

CA Plex

Release Notes

r7.1



This Documentation, which includes embedded help systems and electronically distributed materials, (hereinafter referred to as the "Documentation") is for your informational purposes only and is subject to change or withdrawal by CA at any time. This Documentation is proprietary information of CA and may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA.

If you are a licensed user of the software product(s) addressed in the Documentation, you may print or otherwise make available a reasonable number of copies of the Documentation for internal use by you and your employees in connection with that software, provided that all CA copyright notices and legends are affixed to each reproduced copy.

The right to print or otherwise make available copies of the Documentation is limited to the period during which the applicable license for such software remains in full force and effect. Should the license terminate for any reason, it is your responsibility to certify in writing to CA that all copies and partial copies of the Documentation have been returned to CA or destroyed.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENTATION "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENTATION, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE.

The use of any software product referenced in the Documentation is governed by the applicable license agreement and such license agreement is not modified in any way by the terms of this notice.

The manufacturer of this Documentation is CA.

Provided with "Restricted Rights." Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7014(b)(3), as applicable, or their successors.

Copyright © 2014 CA. All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At <http://ca.com/support>, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at <http://ca.com/docs>.

Documentation Changes

The following documentation updates have been made since the last release of this documentation:

- [Generic .NET Runtime Bootstrap Executable](#) (see page 11)—Added this section to update the .NET runtime bootstrap executable feature information.
- [WinC/WinNTC DLL File Version Marking](#) (see page 12)—Added this section to update the WinC and WinNTC functions information.
- [.NET Client Localization Support](#) (see page 12)—Added this section to update the .NET client localization support feature information.
- [C++ Client Testing Tool Support](#) (see page 13)—Added this section to update the C++ client testing tool support feature information.
- [C# Client Testing Tool Support \(UID\)](#) (see page 13)—Added this section to update the C# client testing tool support feature information.
- [Additional Keycode Support for GUI Panels](#) (see page 13)—Added this section to update the additional keycode support for the GUI panels feature information.
- [.NET Client ActiveX Control Support](#) (see page 14)—Added this section to update the .NET client ActiveX support feature information.
- [C# Code Library Executable Support](#) (see page 15)—Added this section to update the .NET client ClickOnce deployment support feature information.
- [Features Removed from CA Plex r7.1](#) (see page 15)—Updated this section with the removed features in this release information.
- [Upgrading from Earlier Releases](#) (see page 17)—Updated this section with the upgrade information.
- [Before You Begin](#) (see page 17)—Updated this section with the upgrade information.
- [Group and Local Model Upgrade](#) (see page 17)—Updated this section with the upgrade information.
- [Upgrading from CA Plex r6.0, r6.1, and r7.0](#) (see page 18)—Updated this section with the upgrade information.
- [Windows C++ Functions Upgrade Requirements](#) (see page 18)—Updated this section with the upgrade information.
- [Java, C#, and RPG Function Upgrade Requirements](#) (see page 19)—Updated this section with the upgrade information.
- [Version Property](#) (see page 19)—Updated this section with the upgrade information.
- [Upgrading from CA Plex r5.5 SP1](#) (see page 21)—Updated this section with the upgrade information.

- [Upgrading from CA Plex r5.5](#) (see page 22)—Updated this section with the upgrade information.
- [Java and C# Exec SQL Action Diagram Source Code](#) (see page 22)—Updated this section with the upgrade information.
- [Java Source Code When Upgrading from CA Plex r5.5](#) (see page 23)—Updated this section with the upgrade information.
- [Windows Application Server Environment Settings](#) (see page 24)—Updated this section with the upgrade information.
- [RPG Functions Upgrade Requirements](#) (see page 24)—Updated this section with the upgrade information.
- [Build \(.BLD\) File Compatibility](#) (see page 24)—Updated this section with the upgrade information.
- [Windows C++ Client \(WinC Generator\)](#) (see page 42)—Updated this section with the upgrade information.
- [PC Development Environment](#) (see page 41)—Updated this section with the platform support information.
- [Windows C++ Server \(WinNTC Generator\)](#) (see page 44)—Updated this section with the platform support information.
- [Access Bookshelf from Plex IDE](#) (see page 15)—Added this section to update the bookshelf access from the Plex IDE information.

