (JMSX) et les propriétés comptables JBoss personnalisées (JMS_.) sont exclues.

Valeur par défaut : JMSX.*, JMS_.*

Taille des lots de conversation de l'enregistreur de VSE

lisa.vse.recorder.conversation.batch.size

Cette propriété définit le nombre de conversations traitées avant de les valider dans la base de données. Pour un petit nombre de longues conversations, définissez un petit nombre pour cette propriété. Pour un grand nombre de conversations plus petites, définissez un grand nombre pour cette propriété. Une taille de lot <= 0 est identique à une taille de lot de 1.

Valeur par défaut : 10

Paramètres de base de données d'image de service VSE

eclipselink.ddl-generation

Valeur par défaut : create-tables

eclipselink.ddl-generation.output-mode=database

Définit la base de données à utiliser comme référentiel pour des images de service. VSE utilise le fournisseur open source EclipseLink JPA. Par défaut, la base de données de rapports de l'application est utilisée comme référentiel pour les images de service.

Valeur par défaut : Base de données de rapports LISA

Validation du schéma de VSE

lisa.eclipselink.query.warn.threshold

Si EclipseLink est configuré par défaut pour effectuer des calculs de temps de requêtes, cette valeur correspond au temps maximum en millisecondes qu'une requête sur la base de données peut prendre avant que l'enregistreur **com.itko.lisa.vse.stateful.model.SqlTimer** déclenche un message de niveau WARN. Si vous définissez le seuil de cet enregistreur sur DEBUG (Déboguer), les calculs de temps de tous les problèmes EclipseLink liés aux instructions SELECT sont affichés. Cette journalisation constitue la première étape du débogage des problèmes de performance de VSE. Si la base de données est locale et tous les index ont été créés, aucune valeur ne devrait dépasser les 20 ms. Toutefois, une base de données du réseau qui n'est pas sous votre contrôle et dont certains index peuvent être manquants, peut facilement prendre plus de 100 ms pour renvoyer des résultats. Vous pouvez donc être confronté à des lenteurs, alors que VSE fonctionne très rapidement s'il est correctement déployé. La plupart des images sont suffisamment petites ou leur plage de travail est suffisamment réduite pour être contenue dans le cache d'entité du VSE. Par conséquent, le nombre de problèmes liés aux instructions SELECT renvoyés par que VSE devrait diminuer et finalement disparaître, tant que le segment de mémoire du VSE est un assez grand. Le cache contient des références soft par défaut.

Valeur par défaut : 100

Propriétés VSE de détection de contenu délimité

lisa.vse.delimited.delimiter.check.maxchars

Définit le nombre maximum de caractères à examiner pour déterminer le type de contenu.

Valeur par défaut : 1000

lisa.vse.delimited.delimiter.check.maxpercent

Définit le pourcentage maximum de caractères à examiner pour déterminer le type de contenu. Cette propriété est utilisée si sa valeur est supérieure à celle de **lisa.vse.delimited.delimiter.check.maxchars**.

Valeur par défaut : 2

lisa.vse.delimited.delimiter.list

Définit les caractères que le détecteur de contenu délimité recherche en tant que délimiteurs.

Valeur par défaut : :,\\,,;,|,\u00A6,\t,\n

lisa.vse.delimited.delimiter.minimum.threshold

Définit le nombre minimum de délimiteurs comptés par le détecteur de contenu délimité. La charge utile doit au moins contenir ce nombre de délimiteurs.

Valeur par défaut : 3

lisa.vse.delimited.namevalue.separator.list

Définit les séparateurs nom-valeur utilisés pour détecter la charge utile de nom-valeur délimitée. Ces valeurs ne doivent pas figurées dans la liste **lisa.vse.delimited.delimiter.list**.

Propriétés de protocole de données REST

lisa.protocol.rest.editor.observedtraffic.max

Définit le nombre maximum d'éléments de trafic correspondant qui sont remplis dans le volet inférieur de l'assistant de règles d'URI. Si l'assistant tarde longtemps à remplir le volet inférieur avec le trafic correspondant, spécifiez une valeur inférieure.

Valeur par défaut : 100

lisa.protocol.rest.editor.unmatchedtraffic.max

Définit le nombre maximum d'éléments de trafic non correspondant qui s'affichent dans la boîte de dialogue de trafic non correspondant. Si l'assistant tarde longtemps à remplir les champs de l'assistant, spécifiez une valeur inférieure.

Par défaut : 100

lisa.protocol.rest.idPattern

Définit pour le protocole de données REST une expression régulière qui détecte les parties d'une demande HTTP qu'il peut considérer comme identificateur. Le protocole de données convertit ces identificateurs en paramètres dynamiques. Lorsque vous avez des paramètres dynamiques qui suivent un format spécifique, vous pouvez utiliser cette propriété pour les détecter.

Valeur par défaut : [a-zA-Z]+[0-9]{5,}[a-zA-Z]*

Exemple : La valeur **id12345** peut représenter un élément prédéfini et la règle suivante contient {URLPARAM0}.

lisa.protocol.rest.maxChanges

Définit le nombre maximum de modifications permises par VSE pour un jeton avant que la variabilité soit considérée suffisamment significative pour générer une règle.

Valeur par défaut : 1

lisa.protocol.rest.parameterBaseName

Définit le préfixe que le protocole de données REST utilise pour les paramètres dans des règles.

Par défaut : URLPARAM

lisa.protocol.rest.startPosition

Définit la position dans l'URL à partir de laquelle le protocole de données REST commence à rechercher des jetons variables.

Valeur par défaut : 3

Exemple : dans l'URI **GET /a/b/c/d**, **GET** est à la position 0, **a** à la position 1 et **b** à la position 2.

vse.log.trace.truncate.response.at=2048

Limite la taille des réponses que DevTest écrit dans le vse.log. Cette propriété s'applique uniquement lorsque la journalisation de niveau de suivi est activée : particulièrement, lorsque **com.itko.lisa.vse.http.Transaction**=TRACE (suivi) ou lorsque **com.itko.lisa.vse.stateful.protocol.http.Coordinator**=TRACE (suivi).

Valeurs :

- **0** : la réponse complète est journalisée.
- **>0** : La réponse journalisée est tronquée au niveau du caractère spécifié.

Valeur par défaut : 2048

Propriétés de port réseau

Les propriétés de port réseau sont définies sous la forme **lisa.net.APPID.port**.

Si vous devez modifier ces valeurs par défaut, remplacez-les dans le fichier **site.properties.** Les clients déterminent le port à utiliser pour la connexion au serveur en fonction de ces valeurs.

lisa.net.0.port=2008

DevTest Workstation

lisa.net.2.port=2011

Coordinateur

lisa.net.3.port=2010

Registre

lisa.net.4.port=2012

JUnit exec

lisa.net.5.port=2005

Exécuteur de tests (cmdline)

lisa.net.6.port=2007

Autres

lisa.net.8.port=2013

VSE

lisa.net.9.port=2004

VSEManager

lisa.net.11.port=2006

ServiceManager

lisa.net.1.port=2014

Simulateur (commencer à partir de ce port permet d'en définir plusieurs sur le même ordinateur).

Propriétés CA Continuous Application Insight

lisa.pathfinder.on

Valeur par défaut : true

lisa.pathfinder.broker.host

Valeur par défaut : 0.0.0.0

lisa.pathfinder.broker.port

Valeur par défaut : 2009

lisa.webserver.port

Serveur Web intégré.

Valeur par défaut : 1505

lisa.webserver.https.enabled

Valeur par défaut : false

devtest.port

Valeur par défaut : 1507

lisa.enable.workstation.pathfinder.browser.ui

Valeur par défaut : true

lisa.portal.enable.pathfinder.browser.ui

Valeur par défaut : true

lisa.webserver.host

Hôte du serveur Web intégré. *0.0.0.0* permet de lier toutes les adresses locales.

Valeur par défaut : 0.0.0.0

lisa.webserver.acl.authmodule.baseurl

Valeur par défaut : /acl/

Les propriétés suivantes définissent les valeurs pour tous les lanceurs d'application de portail de DevTest Workstation. Actuellement, le nom d'hôte du Test Runner (Exécuteur de tests) est utilisé pour la valeur de l'hôte. La valeur est donc obtenue de façon dynamique. L'URL est lancée de façon externe dans le navigateur de votre système.

lisa.portal.url.prefix

Valeur par défaut : http://* (http://*)

lisa.portal.root.base.url

Valeur par défaut : /index.html

lisa.portal.cvsdashboard.base.url

Valeur par défaut : /index.html?lisaPortal=cvsdashboard

lisa.portal.pathfinder.console.base.url

Valeur par défaut : /index.html?lisaPortal=pathfinder

lisa.portal.server.console.base.url

Valeur par défaut : /index.html?lisaPortal=serverconsole

lisa.portal.model.execution.url

Valeur par défaut : /index.html?lisaPortal=serverconsole

lisa.portal.reporting.context

Valeur par défaut : reporting

lisa.portal.reporting.console.base.url

Valeur par défaut : /index.html?lisaPortal=reporting

lisa.portal.defect.capture.url

Valeur par défaut : /pathfinder/lisa_pathfinder_agent/defectcapture

lisa.portal.save.defect.data.url

Valeur par défaut : /pathfinder/lisa_pathfinder_agent/saveDefectData

lisa.portal.invoke.base.url

Valeur par défaut : /lisa-invoke

lisa.portal.invoke.report.url

Valeur par défaut : /reports

lisa.portal.invoke.server.report.directory

Valeur par défaut : lisa.tmpdirlisa.portal.invoke.report.url

devtest.portal.base.url.path

Valeur par défaut : /devtest/#/main

devtest.portal.homepage.url.path

Valeur par défaut : {{devtest.portal.base.url.path}}/dashboard

lisa.portal.pathfinder.explorer.base.url

Valeur par défaut : {{devtest.portal.base.url.path}}/createartifacts

lisa.portal.pathfinder.management.base.url

Valeur par défaut : {{devtest.portal.base.url.path}}/pathManageAgents

lisa.portal.invoke.test.root

Valeur par défaut : LISA_HOME

Propriétés de vue graphique des rapports

rpt.lisa.graph.view.threshhold

Par défaut : 100

rpt.lisa.graph.view.infomessage

Résultats supérieurs au seuil maximum. Permet de modifier un filtre pour extraire les détails.

rpt.lisa.graph.scatter.view.threshhold

Valeur par défaut : 10000

rpt.lisa.graph.scatter.view.infomessage

Résultats supérieurs au seuil maximum. Pour extraire les détails, modifiez un filtre.

Prise en charge de la génération de rapports asynchrone

lisa.reporting.useAsync

Si vous exécutez des tests de charge et qu'un goulot d'étranglement dans les scénarios de test écrit les événements dans la base de données de rapports, supprimez toutes les données sauf les mesures du générateur de rapports. Si le moteur de génération de rapports continue à être un goulot d'étranglement, activez cette propriété. La propriété utilise JMS pour envoyer l'événement de rapport et les threads d'arrière-plan dans les simulateurs et le coordinateur écrit les événements dans la base de données de façon asynchrone. Votre test de charge se termine donc avant que tous les événements soient écrits dans la base de données et le rapport ne s'affiche pas tout de suite (le délai dépend de vos scénarios de test et du nombre d'événements générés). La file d'attente de simulateur tarde généralement plus longtemps à se vider. Vous pouvez consulter le pourcentage effectué dans un message au niveau INFO du journal de simulateur.