Contents

Chapter 1: New and Updated Features 11

Generic .NET Runtime Bootstrap Executable	11
WinC/WinNTC DLL File Version Marking	12
.NET Client Localization Support	12
C++ Client Testing Tool Support	13
C# Client Testing Tool Support (UID)	13
Additional Keycode Support for GUI Panels	13
.NET Client ActiveX Control Support	14
C# Code Library Executable Support	15
Access Bookshelf from Plex IDE	15
Features Removed from CA Plex r7.1	15

Chapter 2: Upgrading from Earlier Releases 17

Before You Begin	17
Group and Local Model Upgrade	17
Upgrading from CA Plex r6.0, r6.1, and r7.0	18
Windows C++ Functions Upgrade Requirements	18
New Runtime .INI Search Path	18
Java, C# and RPG Function Upgrade Requirements	19
.NET Server Runtime Upgrade Considerations	19
Upgrading from CA Plex r5.5	20
Windows C++ Functions Upgrade Requirements	20
COM Import Upgrade Considerations	21
Java Functions Upgrade Requirements	21
Windows Application Server Environment Settings	24
RPG Functions Upgrade Requirements	24
Long File Names Not Truncated By Default	24
Upgrading from CA Plex r5.1	25
Inheritance Resolution Changes in CA Plex r5.5	25
Case 1: Multiple Inheritance Call Resolution	25
Case 2: Changes to Sequence of Parameters	29
Case 3: Changes to the Sequence of Events, Subroutines and Collection Points	30
Upgrading from CA Plex r5.0	30
MFC Native Controls in Windows Clients	31
Calling Java Functions	33
Upgrading from CA Plex r4.5	34

Inheritance and Property Resolution Changes	34
Replacing the Target of an Inheritance Triple	34
Triples for the Same Property That Have Been Entered at More Than One Level.....	34
Meta-Variables in RPG Internal Functions and OBASE/Set Current Date and Time	35
Use of Single Quote Character in Java Messages and Source Code.....	35
Use of Backslash Character in Values for Java	36
Upgrading from CA Plex r4.0	36
Replacement and Scoped Objects.....	36
Upgrading from CA Plex r3.5	36
Change of Behavior with For Update Option	36
Change in CONCAT Operator in C++ Code	37
Avoiding Run-Time Level Checks When Accessing the System i Field Values File	37
Dynamic Application Partitioning APIs.....	38
Linker Options in Upgraded Local Models	38
Loading the Run-Time Property DLL	38
User Data (in BSUPPORT) and Business Contacts (in BCONTACT) Patterns	39
Upgrading from CA Plex r3.1 and r3.0.....	39
Upgrading from CA Plex 2.51 and Earlier	39

Chapter 3: System Requirements **41**

PC Development Environment	41
Platform-Specific System Requirements	41
Windows C++ Client (WinC Generator)	42
System i (System i 5250 and System i Client/Server Generators)	42
C# Server Generator	43
C# Client Generator	43
Java Generator	43
EJB Option	44
Windows C++ Server (WinNTC Generator)	44
Deployment System Requirements.....	44

Chapter 4: Installation Considerations **45**

Microsoft Help Compiler Installation	45
--	----

Chapter 5: Published Fixes **47**

Chapter 6: Known Issues **49**

Modification of Internal Tables	49
C++ Build Performance with Visual Studio 2005	49

Chapter 7: International Support	51
Chapter 8: Documentation	53

Chapter 1: New and Updated Features

This chapter documents the new and updated features for CA Plex r7.1.

This section contains the following topics:

[Generic .NET Runtime Bootstrap Executable](#) (see page 11)

[WinC/WinNTC DLL File Version Marking](#) (see page 12)

[.NET Client Localization Support](#) (see page 12)

[C++ Client Testing Tool Support](#) (see page 13)

[C# Client Testing Tool Support \(UID\)](#) (see page 13)

[Additional Keycode Support for GUI Panels](#) (see page 13)

[.NET Client ActiveX Control Support](#) (see page 14)

[C# Code Library Executable Support](#) (see page 15)

[Access Bookshelf from Plex IDE](#) (see page 15)

[Features Removed from CA Plex r7.1](#) (see page 15)

Generic .NET Runtime Bootstrap Executable

CA Plex r7.1 provides the ability to call any CA Plex generated C# function through the command line using the .NET runtime bootstrap executable called PlexRuntimeLauncher.exe. You can use this executable to call both .NET Client and Server functions.

The following documentation deliverables were updated for this feature:

- .NET Platform Guide—*Calling a .NET Application Without Creating an Executable* topic
- User Guide—*Running an Application* topic

WinC/WinNTC DLL File Version Marking

With CA Plex r7.1, for unmanaged C++ applications, version information is added to a compiled DLL by adding a VERSIONINFO structure into a resource file (.rc) associated with your function. This information is then compiled into the target DLL through the MS Resource Compiler when the function is build.

By adding the File Version information to your WinC and WinNTC functions, you can perform the following tasks:

1. Track changes within your applications.
2. Create robust installation programs that work with the compiled Plex generated C++ artifacts.
3. Identify which CA Plex model function, a Plex generated DLL is related to.
4. Identify which version and PTF build of CA Plex was used to generate and build a particular DLL.

The following documentation deliverables were updated for this feature:

- User Guide—*Adding File Version Information to WinC and WinNTC Functions* topic

.NET Client Localization Support

CA Plex supports the ability to generate multilingual user interface (UI) applications for WinC, Java, and .NET based applications. For .NET Client applications, this feature has been enhanced to include the generation and building of resource DLLs that can be loaded at runtime to customize your .NET Client application UI.

The following documentation deliverables were updated for this feature:

- .NET Platform Guide
 - *Supporting Multilingual User Interface with .NET Application* topic
 - *Default Assembly and Resource Dictionary Assembly* topic
 - *Build Resource Dictionary Assembly* topic
 - *Specifying the Resource Dictionary Assembly* topic
- Dialog and Window Help
 - *Generate and Build Options* topic

C++ Client Testing Tool Support

With CA Plex r7.1, the Control IDs allocated for GUI panel elements generated onto a panel can now be stored in the panel design for each panel. This means that external testing tools can rely on the fact that the control IDs allocated against C++ resource files always remain to be the same for the same controls. This feature makes creating test scripts against WinC applications much more robust against changes to label text or panel element reordering.

The following documentation deliverables were updated for this feature:

- User Guide—*Allocating Fixed Control IDs Against User Interface Elements* topic

C# Client Testing Tool Support (UID)

With CA Plex r7.1, the Control IDs allocated for GUI panel elements generated onto a panel can now be stored in the panel design for each panel. This means that QA automation tool such as HP QTP can work on the Plex generated .NET Client functions and also can run the recorded project/script repeatedly using the Unique Identifier (UID) property against controls.

The following documentation deliverables were updated for this feature:

- .NET Platform Guide—*Generating .NET QA Automation Tool Support UID (Unique Identifier)* topic
- User Guide—*Allocating Fixed Control IDs Against User Interface Elements* topic

Additional Keycode Support for GUI Panels

As of CA Plex r7.1, you can select from an extended set of keys to use as shortcuts in your GUI applications. This feature includes keys such as the numeric keypad and Spacebar keys. Refer the table in the User Guide for a full list of the keys that your application can use with a list of exceptions.

The following documentation deliverables were updated for this feature:

- User Guide—*Providing Keyboard Support* topic

.NET Client ActiveX Control Support

The GUI Panels generated as C# functions that target Windows Presentation Foundation (WPF) now have the ability to host the ActiveX controls. This feature provides you with the ability to support the same ActiveX GUI controls that you use in your Windows C++ Client (WinC) applications.

Since the VBScript engine cannot be hosted in the .NET runtime, migrate any VBScript code that you currently use to access your ActiveX controls, to use C# source. Currently, the ActiveX controls imported through the CA Plex COM import feature are not supported.

The following documentation deliverables were updated for this feature:

- .NET Platform Guide
 - *.NET Client ActiveX Control Support* topic
 - *.NET Client Panels and ActiveX Controls* topic
 - *.NET Client Functions and ActiveX Controls* topic
 - *Using Properties of an ActiveX Control Through C# Source* topic
 - *Calling ActiveX Control Methods Through C# Source* topic
 - *ActiveX Control Event Handling Through C# Source* topic
 - *Additional Considerations for Writing C# Source for ActiveX Controls* topic
 - *Generating and Building C# Client Functions with ActiveX Controls* topic
 - *Deploying C# Client Applications with ActiveX Controls* topic


C# Code Library Executable Support

As of CA Plex r7.1, you can select an entry function for a Code Library. When you compile a C# Code Library that contains an entry function, an executable is created for it, which calls the specified function. You can now model your application deployments completely through Code Libraries without having to resort to creating executable through the Generate and Build window.