Cette fonctionnalité est réservée à une utilisation avancée pour le moment et désactivée par défaut. N'hésitez pas à nous laisser vos commentaires sur le site http://ca.com/support.

Valeur par défaut : false

lisa.reporting.step.max.propsused.buffersize

Par défaut : 100

lisa.reporting.step.max.propsset.buffersize

Ces propriétés contrôlent la collecte de statistiques pour une étape de test, pendant l'exécution d'un scénario de test. Les valeurs spécifient le nombre maximum d'occurrences enregistrées pour les propriétés utilisées et les propriétés définies. Il n'est pas nécessaire de modifier les valeurs par défaut.

Par défaut : 100

lisa.threadDump.generate

Valeur par défaut : true

lisa.threadDump.interval

Valeur par défaut : 30

lisa.threadDump.loggerName

Permet d'activer des fils de discussion réguliers. Il est recommandé de laisser cette propriété telle quelle. Elle vérifie efficacement si le threadDumpLogger est défini sur INFO ou un niveau inférieur, et n'effectue aucune opération s'il est défini sur WARN ou un niveau supérieur. Les propriétés sont lues une seule fois au démarrage et le fichier **logging.properties** est vérifié toutes les 10 secondes à la recherche de modifications. Il vous suffit de conserver la configuration telle quelle et de définir le niveau de journalisation du threadDumpLogger sur INFO, puis de patienter 30 secondes maximum pour obtenir les fils de discussion périodiques d'un DevTest Server en cours d'exécution. Ces journaux sont utiles pour le débogage des problèmes de performance. Pour plus d'informations, consultez les commentaires dans le fichier **logging.properties**.

Valeur par défaut : threadDumpLogger

Ces propriétés sont utilisées pour contrôler la collecte de mesures dans les serveurs de VSE. Vous pouvez afficher les mesures qui sont collectées dans le tableau de bord Web de VSE.

lisa.vse.metrics.collect

Propriété principale permettant d'activer la collecte de mesures globale (true) ou de la désactiver (false).

Valeur par défaut : true

lisa.vse.metrics.txn.counts.level

Cette propriété contrôle le niveau auquel le nombre de transactions est enregistré. Lorsque les différents nombres de transactions nommés sont ajoutés, vous obtenez les transactions par périodes.

Valeurs : none (aucun ou false), service, request (demande)

- **Service :** nom des nombres de transactions de noms de service.
- **Request :** nombres de transaction de nom de service et de noms d'opération de demande.

Valeur par défaut : service

lisa.vse.metrics.sample.interval

Cette propriété contrôle la fréquence d'échantillonnage du taux de transactions et des temps de réponse.

Valeur par défaut : 5m

lisa.vse.metrics.delete.cycle

Valeur par défaut : 1h

lisa.vse.metrics.delete.age

Ces propriétés contrôlent la fréquence à laquelle les anciennes données de mesure sont analysées et supprimées, et la définition de l'ancienneté.

Valeur par défaut : 30d

lisadb.internal.enabled

Indique si l'instance de base de données Derby interne doit être démarrée dans le registre.

Valeur par défaut : true

lisadb.internal.host

Interface réseau utilisée par la base de données Derby interne. La valeur par défaut, 0.0.0.0, indique que toutes les interfaces sont utilisées.

Valeur par défaut : 0.0.0.0

lisadb.internal.port

Numéro de port sur lequel la base de données Derby interne effectue ses écoutes.

Valeur par défaut : 1528

lisa.acl.audit.logs.delete.frequency

Valeur par défaut : 1d

lisa.acl.audit.logs.delete.age

Ces propriétés contrôlent la fréquence à laquelle les anciennes données de journal d'audit de liste de contrôle d'accès sont analysées et supprimées, et la définition de l'ancienneté.

Valeur par défaut : 30d

Propriétés de base de données

Les composants suivants utilisent une base de données : le moteur de génération de rapports, l'intermédiaire d'agent, VSE et la liste de contrôle d'accès. En général, vous les configurez de manière à ce qu'ils pointent tous vers le même pool de connexions, mais vous pouvez définir un pool distinct pour chaque composant ou combiner les deux. Pour définir un nouveau pool, configurez des propriétés comme **lisa.db.pool.myPool.url**. L'implémentation de pool sous-jacente est le pool c3p0 open source, qui permet de transférer les différentes propriétés. Pour en savoir plus sur les paramètres disponibles, consultez la page http://www.mchange.com/projects/c3p0/index.html#configuration_properties.

lisadb.reporting.poolName

Valeur par défaut : common

lisadb.vse.poolName

Valeur par défaut : common

lisadb.acl.poolName

Valeur par défaut : common

lisadb.broker.poolName

Valeur par défaut : common

lisadb.pool.common.driverClass

Valeur par défaut : org.apache.derby.jdbc.ClientDriver

lisadb.pool.common.url

Valeur par défaut : jdbc:derby://localhost:1528/database/lisa.db;create=true

lisadb.pool.common.user

Valeur par défaut : rpt

lisadb.pool.common.password_enc

Pour définir le mot de passe, supprimez _enc à la fin du nom de propriété et ajoutez votre *mot de passe en texte brut* après le signe égal. Le mot de passe est automatiquement codé au démarrage.

Valeur par défaut : 76f271db3661fd50082e68d4b953fbee

Les propriétés de pool suivantes maintiennent le nombre de connexions au minimum lorsque DevTest est inactif.

lisadb.pool.common.minPoolSize

Valeur par défaut : 0

lisadb.pool.common.minPoolSize

Valeur par défaut : 0

lisadb.pool.common.maxPoolSize

Valeur par défaut : 10

lisadb.pool.common.acquireIncrement

Valeur par défaut : 1

lisadb.pool.common.maxIdleTime

Valeur par défaut : 45

lisadb.pool.common.idleConnectionTestPeriod

Valeur par défaut : 5

Autres paramètres de base de données communs : copier et coller l'utilisateur approprié, le mot de passe et les paramètres de taille de pool à partir du modèle commun.

lisadb.pool.POOLNAME.driverClass

Valeur par défaut : oracle.jdbc.OracleDriver

lisadb.pool.POOLNAME.url

Valeur par défaut : jdbc:oracle:thin:@HOST:1521:SID

lisadb.pool.POOLNAME.driverClass

Valeur par défaut : com.ibm.db2.jcc.DB2Driver

lisadb.pool.POOLNAME.url

Valeur par défaut : jdbc:db2://HOST:50000/DBNAME

lisadb.pool.POOLNAME.driverClass

Valeur par défaut : com.microsoft.sqlserver.jdbc.SQLServerDriver

lisadb.pool.POOLNAME.url

Valeur par défaut : jdbc:sqlserver://durry;databaseName=LISA

lisadb.pool.POOLNAME.driverClass

Valeur par défaut : com.mysql.jdbc.Driver

lisadb.pool.POOLNAME.url

Valeur par défaut : jdbc:mysql://HOST:3306/DBNAME

lisa.jdbc.asset.pool.size

Lorsque vous définissez un actif de connexion JDBC, la taille de pool de connexions peut être configurée par cette propriété.

Valeur par défaut : 5

Propriétés mainframe

lisa.mainframe.bridge.enabled

Indique si la passerelle mainframe des systèmes de contrôle des informations client doit être activée.

Valeur par défaut : false

lisa.mainframe.bridge.mode

Indique si la passerelle mainframe s'exécute en mode Client ou en mode Serveur.

Valeur par défaut : server

lisa.mainframe.bridge.port

Lorsque la passerelle mainframe s'exécute en mode Serveur, ce port est le port connu sur lequel la passerelle mainframe effectue ses écoutes.

Valeur par défaut : 61617

lisa.mainframe.bridge.server.host

Valeur par défaut : 127.0.0.1

lisa.mainframe.bridge.server.port

Lorsque la passerelle mainframe s'exécute en mode Client, ces valeurs sont l'adresse IP et le port connu de l'agent de partition logique.

Valeur par défaut : 3997

lisa.mainframe.bridge.connid

ID unique à deux caractères de chaque client.

Par défaut : AA

Pour plus d'informations, consultez la section Passerelle mainframe de la rubrique *Agents.*

Propriétés WebSphere MQ

lisa.mq.ccsid.default=819

Vous pouvez utiliser cette propriété pour remplacer le CCSID (Coded Character Set Identifier) par défaut pour tous les messages IBM WebSphere MQ envoyés à partir de DevTest. Vous pouvez également remplacer cette valeur au cas par cas dans les propriétés du message figurant dans la section Publisher info (Informations sur l'éditeur) de l'étape **IBM WebSphere MQ**. Pour plus d'informations, reportez-vous à la rubrique *Utilisation de CA Application Test*. Pour obtenir une liste complète des CCSID, consultez le site http://www-01.ibm.com/software/globalization/ccsid/ccsid_registered.html. La valeur par défaut pour l'environnement linguistique des Etats-Unis est 819 (ASCII).

Options de localisation

lisa.locale.languages=en

Vous pouvez utiliser cette propriété pour indiquer les langues à prendre en charge dans l'interface utilisateur. Les valeurs valides sont **en** et **ja**, pour l'anglais et le japonais respectivement.

Si vous spécifiez **en** et **ja**, vous pouvez interchanger la langue de l'interface utilisateur en anglais ou en japonais à l'aide de l'option System (Système), Language (Langue) du menu Main (Principal).

lisa.supported.html.request.encodings=

Les valeurs valides sont ISO-8859-1, UTF-8, Shift_JIS, EUC-JP, Windows-31J.

Propriétés d'intégration de Selenium

selenium.enable.waitfor=true

Spécifie si une étape waitForElementPresent implicite doit être ajoutée pour chaque étape du scénario de test qui doit localiser un élément Web.

Valeurs :

- **True** : ajoute l'étape implicite exécutée avant l'étape réelle de sorte que l'élément existe et que la page soit chargée.
- **False :** l'étape implicite n'est pas ajoutée. Si vous définissez cette propriété sur false, des scénarios peuvent se produire dans lesquels le script est exécuté dans le générateur Selenium, mais échoue dans DevTest en raison de retards dans le chargement la de page. Si vous avez déjà défini des étapes waitForElementPresent dans Selenium Builder, vous pouvez définir cette propriété sur false.

selenium.browser.type

Définit le type de navigateur sur lequel exécuter un scénario de test d'intégration de Selenium. Utilisez cette propriété dans un fichier de configuration de projet.

Valeurs :

- Chrome
- Internet Explorer
- Firefox

Remarque : Si cette propriété n'est pas spécifiée, les tests sont exécutés dans Firefox par défaut.

selenium.ie.driver.path

Définit le chemin complet du pilote Selenium sur un ordinateur local utilisé pour exécuter un test d'intégration de Selenium dans un navigateur Microsoft Internet Explorer. Utilisez cette propriété dans un fichier de configuration de projet.

Exemple :

```
C:\lisa-se\IEDriverServer.exe
```

selenium.chrome.driver.path

Définit le chemin complet du pilote Selenium sur un ordinateur local utilisé pour exécuter un test d'intégration de Selenium dans un navigateur Chrome. Utilisez cette propriété dans un fichier de configuration de projet.

Exemple :

```
C:\lisa-se\ChromeDriverServer.exe
```

selenium.remote.url

Définit l'URL pour un concentrateur de serveur Selenium distant pour exécuter des scénarios de test d'intégration de Selenium sur un navigateur distant. Utilisez cette propriété dans un fichier de configuration de projet.

Exemple :

```
http://nom_votre_hôte_distant:4444/wd/hub.
```

Remarque : Vous pouvez définir **selenium.chrome.driver.path** et **selenium.ie.driver.path** dans le même fichier de configuration du projet. Un fichier de configuration qui contient **selenium.chrome.driver.path** ou **selenium.ie.driver.path** ne peut pas contenir **selenium.remote.url**.

selenium.WebDriver.DesiredCapabilities.filePath

Spécifie l'emplacement du fichier de paramètres utilisé pour définir des options avancées dans le pilote Web Selenium.

Valeur par défaut : {{LISA_PROJ_ROOT}}/Data/selenium-capabilities.conf

Propriétés de VSEasy

Ces propriétés contrôlent l'affectation dynamique de ports lorsque vous utilisez la configuration automatique avec le protocole HTTP dans VSEasy.

Si le port minimum est inférieur à 1024, la valeur sera définie sur 1024. Si le port minimum est supérieur à 65535, la valeur sera définie sur 65535. Si le port maximum est inférieur ou égal au port minimum, les deux valeurs seront définies sur les valeurs par défaut.

lisa.vseasy.http.min.dynamic.port

Valeur par défaut : 8000

lisa.vseasy.http.max.dynamic.port

Valeur par défaut : 65535

lisa.vseasy.default.group.tag

Spécifie le nom du groupe par défaut dans la console VSEasy.

Par défaut : VSEasy

Annexe B: Fichiers de propriété personnalisés

Vous pouvez utiliser deux autres fichiers de propriété pour vos propriétés personnalisées :

- local.properties
- site.properties

Vous pouvez stocker des propriétés de site sur le serveur de test, qui envoie automatiquement le fichier **site.properties** à toutes les stations de travail qui s'y connectent.

Les fichiers de propriétés sont évalués dans l'ordre de priorité suivant : la ligne de commande et les fichiers .vmoptions sont toujours prioritaires par rapport aux fichiers de propriétés. Puis, **local.properties** est prioritaire par rapport à **site.properties**, qui est prioritaire par rapport à **lisa.properties**.

Remarque : Lorsque DevTest Workstation est démarré, vous êtes invité à vous connecter à un registre. Une fois que le registre est connecté, un fichier site.properties est envoyé à la station de travail. Si vous remplacez le registre Registry1 par Registry2, le fichier **site.properties** de Registry2 est envoyé à la station de travail.

Pour utiliser un fichier de propriété personnalisé :

1. Accédez au répertoire LISA_HOME.
2. Copiez le fichier **_site.properties** ou **_local.properties** et collez-le dans le même répertoire.
3. Changez le nom du fichier que vous avez collé dans **site.properties** ou **local.properties** (sans le trait de soulignement).

Ce chapitre traite des sujets suivants :

Fichier de propriétés local

lisaAutoConnect

Pour charger des propriétés de site automatiquement à partir d'un registre DevTest, supprimez les commentaires de cette propriété et définissez l'URL du registre DevTest approprié. La propriété **hostname/lisa.TestRegistry** fonctionne généralement. Les propriétés de ce fichier remplacent toutes les propriétés définies au niveau du site.

Pour charger des propriétés de site automatiquement à partir d'un registre DevTest, supprimez les commentaires de cette propriété et définissez l'URL du registre DevTest approprié. La propriété **hostname/lisa.TestRegistry** fonctionne généralement. Les propriétés de ce fichier remplacent toutes les propriétés définies au niveau du site pour les composants sans interface utilisateur.

DevTest Workstation n'utilise pas cette propriété. Vous pouvez sélectionner le registre auquel vous voulez vous connecter dans la boîte de dialogue Select LISA Registry (Sélectionner un registre LISA) ou en cliquant sur Change Registry (Modifier le registre) dans la station de travail. Si vous décochez la case d'invite au démarrage dans la boîte de dialogue Select Registry (Sélectionner un registre), DevTest Workstation se connecte au dernier registre utilisé.

Format : tcp://hôte-quelconque/Registre

Propriétés de licence

laf.server.url=https://license.itko.com

laf.domain=iTKO/LISA/YOURCO

laf.username=YOURUSERNAME

laf.password=YOURPASSWORD

Propriétés de VSE

lisa.vse.deploy.dir

Permet de remplacer le répertoire dans lequel les fichiers d'exécution sont gérés. Définissez cette propriété sur un chemin d'accès au répertoire absolu. Si vous ne spécifiez pas un lecteur au début du chemin d'accès, il est considéré comme relatif au répertoire LISA_HOME. Si vous spécifiez un répertoire à plusieurs niveaux, spécifiez le chemin d'accès sous la forme : C:\\Temp\\myVSE.

Remarque : Dans les fichiers de propriétés, vous devez doubler les barres obliques inversées pour qu'elles soient reconnues.

lisa.vse.si.editor.prefer.xml

Les valeurs valides se trouvent dans la liste des noms de classe d'éditeur de texte d'images de service VSE qui figure dans **typemap.properties**, affectés au nom **vseTextBodyEditors**. Si un éditeur de texte personnalisé est écrit et mis à disposition via des méthodes de kit de développement logiciel standard, ce nom de classe peut également être une valeur pour la propriété.

Les propriétés suivantes permettent à VSE d'utiliser une E/S de socket ancienne, au lieu de la nouvelle E/S (NIO). Tenez compte des restrictions suivantes lorsque vous utilisez l'ancienne prise en charge de socket :

- La capacité de VSE à gérer automatiquement un proxy avec une connexion SSL n'est pas prise en charge. La solution est de créer un module de service virtuel en mode Système dynamique qui pointe vers le service virtuel réel, avec l'option SSL activée.
- La capacité de VSE à gérer le trafic de texte brut, même lorsque l'étape d'écoute a été configurée pour utiliser une connexion SSL, n'est pas prise en charge. La solution est de fournir deux services virtuels, un avec une connexion SSL configurée et un autre sans.
- L'utilisation de l'ancienne prise en charge de socket ne permet pas une modularité aussi efficace que la prise en charge de socket par défaut utilisée par VSE.

lisa.vse.tcp.uses.nio

Lorsque cette propriété est définie sur false, les sockets simples et SSL utilisent l'ancienne E/S.

Valeur par défaut : true

lisa.vse.plain.tcp.uses.nio

Cette propriété contrôle uniquement des sockets simples.

Valeur par défaut : sa.vse.tcp.uses.nio

lisa.vse.ssl.tcp.uses.nio

Cette propriété contrôle uniquement des sockets SSL.

Valeur par défaut : isa.vse.tcp.uses.nio

lisa.vse.execution.mode

- EFFICIENT : ce mode permet d'utiliser le chemin le plus efficace via un modèle de service virtuel.
- TRACK : ce mode permet d'enregistrer l'activité de VSE au niveau du service virtuel.
- LIVE : ce mode permet de router les demandes reçues par un modèle de service virtuel vers un système dynamique (de manière semblable à un mode d'authentification directe).
- VALIDATION : ce mode utilise VSE et le système dynamique pour déterminer une réponse. Ils sont enregistrés en tant qu'informations de suivi et alimentent le processus de réparation de modèle. La réponse vivante devient la réponse du service virtuel.
- DYNAMIC : ce mode permet d'appeler un script ou un sous-processus pour toutes les demandes qui doivent renvoyer l'un des quatre autres modes. Cette approche est utile pour les demandes de suivi que seul le modèle de service virtuel peut voir.
- LEARNING : ce mode permet de réparer ou de corrige automatiquement le service virtuel pour obtenir la réponse nouvelle ou mise à jour du système dynamique.
- STAND_IN : ce mode route d'abord une demande vers le service virtuel (identique à celui du mode Most Efficient). Toutefois, si le service virtuel n'a pas de réponse, la demande est alors automatiquement routée vers le système dynamique.
- FAILOVER : ce mode route d'abord une demande vers le système dynamique (le même que le mode Live System). Toutefois, si le système dynamique n'a pas de réponse, la demande est alors automatiquement routée vers le service virtuel.

Par défaut : EFFICIENT

lisa.vse.body.cache.weight

Définit la taille en octets des valeurs de cache persistantes dans certains caches.

Valeur par défaut : 1024 * 1024 * 2 (2 Mo)

lisa.vse.metric.collect

Lorsque cette propriété est définie sur false, DevTest arrête la collecte d'échantillons périodiques qui permettent de calculer les valeurs des paramètres Response Time (Temps de réponse), Forced Delay (Retard forcé) et Transactions per Second (Transactions par seconde). Le reste des données de mesures suivies dans la table VSE_METRICS_TXN_COUNTS sont encore capturées, afin de générer les graphiques pour les éléments suivants :

- Graphiques de serveur
- Total Lifetime Transactions (Total des transactions pendant la durée de vie)

- Daily Transaction Counts (Nombres de transactions quotidiennes)
- Server Availability (Disponibilité du serveur)
- Graphiques de service
- Total Transactions per Day (Total de transactions par jour)

Lorsque cet élément est défini sur false, les éléments suivants dans la liste de graphiques de la console VSE n'indiquent aucune donnée :

- Graphiques de service
- Transaction Throughput (Débit de transactions)
- Transaction Hits and Misses (Transactions correspondantes et transactions non correspondantes)
- Response Time (Temps de réponse)
- Forced Delay (Retard forcé)
- Transactions Per Second (Transactions par seconde)

Valeur par défaut : true

lisa.vse.max.hard.errors

Permet de configurer le nombre d'erreurs qui entraînent l'arrêt d'un service de VSE.

Les erreurs sont encore indiquées dans la console VSE par un cercle rouge et un nombre d'erreurs, indépendamment de la valeur de cette propriété.

Pour indiquer le nombre d'erreurs qui entraînent l'arrêt du service, entrez un nombre valide, 0 ou une valeur supérieure. La valeur 0 indique qu'aucune erreur n'est permise et le service s'arrête à la première erreur.

Pour spécifier un nombre illimité d'erreurs, entrez un nombre négatif.

Si vous entrez une valeur non valide ou aucune valeur, la valeur par défaut de 3 erreurs est utilisée.

Valeur par défaut : 3

Remarque : Selon le type d'erreur et de modèle, le nombre d'erreurs final peut être supérieur à la valeur définie dans **lisa.vse.max.hard.errors**. L'arrêt du service virtuel peut prendre un certain temps. Lorsque l'arrêt est en cours, certains types d'erreurs peuvent continuer d'augmenter.

lisa.tcp.tcprecorder.close.immediately

Spécifie si les sockets sont fermés immédiatement à la fin (ou à l'arrêt) d'un enregistrement TCP.

Les sockets locaux (port d'écoute) et distants sont affectés.

Si une opération d'écriture ou de lecture est en cours sur un socket (s'il y a un long retard du serveur ou du client) et l'enregistrement est arrêté, cette propriété spécifie si la communication doit être arrêtée ou se poursuivre jusqu'à la fin.

Pour des conversations longues (par exemple, le téléchargement d'un fichier volumineux), cette propriété indique si la conversation doit se terminer ou se poursuivre jusqu'à la fin.

Valeurs : vide, true et false. Si la valeur est vide, la valeur est remplacée par défaut sur true.

Valeur par défaut : true

lisa.vse.body.cache.size

Spécifie la quantité de mémoire réservée à la mise en cache du corps non compressé des objets de demande et de réponse de VSE. Les unités sont en Mo.