The following documentation deliverables were updated for this feature:

- User Guide
 - *Specifying an Entry Function for a Code Library* topic
- .NET Platform Guide
 - *Calling a .NET Application from an Executable* topic
 - *Creating an Executable for a .NET Client Application from the Generate and Build Window* topic
 - *Calling a .NET Application Without Creating an Executable* topic
 - *Plex Executable Command-Line Parameters* topic

Access Bookshelf from Plex IDE

With CA Plex r7.1, you can launch the End-to-End (E2E) bookshelf by clicking the  icon on the Plex IDE.

Features Removed from CA Plex r7.1

No features have been removed from this release.

Chapter 2: Upgrading from Earlier Releases

This chapter provides an overview of upgrade requirements for upgrading from previous releases of CA Plex.

This section contains the following topics:

[Before You Begin](#) (see page 17)

[Upgrading from CA Plex r6.0, r6.1, and r7.0](#) (see page 18)

[Upgrading from CA Plex r5.5](#) (see page 20)

[Upgrading from CA Plex r5.1](#) (see page 25)

[Upgrading from CA Plex r5.0](#) (see page 30)

[Upgrading from CA Plex r4.5](#) (see page 34)

[Upgrading from CA Plex r4.0](#) (see page 36)

[Upgrading from CA Plex r3.5](#) (see page 36)

[Upgrading from CA Plex r3.1 and r3.0](#) (see page 39)

[Upgrading from CA Plex 2.51 and Earlier](#) (see page 39)

Before You Begin

Before upgrading, check the CA Plex home page at CA Technical Support (<http://ca.com/support>) for additional information or fixes that may have been made available after this guide was published.

Multiple releases of CA Plex and its generated applications can be installed on the same machine. For example, CA Plex r7.1, r7.0, and r6.1 can be installed and run on the same machine.

Group and Local Model Upgrade

It is a good practice to update local models to the group model regularly. Therefore, we recommend that you update the group model and make offline backups of all group and local models before you start the upgrade. Then create new local models (and build files) after upgrading the group model to the new release.

All group models and associated library models must be upgraded at the same time.

Logging in to a group model causes it to be upgraded to CA Plex r7.1. Thereafter, once you save the group model, you will not be able to access the model with a previous release. For this reason, it is necessary for all developers in your work group to upgrade to the new release of CA Plex at the same time. Two developers cannot work on the same model simultaneously if they are using different releases of CA Plex.

Local Model Upgrade

It is possible to upgrade a local model from an earlier release simply by opening it with the later release. This is often useful for testing purposes, however as described above this is not the recommended approach for a formal release upgrade.

Upgrading from CA Plex r6.0, r6.1, and r7.0

This section explains upgrade requirements related to upgrading from r6.0, r6.1, and r7.0.

Windows C++ Functions Upgrade Requirements

When upgrading from CA Plex r6.0, r6.1, or r7.0 to CA Plex r7.1 you do *not* need to regenerate or recompile existing C++ functions.

New Runtime .INI Search Path

To better support Windows Vista security requirements and multi-user deployments, the algorithm that the WinC runtime uses to locate and create runtime .INI files has been changed.

Review these changes carefully, especially, if you have developed custom .INI file handling or deploy CA Plex applications in server-based multi-user environments.

Typical Example

In a typical end-user environment, your applications are installed in the Program Files folder including an .ini file. The files in Program Files are read-only for standard users.

When the Plex WinC application starts, it searches for a *writable* .INI file. If the file is not found, a new .INI file is created in the end-user's personal folder (My Documents on Windows XP, or Documents on Windows Vista). The .INI file is located in a sub-folder with the same name as the executable. For example, MyApp.exe would have the .INI file "My Documents\Myapp\Myapp.ini"

Therefore, in this example, each standard user who runs the application will have their own copy of the application's .INI file.

.INI File Algorithm

The full set of rules is set out below and provides flexibility for other deployment scenarios dependent on the access rights of the user who runs the application:

If a Plex generated executable is called app.exe, then the respective .ini file will be app.ini.

At run time, an application app.exe searches for an app.ini file as follows:

1. In the same directory where the executable (app.exe) is located. If app.ini file is found there and has read-write (RW) access then the executable app.exe uses it.
2. Otherwise, the app.exe searches in the local user \Documents\app\ directory (where app is the name of the executable). If the file app.ini is found there and has RW access, then the executable app.exe uses it.
3. Otherwise, the executable uses the PATH statement to locate an app.ini file that has RW access.
4. If such an app.ini file is still not found, then the executable app.exe creates a new app.ini file inside the local user \Documents\app directory with RW access. This new app.ini will be a copy of the first read-only app.ini found, or, If not found, a blank app.ini is created.