Valeur par défaut : 10

lisa.vse.body.cache.timeout

Spécifie la durée de la mise en cache des corps non compressés des demandes et des réponses de VSE. Les unités sont en secondes.

Par défaut : 60

lisa.vse.argument.match.allow.whitespace

Cette propriété indique à VSE d'autoriser les espaces au début et à la fin dans les arguments de demande entrants lors de la mise en correspondance d'une image de service. La valeur par défaut est *false*, ce qui indique à VSE d'éliminer les espaces au début et à la fin des arguments de demande entrants avant d'effectuer la mise en correspondance de la demande avec une image de service.

Valeur par défaut : false

Ces propriétés permettent de définir les paramètres de messagerie de la solution IMS Connect, pour le protocole à utiliser à des fins d'enregistrement et de lecture.

lisa.vse.protocol.ims.encoding

Par défaut : EBCDIC

lisa.vse.protocol.ims.header.length

Valeur par défaut : 80

lisa.vse.session.timeout.ms

Si aucune session n'existe, VSE tente d'établir une correspondance entre la demande et les transactions de démarrage de chaque conversation dans l'image. En cas de correspondance, une nouvelle session est créée et la réponse pertinente est renvoyée. La session est conservée pendant 2 minutes après le dernier affichage par le VSE. Vous pouvez modifier ce comportement en définissant **lisa.vse.session.timeout.ms**. Si aucun démarrage de conversations ne correspond, aucune session ne sera créée et la liste de transactions sans état sera consultée par ordre de définition. En cas de correspondance, la réponse appropriée sera renvoyée. Sinon, la réponse Unknown request (Demande inconnue) sera envoyée.

Valeur par défaut : 120000

lisa.vse.tcp.oldio.read.timeout

Contrôle le délai d'attente de la fin d'une lecture TCP avant d'expirer et d'exécuter la logique de nouvelle tentative. La valeur par défaut de 500 millisecondes est suffisante pour les réseaux normaux ou bien paramétrés, pour lesquels la latence réseau n'est pas un problème. Pour les réseaux souffrant de retards supérieurs à 0,5 seconde, vous pouvez définir cette propriété avec un délai d'attente supérieur, pour laisser plus de temps avant une nouvelle tentative. La logique de nouvelle tentative est suffisante pour les communications sur le socket TCP, malgré une légère perte de débit (un nombre inférieur de transactions par seconde).

La fonctionnalité NIO doit être désactivée pour que la propriété **lisa.vse.tcp.oldio.read.timeout** puisse fonctionner. Définissez **lisa.vse.ssl.tcp.uses.nio** sur false.

Valeur par défaut : 500

lisa.vse.conversational.back.navigation.allowed

VSE prend en charge la navigation arrière non enregistrée pendant la lecture dans une transaction conversationnelle. Définissez cette propriété sur *true* pour activer la navigation arrière pour tous les services. La session conversationnelle suit les demandes reçues et permet d'utiliser une demande précédente pour une demande suivante.

Valeur par défaut : false

lisa.vse.tracking.delete.data.age

Définit la durée de conservation des informations de suivi de session, en heures.

Valeur par défaut : 8

lisa.vse.protocol.ims.response.includes.llll

Indique la présence du champ de longueur LLLL de 4 octets IMS Connect dans la charge utile de la demande. La valeur true inclut le champ LLLL. La valeur false l'exclut.

Valeur par défaut : false

lisa.vse.ims.connect.llzz.request

Indique la présence du champ de longueur LLZZ de 4 octets IMS Connect dans la charge utile de la demande. La valeur **true** inclut le champ LLZZ. La valeur **false** l'exclut.

Valeur par défaut : false

lisa.vse.ims.connect.llzz.response

Indique au protocole de générer le champ de longueur LLZZ de 4 octets IMS Connect dans les messages de réponse. La valeur **true** crée le champ LLZZ. La valeur **false** ne le crée pas.

Valeur par défaut : true

Propriétés du tableau de bord Enterprise Dashboard

lisa.enterprisedashboard.service.url

URL du serveur sur lequel se trouve le tableau de bord Enterprise Dashboard pour surveiller le registre.

Valeur par défaut : tcp://localhost:2003/EnterpriseDashboard

Ce sont les propriétés utilisées par DevTest pour se connecter à la base de données du tableau de bord Enterprise Dashboard. La connexion par défaut est établie avec la base de données Derby interne. Pour plus d'informations sur la connexion à des bases de données externes, reportez-vous à la section Configuration de base de données externe.

lisadb.dradis.poolName

Valeur par défaut : common

lisadb.pool.common.driverClass

Valeur par défaut : org.apache.derby.jdbc.ClientDriver

lisadb.pool.common.url

Valeur par défaut : jdbc:derby://localhost:1530/database/dradis.db;create=true

lisadb.pool.common.user

Valeur par défaut : app

lisadb.pool.common.password_enc

Pour définir le mot de passe, supprimez _enc à la fin du nom de propriété et ajoutez votre *mot de passe en texte brut* après le signe égal. Le mot de passe est automatiquement codé au démarrage.

Valeur par défaut : 76f271db3661fd50082e68d4b953fbee

Les propriétés de pool suivantes maintiennent le nombre de connexions au minimum lorsque DevTest est inactif.

lisadb.pool.common.minPoolSize

Valeur par défaut : 0

lisadb.pool.common.minPoolSize

Valeur par défaut : 0

lisadb.pool.common.initialPoolSize

Valeur par défaut : 0

lisadb.pool.common.maxPoolSize

Valeur par défaut : 10

lisadb.pool.common.acquireIncrement

Valeur par défaut : 1

lisadb.pool.common.maxIdleTime

Valeur par défaut : 45

lisadb.pool.common.idleConnectionTestPeriod

Valeur par défaut : 5

lisadb.internal.enabled

Permet de contrôler si le registre démarre une base de données Derby interne. Si vous avez configuré tous les composants précédents pour utiliser une base de données externe, vous pouvez enregistrer des ressources en définissant cette propriété sur *false*.

Valeur par défaut : true

lisa.default.keystore

Valeur par défaut : {{LISA_HOME}}webreckeys.ks

lisa.default.keystore.pass

Valeur par défaut : passphrase (phrase secrète)

Chiffrement des communications DevTest à DevTest. En général, le trafic réseau n'est pas chiffré. Pour utiliser le chiffrement, SSL est pris en charge par défaut. Au lieu d'utiliser tcp dans les noms de terminaux de serveur, utilisez ssl. Par exemple : ssl://hostname:2010/Registry. Vous n'êtes pas obligé de définir les propriétés suivantes, il suffit d'utiliser ssl dans les noms de terminaux (et le nom du serveur). Par exemple, vous pouvez démarrer un nouveau simulateur :

```
Simulator -name ssl://thishost:2014/Simulator -labName
ssl://regHost:2010/Registry
```

Un certificat autosigné interne par défaut est fourni sous **LISA_HOME\webreckeys.ks**. Pour un certificat plus fort, spécifiez la propriété **lisa.net.keyStore** et le mot de passe en clair dans **lisa.net.keyStore.password**.

Au démarrage suivant de DevTest, une chaîne chiffrée **lisa.net.keyStore.password_enc** remplace le mot de passe en clair.

lisa.net.keyStore

Valeur par défaut : {{LISA_HOME}}lisa.ks

lisa.net.keyStore.password

Il s'agit de l'emplacement dans lequel le certificat d'identification sera conservé. Dans le cas d'une installation de serveur, les clients doivent ajouter ce certificat à leur liste de serveurs fiables. Dans le cas d'une installation cliente et d'une authentification réciproque (client), ce certificat doit être ajouté au référentiel d'approbations côté serveur. Pour une installation cliente qui n'utilise pas l'authentification réciproque, ce paramètre n'est pas requis.

lisa.net.trustStore

Valeur par défaut : {{LISA_HOME}}lisa.ts

lisa.net.trustStore.password_enc

Valeur par défaut : 079f6a3d304a978146e547802ed3f3a4

Il s'agit de l'emplacement de stockage des certificats approuvés. Dans le cas d'une installation cliente, il contient une liste de serveurs sécurisés. Dans le cas d'une installation de serveur utilisant l'authentification réciproque (client), il s'agit de l'emplacement des certificats clients approuvés.

lisa.net.clientAuth

Indique si l'authentification réciproque doit être utilisée.

Valeur par défaut : false

lisa.net.default.protocol

Valeur par défaut : SSL ou TCP

Par défaut : SSL

Par défaut : ssl

Propriétés de licence relatives à l'utilisation d'un serveur proxy HTTP

laf.usehttpproxy.server

Valeur par défaut : true

laf.httpproxy.server

Définit le serveur proxy, au format *mon_serveur_proxy.com*.

laf.httpproxy.port

Si votre serveur proxy requiert des informations d'identification, laissez cette propriété vide pour utiliser l'authentification NTLM native.

Valeur par défaut : 3128

laf.httpproxy.domain

Définit le domaine de proxy, s'il est requis pour l'authentification NTLM.

laf.httpproxy.username

Facultatif

laf.httpproxy.password

Facultatif

laf.email.alert

Si une erreur de licence se produit, envoie un courriel. Reportez-vous à l'exemple suivant : Remplacez *mailhost* par le nom de votre hôte de messagerie. Remplacez *recipient@mycompany.com* par l'adresse électronique du destinataire. Remplacez *from@mycompany.com* par l'adresse électronique à partir de laquelle le courriel est envoyé.

Exemple : *mailhost,recipient@mycompany.com,from@mycompany.com*

Propriétés de SDK

Propriétés requises pour les exemples de kit de développement logiciel de DevTest :

asserts

Format : com.mycompany.lisa.AssertFileStartsWith

com.mycompany.lisa.AssertFileStartsWith

Format :
com.itko.lisa.editor.DefaultAssertController,com.itko.lisa.editor.DefaultAssertEditor

filtres

Format : com.mycompany.lisa.FilterFileFirstLine

com.mycompany.lisa.FilterFileFirstLine

Format : com.itko.lisa.editor.FilterController,com.itko.lisa.editor.DefaultFilterEditor

noeuds

Format : com.mycompany.lisa.node.FTPTestNode

com.mycompany.lisa.node.FTPTestNode

Format :
com.mycompany.lisa.node.FTPTestNodeController,com.mycompany.lisa.node.FTPTestNodeEditor

Propriétés de SSL

Ces propriétés permettent de modifier le comportement par défaut pour la validation des certificats SSL.

ssl.checkexpiry

Une valeur true indique de valider les dates de validité du certificat.

Valeur par défaut : false

ssl.checkcrl

Une valeur true indique de valider la liste de révocation de certificats spécifiée dans le certificat.

Valeur par défaut : false

Activer un certificat de client et un mot de passe pour la connexion SSL (utilisée par l'étape HTTP et l'étape SOAP brute. Egalement utilisée par l'étape de service Web si elle n'est pas remplacée).

ssl.client.cert.path

Chemin complet du référentiel de clés.

ssl.client.cert.pass

Mot de passe du référentiel de clés. Ce mot de passe est automatiquement chiffré lors de l'exécution de DevTest.

ssl.client.key.pass

Mot de passe facultatif pour l'entrée de clé si un référentiel de clés JKS est utilisé et que la clé a un mot de passe différent du mot de passe du référentiel de clés. Ce mot de passe est automatiquement chiffré lors de l'exécution de DevTest.