Java, C# and RPG Function Upgrade Requirements

When upgrading to CA Plex r7.1, you do not need to regenerate or rebuild existing Java, C# or RPG functions.

.NET Server Runtime Upgrade Considerations

The following sections explain the NET Server Runtime upgrade considerations.

Version Property

Unless you have a specific technical reason, the Version property for the .NET listener must be left blank. The value of 600 or 610 must be deleted if it is present.

At CA Plex r7.1, if this property is empty, the default version number from the Plex .NET runtime is taken.

The Version property is provided for backwards compatibility reasons. For example, to use r6.0 WinC clients to connect to the r7.1 version of .NET runtime, you must set this property to 600.

Upgrading from CA Plex r5.5

This section explains upgrade requirements related to upgrading from r5.5 (including r5.5 SP1).

Windows C++ Functions Upgrade Requirements

Windows functions (WinC and WinNTC) created with CA Plex releases earlier than r6.0 must be recompiled to be compatible with the CA Plex r7.1 C++ run-time system.

A single Windows application cannot contain DLLs created with different releases of CA Plex (except in the case of r6.0, r6.1, r7.0, and r7.1). CA Plex includes a run-time version check whenever a generated DLL calls another. A run-time error occurs if you attempt to make a call between DLLs created with incompatible releases. In your development environment, you need to rebuild all of your application DLLs when you move from CA Plex r5.5 or earlier up to CA Plex r7.1.

For more information about run-time applications, see the *Running Applications Created with Different Versions of CA Plex* in the online help.

For more information about the new features in CA Plex r7.1, see the *New Features* chapter in this guide.

COM Import Upgrade Considerations

If you have used the COM Component Import feature, CA recommends that you reimport and regenerate the imported COM packages as part of the release upgrade process. This is due to fixes and improvements to the COM import and wrapper generation processes at this release. You may need to revise the wrapper attributes of the package before it compiles. Note the following:

- The COM Component Import wizard provides an option *Do not overwrite existing package*. However, clearing this option is not appropriate for upgrading previously imported COM packages.
- An alternative is to delete the COM package object before running COM Component Import. However, this will invalidate existing action diagram statements that reference the COM package.

To avoid these limitations, the following upgrade procedure is recommended:

1. Create a new group model and attach the COMPONENT library.
2. Extract a new local model and use the COM Component Wizard to import a new COM package for component you need to upgrade.
3. In Object Browser, right-click the COM package and select XML Import from the Tools menu. Export the package to a named XML file.
4. Open a local model containing the COM package that you need to upgrade.
5. From the Tools menu, select Import and then XML Import. Select the Clear option and then import the XML file you created earlier.

The above procedure will upgrade the COM package while preserving existing action diagram code.

Java Functions Upgrade Requirements

This section discusses the upgrade requirements for Java functions.

Upgrading from CA Plex r5.5 SP1

You do not need to regenerate existing Java functions if you are upgrading from r5.5 SP1 (Build 5.5.93).

The CA Plex r7.1 Java run time (obrun.jar) is backwards compatible with earlier releases. You can use the new runtime with functions created at r5.5 and previous releases. This enables you to take advantage of fixes and improvements in the new runtime without necessarily upgrading to the full CA Plex r7.1.

Upgrading from CA Plex r5.5

If you regenerate a Java function at CA Plex r7.1, then you must also regenerate all other 5.5-generated functions in the call graph of that Function. This is due to changes in the parameter formats at CA Plex r5.5 SP1 and later. We recommend a full regeneration and recompilation of all Java functions when upgrading from CA Plex r5.5 (Build 5.5.63) or previous releases. This reduces the number of generated classes and identifies any source code objects that need to be modified for compatibility with the new format of generated Java code.

Note: This Java regeneration and rebuild requirement is only required when upgrading from CA Plex r5.5 and earlier. CA does not anticipate that future releases of the Java generator will have this requirement. No such requirement exists if you are upgrading from r5.5 SP1.

If you want to use the CA Plex r7.1 Java run time (obrun.jar) with a CA Plex r5.5 Java application, you need to have these settings in your ob600xxx.properties file:

```
Version=710
```

```
SPVersion=0
```

Java and C# Exec SQL Action Diagram Source Code

In CA Plex r7.1, r7.0, r6.1, and r6.0, the Java and C# generators generate all Exec SQL statements as prepared statements by default. The SRC SQL statement type SYS triple can be used to indicate source code objects that must not be generated with prepared statements.