Remplacer le certificat de client et le mot de passe pour la connexion SSL (ssl.client.cert.path et ssl.client.cert.pass) pour l'étape de service Web.

ws.ssl.client.cert.path

Chemin complet du référentiel de clés.

ws.ssl.client.cert.pass

Mot de passe du référentiel de clés. Ce mot de passe est automatiquement chiffré lors de l'exécution de DevTest.

ws.ssl.client.key.pass

Mot de passe facultatif pour l'entrée de clé si un référentiel de clés JKS est utilisé et que la clé a un mot de passe différent du mot de passe du référentiel de clés. Ce mot de passe est automatiquement chiffré lors de l'exécution de DevTest.

Propriétés d'autorisation HTTP

Les informations d'identification suivantes sont automatiquement chiffrées lors de l'exécution de DevTest. Pour réinitialiser les valeurs, utilisez des noms de propriété non chiffrés. Si vous voulez utiliser l'autorisation NTLM native (Windows uniquement), laissez les commentaires de ces paramètres.

lisa.http.domain

Il s'agit du nom de domaine ; utilisez cette option pour NTLM.

lisa.http.user

Nom d'utilisateur

lisa.http.pass

Password (Mot de passe)

lisa.http.preemptiveAuthenticationType

Permet d'envoyer de manière préventive les informations d'autorisation au lieu d'attendre une demande d'accès.

Valeurs : basic, ntlm, negotiate

Valeur par défaut : ntlm

lisa.http.forceNTLMv1

NTLMv2 est utilisé par défaut, mais en définissant cette propriété sur *true*, l'utilisation de NTLMv1 peut être forcée lorsque l'authentification Windows intégrée native n'est pas utilisée.

Valeur par défaut : true

Propriétés d'authentification Kerberos

lisa.java.security.auth.login.config

Indique l'emplacement du fichier de configuration de connexion.

lisa.java.security.krb5.conf

Emplacement du fichier de configuration Kerberos utilisé pour remplacer tous les emplacements prédéfinis.

lisa.http.kerberos.principal

Nom du principal à utiliser pour la connexion lorsque DevTest est utilisé pour l'authentification par principal et mot de passe. Ce paramètre est chiffré lors du démarrage de DevTest Workstation.

lisa.http.kerberos.pass

Mot de passe à utiliser pour la connexion lorsque DevTest est utilisé pour l'authentification par principal et mot de passe. Ce paramètre est chiffré lors du démarrage de DevTest Workstation.

Propriétés de serveur proxy HTTP

lisa.http.webProxy.host

Nom ou adresse IP de l'ordinateur

lisa.http.webProxy.nonProxyHosts

Nom ou adresse IP de l'ordinateur à exclure de l'authentification du proxy. Pour saisir plusieurs valeurs, séparez-les par des barres verticales (|). Utilisez l'astérisque (*) comme caractère générique.

Valeur par défaut : 127.0.0.1

lisa.http.webProxy.port

lisa.http.webProxy.ssl.host

Nom ou adresse IP de l'ordinateur

lisa.http.webProxy.ssl.nonProxyHosts

Nom ou adresse IP de l'ordinateur à exclure de l'authentification du proxy SSL. Pour saisir plusieurs valeurs, séparez-les par des barres verticales (|). Utilisez l'astérisque (*) comme caractère générique.

Valeur par défaut : 127.0.0.1

lisa.http.webProxy.ssl.port

Laissez cette propriété vide pour utiliser l'authentification NTLM intégrée.

lisa.http.webProxy.host.domain

Utilisée pour l'authentification NTLM

lisa.http.webProxy.host.account

lisa.http.webProxy.host.credential

lisa.http.webProxy.nonProxyHosts.excludeSimple

Permet d'exclure des noms d'hôte simples de l'utilisation de proxy.

Valeur par défaut : true

lisa.http.webProxy.preemptiveAuthenticationType

Permet d'envoyer de manière préventive les informations d'autorisation au lieu d'attendre une demande d'accès.

Valeurs : basic, ntlm

Valeur par défaut : ntlm

lisa.http.timeout.connection

Délai d'expiration HTTP (en millisecondes). Pour étendre le délai d'expiration indéfiniment, définissez la valeur sur zéro.

Valeur par défaut : 15000

lisa.http.timeout.socket

Délai d'expiration HTTP (millisecondes). Pour étendre le délai d'expiration indéfiniment, définissez la valeur sur zéro.

Valeur par défaut : 180000

Paramètres de sérialisation XML

lisa.toxml.serialize.mode=NOREFERENCES

NOREFERENCES signifie qu'une arborescence simple est rendue. Les références circulaires ne sont pas permises. XPATHREFERENCES signifie que la notation XPath est utilisée pour les références. Vous pouvez utiliser des références circulaires. Si une référence circulaire existe, elle renvoie à la valeur XPATHREFERENCES. Toutefois, les sérialisations antérieures à la version 3.6 qui sont relues échouent et peuvent vous obliger à définir la valeur par défaut XPATHREFERENCES.

Gestion de stations de travail

lisa.ui.admin.tr.control=no

lisa.ws.jms.SoapAction.quoted=false

SOAP via JMS avec TIBCO BusinessWorks/Active Matrix : l'en-tête SOAPAction doit être placé entre guillemets.

Lorsque DevTest sérialise un paramètre date, time ou dateTime, les formats suivants sont utilisés par défaut. Pour modifier les formats/fuseau horaire par défaut, utilisez ces propriétés. Vous pouvez également les définir de façon dynamique pendant l'exécution du scénario de test. Vous pouvez définir un format qui ne respecte pas les spécifications du schéma XML. Toutefois, cela n'est pas recommandé (sauf dans le cas d'un scénario de test négatif).

lisa.ws.ser.dateFormat=yyyy-MM-dd

lisa.ws.ser.timeFormat=HH:mm:ss.SSS'Z'

lisa.ws.ser.timeFormat.timeZone=GMT

lisa.ws.ser.dateTimeFormat=yyyy-MM-dd'T'HH:mm:ss.SSS'Z'

lisa.ws.ser.dateTimeFormat.timeZone=GMT

ws.raw.format=true

Pour formater la réponse SOAP des étapes SOAP brutes, définissez cette propriété sur **true. V**ous pouvez également utiliser ce formatage de façon dynamique lors de l'exécution.

lisa.ws.endpoint.fullautoprop=false

L'URL de terminal de service Web entière est automatiquement convertie en une propriété DevTest unique. Pour convertir l'URL pour utiliser les propriétés WSSERVER et WSPORT, définissez cette propriété sur **false**.

stats.unix.xml.folder={ {LISA_HOME } }/umetrics

Usurpation de l'adresse IP

lisa.ipspoofing.interfaces=2-34-56-78-90-AB, eth0, Realtek PCIe GBE Family Controller

Interfaces : liste d'interfaces séparées par des virgules qui est utilisée pour l'usurpation de l'adresse IP. Vous pouvez nommer ces interfaces à l'aide de l'adresse MAC (kit de développement Java 1.6+), du nom d'interface ou du nom d'affichage de l'interface.

lisa.ui.useNativeFileDialog=true

Impose l'utilisation d'une boîte de dialogue de fichier native.

Propriétés des éléments récents de la fenêtre Quick Start (Démarrage rapide)

lisa.prefill.recent=10

lisa.quickstart.recent=5

Ces propriétés permettent de spécifier le nombre maximum d'éléments récents à afficher dans l'onglet Open recent (Ouvrir un document récent) de la fenêtre Quick Start (Démarrage rapide) (5 au minimum) et dans les zones de liste modifiable Prefill (Préremplir) (10 au minimum).

Propriétés LISA Invoke

lisa.portal.invoke.base.url=/lisa-invoke

lisa.portal.invoke.report.url=/reports

lisa.portal.invoke.server.report.directory=lisa.tmpdirlisa.portal.invoke.report.url

lisa.portal.invoke.test.root=d:/lisatests/

Propriétés de nom d'hôte du serveur

lisa.net.externalIP=localhost

Pour que le registre soit accessible à distance, cette valeur doit être l'adresse IP externe ou le nom DNS. Ce paramètre est référencé dans les propriétés de connexion pertinentes. Si vous conservez la valeur localhost, elle est automatiquement remplacée dans l'adresse IP externe du serveur de registre lorsqu'elle est envoyée aux applications connectées.

Propriétés de chiffrement des communications DevTest à DevTest

En général, le trafic réseau n'est pas chiffré. Le chiffrement SSL est pris en charge de manière native. Au lieu d'utiliser "tcp" pour nommer vos terminaux de serveur, utilisez "ssl". Par exemple : **ssl://hostname:2010/Registry**. Vous ne devez définir aucune des propriétés suivantes, car nommer les terminaux (et le nom de serveur) avec l'indicateur ssl est suffisant. Par exemple, pour démarrer un nouveau simulateur : **Simulator -name ssl://thishost:2014/Simulator -labName ssl://regHost:2010/Registry**.

LISA_HOME\webreckeys.ks est un certificat autosigné interne par défaut. Pour un certificat plus fort, spécifiez la propriété **lisa.net.keyStore** et le mot de passe en clair dans **lisa.net.keyStore.password**. Au démarrage suivant de DevTest, une chaîne chiffrée **lisa.net.keyStore.password_enc** remplace le mot de passe en clair.

lisa.default.keystore={{LISA_HOME}}webreckeys.ks

lisa.default.keystore.pass=passphrase

Les paramètres suivants indiquent l'emplacement dans lequel le certificat d'identification est stocké. Pour une installation de serveur, cette valeur correspond au certificat à ajouter à la liste de serveurs fiables. Dans le cas d'une installation de client utilisant l'authentification réciproque (client), cette valeur correspond au certificat à ajouter au référentiel d'approbations côté serveur. Pour une installation de client qui n'utilise pas l'authentification réciproque, vous ne devez pas spécifier ce paramètre.

lisa.net.keyStore={{LISA_HOME}}lisa.ks

lisa.net.keyStore.password=PlainTextPasswordWilBeConvertedToEncrypted

Les paramètres suivants indiquent l'emplacement dans lequel les certificats fiables sont stockés. Pour une installation de client principalement, ce champ contient une liste de serveurs fiables. Pour une installation de serveur utilisant l'authentification réciproque (client), ce champ est l'emplacement dans lequel placer les certificats de client fiables.

lisa.net.trustStore={{LISA_HOME}}lisa.ts

lisa.net.trustStore.password_enc=079f6a3d304a978146e547802ed3f3a4

Le paramètre suivant indique si l'authentification réciproque est utilisée.

lisa.net.clientAuth=false

Le paramètre suivant permet de définir le protocole par défaut pour des connexions ActiveMQ. La valeur par défaut est **tcp**.

lisa.net.default.protocol=tcp

Propriétés de l'agent DevTest et de CAI

lisa.pathfinder.on

Si vous rencontrez des erreurs d'agent et que vous ne disposez pas d'une licence pour l'agent DevTest, utilisez ce paramètre pour le désactiver.

Valeur par défaut : false

Propriétés d'IBM WebSphere MQ

La valeur de chaîne est fournie par un ensemble de données tel que le générateur de code unique.

lisa.mq.correlation.id=string-value

(Valeur d'exécution) Permet de définir l'ID de corrélation pour un message MQ sortant.

lisa.mq.correlation.id.bytes=byte[]

(Valeur d'exécution) Permet de définir l'ID de corrélation en tant que byte[] pour les valeurs non imprimables.

lisa.mq.message.id=string-value

(Valeur d'exécution) Permet de définir l'ID de message pour un message MQ sortant.

lisa.mq.message.id.bytes=byte[]

(Valeur d'exécution) Permet de définir l'ID de message en tant que byte[] pour les valeurs non imprimables.

Propriétés de JMS

La valeur de chaîne est souvent fournie par un ensemble de données tel que le générateur de code unique.

lisa.jms.correlation.id

Valeur d'exécution qui définit l'ID JMSCorrelationID pour un message JMS sortant.

Valeur : valeur de chaîne

lisa.jms.ttl.milliseconds

Valeur d'exécution qui permet de définir la durée de vie d'un message JMS sortant.

Valeur : valeur numérique

jms.always.close.jndi

Si la valeur définie est true, l'étape JMS ferme toujours le contexte JNDI à la fin de son exécution.

Valeurs : true ou false

Propriétés diverses

lisa.coord.failure.list.size

Si un trop grand nombre d'erreurs est signalé dans un test, le coordinateur utilise tout l'espace de segment de mémoire disponible. Lorsque il n'y a plus d'espace de segment de mémoire disponible, DevTest Server doit être redémarré. Cette situation est problématique si plusieurs tests sont en cours d'exécution et qu'un test simulé ne renvoie que des erreurs. Cette propriété limite la taille de la liste d'échecs du coordinateur.

Valeur par défaut : 15

testexec.lite.longMsgLen

Pour afficher le texte complet des longues réponses, définissez cette propriété sur la longueur de votre réponse la plus longue. Exécutez le test, capturez les données requises, vérifiez que la réponse correcte est bien renvoyée, puis mettez la ligne ajoutée en commentaire dans le fichier local.properties pour revenir à la longueur par défaut.

Valeur par défaut : 1024

java.rmi.server.hostname

Impose les communications via un NIC spécifique de l'ordinateur.

Valeur : adresse IP du NIC sélectionné

lisa.xml.xpath.computeXPath.alwaysUseLocalName

Cette propriété spécifie si la fonction XPath local-name() est toujours utilisée pour la génération d'expression XPath. La valeur par défaut est false, ce qui signifie que la fonction local-name() est utilisée uniquement si nécessaire. Pour générer une expression XPath qui fonctionne indépendamment d'un espace de noms de noeud XML, définissez la valeur de la propriété sur true.

Valeur par défaut : false

lisa.commtrans.ctstats

Pour capturer les informations permettant de remplir le rapport Cumulative HTTP Traffic Summary (Récapitulatif du trafic HTTP cumulatif), entrez cette propriété dans un des fichiers de propriété personnalisés.

Valeur par défaut : true

gui.viewxml.maxResponseSize

Cette propriété permet de spécifier la taille à laquelle la vue d'une réponse XML dans un éditeur ou dans l'ITR est complète, sans qu'aucune vue DOM soit fournie.

Valeur par défaut : 5 Mo

rpt.cleaner.initDelayMin

Valeur par défaut : 10

rpt.cleaner.pulseMin

Ces propriétés ont été ajoutées pour augmenter la fréquence d'exécution du thread de nettoyage de rapport. La première propriété indique le délai initial en minutes avant le démarrage du nettoyeur. La deuxième correspond à l'intervalle entre chaque exécution, en minutes.

Par défaut : 60

lisa.LoadHistoryWriteSupport.max.errors

Définissez cette propriété pour limiter le nombre d'erreurs signalées, afin d'empêcher la sauvegarde du générateur de rapports.

Valeur par défaut : 50

lisa.metric.initialization.timeout

L'initialisation du collecteur de mesures se produit durant la simulation du test ou de la suite et que le test ou la suite ne démarre pas avant l'initialisation des collecteurs. La valeur par défaut de 90000 signifie que l'initialisation expire après 90 secondes (par collecteur).

Valeur par défaut : 90000

lisa.graphical.xml.diff.report.numberofextralines

Lorsque la comparaison de contenu XML graphique affiche une erreur, cette propriété contrôle le nombre de lignes affichées avant et après les lignes différentes. La valeur par défaut est de deux lignes avant et deux lignes après.

Valeurs : 0, 1 et 2.

Valeur par défaut : 2

lisa.gui.dashboard.sleep.ms

Une fonction sleep() indique qu'il y a une courte mise en veille avant que le tableau de bord soit actualisé à la fin du test. La fonction sleep() offre un délai supplémentaire aux collecteurs de mesures afin qu'ils puissent terminer leur travail. Ajustez ce délai à l'aide de cette propriété.

Format : millisecondes

lisa.scripting.default.language

Désigne le langage de script par défaut à utiliser.

Valeurs :

- applescript (OS X)
- beanshell
- freemarker
- groovy
- javascript
- velocity

Valeur par défaut : beanshell

Propriétés de durée de vie de session d'utilisateur

Toutes les durées de vie de session d'utilisateur dans les différents modules DevTest sont configurables à l'aide de propriétés.

Les propriétés suivantes représentent les propriétés de contrôle d'accès de session et leurs valeurs par défaut :

registry.max.user.lifetime.seconds

La propriété **registry.max.user.lifetime.seconds** est la propriété de durée de vie de session principale qui détermine la durée de validité d'une session d'utilisateur après la dernière activité effectuée par un utilisateur dans cette session.

Spécifie la durée en secondes de conservation du jeton de sécurité d'authentification unique dans la mémoire avant l'omission par la station de travail DevTest Workstation de l'authentification des applications Web de console DevTest (par exemple : génération de rapports, service de validation en continu, CAI, console de serveur), du portail DevTest et de la station de travail DevTest Workstation.

Lorsque vous cliquez sur un bouton pour accéder à une application Web de console, DevTest ignore l'authentification de la liste de contrôle d'accès à l'aide d'un jeton de sécurité d'authentification unique. Ce jeton de sécurité d'authentification unique est stocké dans la mémoire pour permettre sa réutilisation. Toutefois, si DevTest Workstation est inactif pendant 20 minutes, ou la durée en secondes spécifiée pour cette propriété, ce jeton de sécurité d'authentification unique expire. Il requerra donc une réauthentification. Si vous démarrez DevTest Workstation, vous pouvez cliquer sur ces boutons et accéder directement à l'application Web sans entrer d'ID d'utilisateur, ni de mot de passe. Lorsqu'aucune activité n'est effectuée pendant 20 minutes et que vous cliquez sur ces boutons, DevTest Workstation vous invitera à spécifier un ID d'utilisateur et un mot de passe pour qu'un nouveau jeton de sécurité soit conservé dans la mémoire.

Valeur par défaut : 1200

Les modules suivants ont une durée de vie de session d'utilisateur, mais ils communiquent toujours avec le registre pour déterminer si la session est encore active.

vse.max.user.lifetime.seconds=1200

console.max.user.lifetime.seconds=1200

coordinator.max.user.lifetime.seconds=1200

simulator.max.user.lifetime.seconds=30

Vous pouvez configurer l'intervalle d'actualisation de console à l'aide de la propriété suivante.

lisa.portal.server.console.polling.interval.seconds=5

Spécifie l'intervalle auquel la console DevTest vérifie si la session actuelle est valide et n'a pas expiré.

user.session.check.interval.seconds=60

Fichier de propriétés de site

Les propriétés du fichier **site.properties** sont envoyées à partir du registre vers toutes les applications ou services DevTest qui s'y connectent. Ces propriétés ont priorité sur toutes les propriétés définies localement dans **lisa.properties**. Toutefois, elles n'ont pas priorité sur les propriétés définies dans **local.properties** ou dans la ligne de commande à l'aide de l'option de ligne de commande -D.

Les composants suivants utilisent une base de données : le moteur de génération de rapports, l'intermédiaire, VSE et la liste de contrôle d'accès. Par défaut, ces composants utilisent le pool de connexions **commun**. Toutefois, vous pouvez définir un pool distinct pour chaque composant ou un mélange des deux. Pour définir un nouveau pool de connexions, ajoutez des propriétés telles que **lisadb.pool.newpool.url**.
L'implémentation de pool sous-jacente est le pool c3p0 open source. DevTest transfère les différentes propriétés. Pour plus d'informations sur les paramètres c3p0 disponibles, consultez le site
http://www.mchange.com/projects/c3p0/index.html#configuration_properties.

Il s'agit des propriétés par défaut que DevTest utilise pour se connecter à la base de données Derby interne par défaut. Pour configurer DevTest pour utiliser une base de données externe, reportez-vous au fichier **site.properties** approprié dans le répertoire **LISA_HOME\database.**

lisadb.reporting.poolName

Valeur par défaut : common

lisadb.vse.poolName

Valeur par défaut : common

lisadb.acl.poolName

Valeur par défaut : common

lisadb.broker.poolName

Valeur par défaut : common

lisadb.pool.common.driverClass

Valeur par défaut : org.apache.derby.jdbc.ClientDriver

lisadb.pool.common.url

Valeur par défaut : jdbc:derby://localhost:1528/database/lisa.db;create=true

lisadb.pool.common.user

Valeur par défaut : rpt

lisadb.pool.common.password_enc

Valeur par défaut : 76f271db3661fd50082e68d4b953fbee

lisadb.pool.common.minPoolSize

Valeur par défaut : 0

lisadb.pool.common.initialPoolSize

Valeur par défaut : 0

lisadb.pool.common.maxPoolSize

Valeur par défaut : 10

lisadb.pool.common.acquireIncrement

Valeur par défaut : 1

lisadb.pool.common.maxIdleTime

Valeur par défaut : 45

lisadb.pool.common.idleConnectionTestPeriod

Par défaut : 5

lisadb.internal.enabled

Permet de contrôler si le registre démarre une base de données Derby interne.

Valeur par défaut : true

lisa.pathfinder.on

Valeur par défaut : true

Paramètres DCM

lisa.dcm.labstartup.min=6

lisa.dcm.lisastartup.min=4

lisa.dcm.lisashutdown.min=2

lisa.dcm.lab.cache.sec=<180 default>

lisa.dcm.lab.factories=com.itko.lisa.cloud.serviceMesh.ServiceMeshCloudSupport;com.itko.lisa.cloud.vCloud.vCloudDirectorCloudSupport;;

lisa.dcm.SERVICEMESH.baseUri=<url>

lisa.dcm.SERVICEMESH.userId=<userId>

lisa.dcm.SERVICEMESH.password=<password>

lisa.dcm.vCLOUD.baseUri=<https://<fqdn>/api/versions>

lisa.dcm.vCLOUD.userId=<userId>

lisa.dcm.vCLOUD.password=<password>

lisa.net.timeout.ms=60000

logging.properties

log4j.rootCategory

Valeur par défaut : INFO,A1

Pour fournir une journalisation centralisée pour les exécutions de test cloud, ajoutez le suffixe de **registre** et supprimez les commentaires des propriétés du suffixe de registre dans le fichier logging.properties. Par exemple :

```
log4j.rootCategory=INFO,A1,registry
```

Les lignes suivantes ajustent les niveaux de journalisation de bibliothèques tierces utilisées par DevTest. La spécification de niveaux de journalisation permet d'éviter d'encombrer les journaux avec des messages sans rapport avec DevTest.

log4j.logger.com.teamdev

Par défaut : WARN

log4j.logger.EventLogger

Par défaut : WARN

log4j.logger.org.apache

Par défaut : ERROR

log4j.logger.com.smardec

Par défaut : ERROR

log4j.logger.org.apache.http

Par défaut : ERROR

log4j.logger.org.apache.http.header

Par défaut : ERROR

log4j.logger.org.apache.http.wire

Par défaut : ERROR

log4j.logger.com.mchange.v2

Par défaut : ERROR

log4j.logger.org.hibernate

Par défaut : WARN

log4j.logger.org.jfree

Par défaut : ERROR

log4j.logger.com.jniwrapper

Par défaut : ERROR

log4j.logger.sun.rmi

Par défaut : INFO

log4j.logger.com.itko.util.ThreadDumper

Par défaut : INFO

log4j.logger.profiler

Définissez cette propriété sur *INFO* si vous voulez que les événements de profil soient journalisés :

Par défaut : OFF

log4j.appender.A1

Valeur par défaut : com.itko.util.log4j.TimedRollingFileAppender

log4j.appender.A1.File

Valeur par défaut : ${lisa.tmpdir}/${LISA_LOG}

log4j.appender.A1.MaxFileSize

Valeur par défaut : 10 Mo

log4j.appender.A1.MaxBackupIndex

Par défaut : 5

log4j.appender.A1.layout

Valeur par défaut : org.apache.log4j.EnhancedPatternLayout

log4j.appender.A1.layout.ConversionPattern

Valeur par défaut : %d{ISO8601}{UTC}Z (%d{HH:mm}) [%t] %-5p %-30c - %m%n

log4j.logger.VSE

Conservez un journal différent pour des événements de correspondance/non-correspondance de transactions du VSE : cela simplifie le débogage.

Pour les systèmes de production, remplacez la valeur INFO par WARN. La journalisation peut ralentir les systèmes dont les taux de transactions sont élevés. N'annulez pas simplement la mise en commentaire de la ligne suivante. Définissez explicitement le niveau de journalisation sur OFF (Désactiver) ou WARN (Avertissement) au lieu d'INFO.

Valeur par défaut : INFO,VSEAPP

log4j.additivity.VSE

Pour ajouter la journalisation de VSE à d'autres destinations de journal, mettez cette ligne en commentaire.

Valeur par défaut : false

log4j.appender.VSEAPP

Valeur par défaut : com.itko.util.log4j.TimedRollingFileAppender

log4j.appender.VSEAPP.File

Valeur par défaut : ${lisa.tmpdir}/vse_matches.log

log4j.appender.VSEAPP.MaxFileSize

Valeur par défaut : 10 Mo

log4j.appender.VSEAPP.MaxBackupIndex

Valeur par défaut : 20

log4j.appender.VSEAPP.layout

Valeur par défaut : org.apache.log4j.EnhancedPatternLayout

log4j.appender.VSEAPP.layout.ConversionPattern

Valeur par défaut : %d{ISO8601}{UTC}Z (%d{HH:mm})[%t] %-5p - %m%n

Définissez un journal distinct pour les événements consultatifs. Ce journal informe des problèmes de configuration potentiels, des fuites de mémoire potentielles, etc. Il est délibérément enregistré dans un journal distinct de celui du journal d'application afin de réduire les interférences.

log4j.logger.ADVICE

Valeur par défaut : INFO, ADVICE_APP

log4j.additivity.ADVICE

Valeur par défaut : false

log4j.appender.ADVICE_APP

Valeur par défaut : org.apache.log4j.RollingFileAppender

log4j.appender.ADVICE_APP.File

Valeur par défaut : ${lisa.tmpdir}/advice.log

log4j.appender.ADVICE_APP.MaxFileSize

Valeur par défaut : 10 Mo

log4j.appender.ADVICE_APP.MaxBackupIndex

Valeur par défaut : 20

log4j.appender.ADVICE_APP.layout

Valeur par défaut : org.apache.log4j.EnhancedPatternLayout

log4j.appender.ADVICE_APP.layout.ConversionPattern

Valeur par défaut : %d{ISO8601}{UTC}Z (%d{HH:mm}) %-5p - %m%n

S'il est activé, les fils de discussion réguliers sont envoyés ici. DevTest écrit des journaux au niveau INFO. Par conséquent, pour obtenir des fils de discussion, remplacez WARN dans la ligne suivante par INFO, même si des serveurs DevTest ou la station de travail DevTest Workstation sont en cours d'exécution. Cette action a pour résultat un fil de discussion dans le fichier nommé en 30 secondes. Pour plus d'informations, recherchez threadDump en **lisa.properties**. Cette action simplifie également l'obtention de fils de discussion pour déboguer des problèmes de performance. Dans la ligne suivante, remplacez WARN par INFO, patientez 1 ou 2 minutes, puis, rétablissez le paramètre sur WARN.

Vous pouvez également générer un fil de discussion à un point dans le temps à l'aide de l'application LISA_HOME/bin/ServiceManager. Par exemple, utilisez des outils Java standard tels que jstack, ou émettez la commande suivante :

`ServiceManager -threadDump tcp://hostname:2014/Simulator`

log4j.logger.threadDumpLogger

Valeur par défaut : WARN, THREAD_DUMPS

log4j.additivity.threadDumpLogger

Valeur par défaut : false

log4j.appender.THREAD_DUMPS

Valeur par défaut : org.apache.log4j.RollingFileAppender

log4j.appender.THREAD_DUMPS.File

Valeur par défaut : ${lisa.tmpdir}/threadDumps/TD_${LISA_LOG}

log4j.appender.THREAD_DUMPS.MaxFileSize

Valeur par défaut : 10 Mo

log4j.appender.THREAD_DUMPS.MaxBackupIndex

Valeur par défaut : 20

log4j.appender.THREAD_DUMPS.layout

Valeur par défaut : org.apache.log4j.EnhancedPatternLayout

log4j.appender.THREAD_DUMPS.layout.ConversionPattern

Valeur par défaut : %d{ISO8601}{UTC}Z (%d{HH:mm}) [%t] %-5p - %m%n

log4j.appender.registry

Met en miroir la journalisation DevTest dans le registre (distant).

Valeur par défaut : com.itko.lisa.net.LoggingToRegistryAppender

log4j.appender.registry.layout

Valeur par défaut : org.apache.log4j.EnhancedPatternLayout

log4j.appender.registry.layout.ConversionPattern

Valeur par défaut : %d{ISO8601}{UTC}Z [%t] %-5p - %m%n

Les propriétés suivantes affichent les demandes et les réponses pour le trafic HTTP dans le vse.log. Ces informations sont utiles pour déboguer un service virtuel qui renvoie la réponse No match found (Correspondance introuvable). Ce résultat peut indiquer que le service virtuel n'a pas reçu la demande attendue.

Ces messages de journal révèlent les mêmes demandes et réponses affichées par TCPMON. Les demandes et les réponses s'affichent dans le vse.log, sans insérer TCPMON entre le service virtuel et son application cliente. Ces options journalisent uniquement des demandes et réponses HTTP. Ces options peuvent aider à déboguer des services virtuels, notamment lorsqu'un service virtuel répond par un message inattendu et que vous voulez faire correspondre une demande brute avec la réponse brute.

log4j.logger.com.itko.lisa.vse.http.Transaction

Permet d'imprimer les demandes transmises via cette classe dans le vse.log. Ces messages de journal commencent par Raw playback response (Réponse de lecture brute).

Valeurs : TRACE, null

Valeur par défaut : null

log4j.logger.com.itko.lisa.vse.stateful.protocol.http.Coordinator

Permet d'imprimer les demandes transmises via cette classe dans le vse.log. Ces messages de journal commencent par Raw request start (Démarrage de demande brute).

Valeurs : TRACE, null

Valeur par défaut : null

log4j.logger.com.itko.lisa.vse.stateful.protocol.http.HttpListenStep

Permet d'imprimer les demandes transmises via cette classe dans le vse.log. Ces messages de journal commencent par Raw request start (Démarrage de demande brute).

Valeurs : TRACE, null

Valeur par défaut : null

Glossaire

Archive de modèle (MAR)

Une *archive de modèle (MAR)* est le principal artefact de déploiement dans DevTest. Les fichiers MAR contiennent un actif principal, tous les fichiers secondaires qui sont requis pour exécuter l'actif principal, un fichier d'informations et un fichier d'audit. Pour plus d'informations, consultez la section Utilisation des archives de modèle (MAR) (page 277) de la rubrique *Utilisation de CA Application Test*.

assertion

Une *assertion* est un élément qui s'exécute après l'exécution d'une étape et de tous ses filtres. Les assertions vérifient que les résultats de l'étape sont conformes aux prévisions. Une assertion est généralement utilisée pour modifier le flux d'un scénario de test ou le modèle de service virtuel. Les assertions globales s'appliquent à chaque étape d'un scénario de test ou d'un modèle de service virtuel. Pour plus d'informations, consultez la section Assertions (page 142) de la rubrique *Utilisation de CA Application Test*.

Audit document (Document d'audit)

Un *document d'audit* permet de définir des critères de réussite d'un test, ou d'un ensemble de tests dans une suite. Pour plus d'informations, consultez la section Génération de documents d'audit (page 258) dans la rubrique *Utilisation de CA Application Test.*

companion (compagnon)

Un *Companion (Compagnon)* est un élément exécuté avant et après chaque exécution de scénario de test. Les compagnons sont comparables à des filtres applicables à l'ensemble du scénario de test, par opposition à des étapes de test spécifiques. Les compagnons sont utilisés pour configurer le comportement global dans le scénario de test. Pour plus d'informations, consultez la section Compagnons (page 173) de la rubrique *Utilisation de CA Application Test*.

Configuration

Une *configuration* est une collection nommée de propriétés qui spécifient en général des valeurs propres à un environnement pour le système testé. La suppression de données d'environnement codées de manière irréversible permet d'exécuter un scénario de test ou un modèle de service virtuel au niveau d'environnements différents en modifiant simplement des configurations. La configuration par défaut dans un projet est appelée project.config. Un projet peut contenir de nombreuses configurations, mais une seule configuration peut être active à la fois. Pour plus d'informations, consultez la section Configurations (page 108) de la rubrique *Utilisation de CA Application Test*.

Conversation tree (Arborescence des conversations)

Une *arborescence des conversations* est un ensemble de noeuds liés qui représentent des chemins de conversation pour les transactions avec état dans une image de service virtuel. Chaque noeud porte une étiquette de nom d'opération ; par exemple : withdrawMoney. Exemple de chemin de conversation pour un système d'opérations bancaires : getNewToken, getAccount, withdrawMoney, deleteToken. Pour plus d'informations, reportez-vous à la rubrique *Utilisation de CA Service Virtualization.*

coordinator (coordinateur)

Le coordinateur reçoit les informations d'exécution de test sous forme de document et coordonne les tests exécutés sur un ou plusieurs serveurs de simulation. Pour plus d'informations, consultez la section Coordinator Server (Serveur de coordination) (page 13) de la rubrique *Utilisation de CA Application Test*.

data protocol (protocole de données)

Un *protocole de données* est également appelé gestionnaire de données. Dans CA Service Virtualization, un protocole de données est responsable de la gestion de l'analyse des demandes. Certains protocoles de transport autorisent (ou requièrent) un protocole de données auquel le job de création de demandes est délégué. C'est pourquoi le protocole doit connaître la charge utile de la demande. Pour plus d'informations, consultez la section Utilisation de protocoles de données dans la rubrique *Utilisation de CA Service Virtualization*.

Data set (Ensemble de données)

Un *Data set (Ensemble de données)* est une collection de valeurs que vous pouvez utiliser pour définir des propriétés dans un scénario de test ou un modèle de service virtuel lors de l'exécution. Les ensembles de données fournissent un mécanisme permettant d'introduire des données de test externes dans un scénario de test ou un modèle de service virtuel. Vous pouvez créer des ensembles de données internes à DevTest, ou de manière externe ; par exemple, dans un fichier ou une table de base de données. Pour plus d'informations, consultez la section Ensembles de données (page 162) de la rubrique *Utilisation de CA Application Test*.

desensitize (désensibiliser)

La *désensibilisation* consiste à convertir des données sensibles par des valeurs de substitution définies par l'utilisateur. Par exemple, les numéros de carte de crédit et les numéros de sécurité sociale sont des données sensibles. Pour plus d'informations, consultez la section Désensibilisation de données de la rubrique *Utilisation de CA Service Virtualization*.

Event (Evénement)

Un *Event (Evénement)* est un message sur une action qui s'est produite. Vous pouvez configurer des événements au niveau d'un scénario de test ou d'un modèle de service virtuel. Pour plus d'informations, consultez la section Introduction aux événements (page 260) de la rubrique *Utilisation de CA Application Test*.

Filter (Filtre)

Un *filtre* est un élément exécuté avant et après une étape. Les filtres permettent de traiter les données des résultats ou de stocker des valeurs dans des propriétés. Les filtres globaux s'appliquent à chaque étape d'un scénario de test ou d'un modèle de service virtuel. Pour plus d'informations, consultez la section Filtres (page 124) de la rubrique *Utilisation de CA Application Test*.

Groupe

Un *groupe* ou un *groupe de services virtuels* est une collection de services virtuels portant la même balise de groupe. Cela permet de les surveiller ensemble dans la console VSE.

Interactive Test Run (ITR) (Exécuter un test interactif)

L'utilitaire *Interactive Test Run (ITR)* (Exécuter un test interactif) permet d'exécuter un scénario de test ou un modèle de service virtuel étape par étape. Vous pouvez changer le scénario de test ou le modèle de service virtuel lors de l'exécution et le réexécuter pour vérifier les résultats. Pour plus d'informations, consultez la section Utilisation de l'utilitaire Interactive Test Run (ITR) (Exécuter un test interactif) (page 293) de la rubrique *Utilisation de CA Application Test.*

Lab (Laboratoire)

Un *laboratoire* est un conteneur logique pour un ou plusieurs membres de laboratoire. Pour plus d'informations, consultez la section Laboratoires et membres de Laboratoire (page 366) dans la rubrique *Utilisation de CA Application Test.*

magic date (date magique)

Pendant un enregistrement, un analyseur de dates analyse les demandes et les réponses. Une valeur correspondant à une définition étendue de formats de la date est convertie en *date magique*. Les dates magiques permettent de vérifier que le modèle de service virtuel fournit des valeurs de date explicites dans des réponses. Exemple e date magique : {{=doDateDeltaFromCurrent("yyyy-MM-dd","10");/*2012-08-14*/}. Pour plus d'informations, consultez la section Chaînes et dates magiques de la rubrique *Utilisation de CA Service Virtualization.*

Magic string (Chaîne magique)

Une *chaîne magique* est une chaîne générée pendant la création d'une image de service. Une chaîne magique est utilisée pour vérifier que les réponses fournies par le modèle de service virtuel contiennent des valeurs de chaîne explicites. Exemple de chaîne magique : {{=request_fname;/chris/}}. Pour plus d'informations, consultez la section Chaînes et dates magiques de la rubrique *Utilisation de CA Service Virtualization.*

Match tolerance (Tolérance de correspondance)

La tolérance de correspondance est un paramètre qui permet de contrôler la méthode utilisée par CA Service Virtualization pour comparer une demande entrante avec les demandes dans une image de service. Les options disponibles sont EXACT, SIGNATURE et OPERATION. Pour plus d'informations, consultez la section Tolérance de correspondance de la rubrique *Utilisation de CA Service Virtualization*.

Metrics (Mesures)

Les mesures permettent d'appliquer des méthodes et des mesures quantitatives aux performances et aux aspects fonctionnels de vos tests, ainsi qu'au système testé. Pour plus d'informations, consultez la section Génération de mesures (page 265) de la rubrique *Utilisation de CA Application Test*.

Model Archive (MAR) Info (Fichier d'informations d'archive de modèle (MAR))

Un *Model Archive (MAR) Info (Fichier d'informations d'archive de modèle (MAR))* est un fichier qui contient des informations requises pour la création d'une archive de modèle MAR. Pour plus d'informations, consultez la section Utilisation des archives de modèle (MAR) (page 277) de la rubrique *Utilisation de CA Application Test*.

navigation tolerance (tolérance de navigation)

La tolérance de navigation est un paramètre qui permet de contrôler le méthode utilisée par CA Service Virtualization pour rechercher la transaction suivante dans une arborescence des conversations. Les options disponibles sont CLOSE, WIDE et LOOSE. Pour plus d'informations, consultez la section Tolérance de navigation de la rubrique *Utilisation de CA Service Virtualization*.

Network graph (Graphique du réseau)

Le graphique de réseau est une zone de la console de serveur qui contient une représentation graphique du composant DevTest Cloud Manager et des laboratoires associés. Pour plus d'informations, consultez la section Démarrage d'un laboratoire (page 378) de la rubrique *Utilisation de CA Application Test.*

Node (Noeud)

Une étape de test interne à DevTest peut également être appelée *node* (noeud), ce qui explique l'intégration du terme node dans l'ID de certains événements.

Path (Chemin)

Un *chemin* contient des informations sur une transaction capturée par l'agent Java. Pour plus d'informations, reportez-vous à la rubrique *Utilisation de CA Continuous Application Insight*.

Path graph (Graphique de chemin)

Un *graphique de chemin* contient une représentation graphique d'un chemin et ses trames. Pour plus d'informations, consultez la section Graphique de chemin de la rubrique *Utilisation de CA Continuous Application Insight*.

Project (Projet)

Un *projet* est une collection de fichiers DevTest liés. Les fichiers peuvent inclure des scénarios de test, des suites, des modèles de service virtuel, des images de service, des configurations, des documents d'audit, des documents de simulation, des ensembles de données, des moniteurs et des fichiers d'informations MAR. Pour plus d'informations, consultez la section Panneau Project (Projet) (page 50) de la rubrique *Utilisation de CA Application Test*.

Property (Propriété)

Une *propriété* est une paire clé-valeur que vous pouvez utiliser comme variable d'exécution. Les propriétés peuvent stocker plusieurs types de données différents. Exemple de propriétés communes : LISA_HOME, LISA_PROJ_ROOT et LISA_PROJ_NAME. Une configuration est une collection nommée de propriétés. Pour plus d'informations, consultez la section Propriétés (page 93) de la rubrique *Utilisation de CA Application Test*.

quick test (test rapide)

La fonctionnalité *Quick test (Test rapide)* permet d'exécuter un scénario de test avec une installation minimale. Pour plus d'informations, consultez la section Simulation d'un test rapide (page 306) du *Utilisation de CA Application Test.*

Registry (Registre)

Le *registre* fournit un emplacement central pour l'enregistrement de tous les composants de DevTest Server et de DevTest Workstation. Pour plus d'informations, consultez la section Registre (page 11) de la rubrique *Utilisation de CA Application Test*.

ressource

Un *actif* est un ensemble de propriétés de configuration groupées dans une unité logique. Pour plus d'informations, consultez la section Actifs (page 115) de la rubrique *Utilisation de CA Application Test*.

service image (image de service)

Une *image de service* est une version normalisée de transactions enregistrées dans CA Service Virtualization. Chaque transaction peut être avec état (conversationnel) ou sans état. Une image de service peut être créée à l'aide de l'enregistreur d'image de service virtuel. Les images de service sont stockées dans un projet. Une image de service est également appelée *virtual service image (image de service virtuel, VSI)*. Pour plus d'informations, consultez la section Images de service de la rubrique *Utilisation de CA Service Virtualization*.

Simulator (Simulateur)

Un *simulateur* exécute les tests sous la surveillance du serveur de coordination. Pour plus d'informations, consultez la section Simulator Server (Serveur de simulation) (page 15) de la rubrique *Utilisation de CA Application Test*.

Staging document (Document de simulation)

Un *document de simulation* contient les informations sur l'exécution d'un scénario de test. Pour plus d'informations, consultez la section Génération de documents de simulation (page 235) dans la rubrique *Utilisation de CA Application Test.*

Subprocess (Sous-processus)

Un *sous-processus* est un scénario de test appelé par un autre scénario de test. Pour plus d'informations, consultez la section Génération de sous-processus de la rubrique *Utilisation de CA Application Test.*

Tableau de bord Continuous Service Validation (CVS) (Service de validation en continu)

Le *Continuous Service Validation (CVS) Dashboard (Tableau de bord du service de validation en continu)* permet de planifier l'exécution régulière de scénarios de test et des suites de test, sur une période étendue. Pour plus d'informations, consultez la section Service de validation en continu (page 385) dans la rubrique *Utilisation de CA Application Test.*

test case (scénario de test)

Un *scénario de test* est une spécification de la procédure de test d'un composant métier dans le système testé. Chaque scénario de test contient une ou plusieurs étapes de test. Pour plus d'informations, consultez la section Génération de scénarios de test (page 79) dans la rubrique *Utilisation de CA Application Test.*

test step (étape de test)

Une *étape de test* est un élément du flux de travaux de scénario de test qui représente une action de test unique à réaliser. Exemples d'étapes : Services Web, JavaBeans, JDBC et messagerie JMS. Une étape de test peut contenir des éléments de DevTest, tels que des filtres, des assertions et des ensembles de données liés. Pour plus d'informations, consultez la section Génération d'étapes de test (page 206) de la rubrique *Utilisation de CA Application Test.*

Test suite (Suite de tests)

Une *suite de tests* est un groupe de scénarios de test, d'autres suites de tests, ou les deux, planifiés pour être exécutés l'un après l'autre. Un document de suite de tests spécifie le contenu de la suite, les rapports à générer et les mesures à collecter. Pour plus d'informations, consultez la section Génération de suites de tests (page 265) de la rubrique *Utilisation de CA Application Test.*

Think time (Délai de réflexion)

Le temps de réflexion est la durée d'attente d'un scénario de test patiente avant d'exécuter une étape de test. Pour plus d'informations, consultez les sections Exemple d'ajout d'étape de test (page 208) et Editeur de documents de simulation - Onglet (page 237) Base de la rubrique *Utilisation de CA Application Test.*

transaction frame (Trame de transaction)

Une *trame de transaction* contient des données sur un appel de méthode intercepté par l'agent Java de DevTest ou un agent Light de CAI. Pour plus d'informations, consultez la section Transactions organisationnelles et trames de transaction de la rubrique *Utilisation de CA Continuous Application Insight.*

Virtual Service Environment (Environnement de service virtuel, VSE)

Le *Virtual Service Environment* (Environnement de service virtuel, VSE) est une application de DevTest Server qui permet de déployer et d'exécuter des modèles de service virtuel. VSE est également appelé CA Service Virtualization. Pour plus d'informations, reportez-vous à la rubrique *Utilisation de CA Service Virtualization*.

virtual service model (VSM, modèle de service virtuel)

Un *modèle de service virtuel* reçoit des demandes de service auxquelles il répond en l'absence du fournisseur de services réel. Pour plus d'informations, consultez la section Modèle de service virtuel de la rubrique *Utilisation de CA Service Virtualization*.