Generating prepared statements may improve performance and scalability in cases where the same SQL statement is executed repeatedly. However, the results may also depend on application and DBMS-specific factors. CA recommends benchmarking with different values for SRC SQL statement type SYS for determining the optimum setting in each case.

Also, generating prepared statements assumes that all parameters to the Exec SQL statement correspond to columns in a table. This assumption is not always true because the Exec SQL statement can be used to implement a wide variety of SQL code. The SRC SQL statement type SYS triple can be used to indicate source code objects that should not be generated with prepared statements. When upgrading Java applications from r5.5 or earlier, it may be necessary to add such triples to some source code objects to avoid errors.

This change does not affect Exec SQL usage with C++.

Java and Oracle varchar null support

Java applications that target the Oracle database and use optional varchar fields may behave differently after upgrading from r5.5 or earlier. Depending on how the varchar is modeled in Plex, unexpected optimistic locking errors may occur after upgrade. For example:

```
"Instance of .... changed by another user"
```

The typical solution is to add FLD null VAL triples to the varchar field where the target value is empty. For more information, search the CA Support Online web site for problem CPLEX 1296.

Java Source Code When Upgrading from CA Plex r5.5

The Java source code you entered into source code objects in your model may need to be modified. This is because of the changes in the structure of the Java classes created by the generator at CA Plex r5.5 SP1. The source code objects supplied in CA's pattern libraries (such as the JAVA API library) are already modified. Re-extract from the shipped JAVA API library before regenerating and compiling with CA Plex r7.1.

Problems in Java source code result in an unexpected type error at compile time. For example:

```
C:\GENJAVA\MyFunction_ObFnc.java:line number: unexpected type
```

```
required: variable
```

```
found    : value
```

To fix such problems use the assign method instead of the = operator.

Prior to r5.5 SP1, you could use the operator = to assign the return value of a method to an output parameter of your source code API. For example:

```
&(4:) = anyMethod(&(1:), &(2:), &(3:));
```

From r5.5 SP1 onwards, the code in the previous example needs to be modified to include the assign method instead of the = operator, as follows:

```
&(4:).assign(anyMethod(&(1:), &(2:), &(3:)));
```

Windows Application Server Environment Settings

The registry keys for the CA Plex Windows Application Server environment settings have changed because of the rebranding from Advantage to AllFusion.

The CA Plex r7.1 installation program automatically copies pre-existing environment settings to the new key values. However, if you do not see the copied settings in the CA Plex Windows Application Environment Manager, execute the migration2.exe program (in the AppServer\Bin folder).

Note: No parameters are required to execute the migration program.

RPG Functions Upgrade Requirements

You do not need to regenerate or rebuild existing RPG functions when upgrading to CA Plex r7.1.

Build (.BLD) File Compatibility

As discussed earlier, it is recommended that you create new local models, and therefore new build files as part of the upgrade process. If you use an old BLD file with CA Plex r7.1, CA Plex sends an error message (E-BLD-1777) each time you open the Generate and Build window. If you want to continue using the old BLD, review your Generate and Build Options to ensure they are compatible with CA Plex r7.1.

Pre 7.0 build files will cause the Build Directories section of the Generate and Build Options to be displayed using the AS/400 brand name instead of System i.

To prevent the error message being displayed you can add the following entry to the BLD file:

```
[Options]  
ReLease=600
```

Long File Names Not Truncated By Default

The Name Allocation option called Truncate Long File Names at Generation is no longer switched on by default. This means that the file names longer than 8 characters will no longer be truncated unless you explicitly select this option. This may have an upgrade impact if your model contains names that were truncated at previous releases.

Upgrading from CA Plex r5.1

The following sections provides the upgrade requirements related to upgrading from r5.1

Inheritance Resolution Changes in CA Plex r5.5

At r5.0 of CA Plex, the inheritance engine was changed to address limitations and bugs in inheritance behavior. Additional changes were also made to support the new Dependency Browser in r5.0. For some customers, these changes caused significant problems when the time came to upgrade to the new release. CA Plex r5.1 included some fixes for these problems, but some significant problems remained.

CA Plex r5.5 resolves the known bugs in the inheritance engine. No changes were made to the inheritance engine between r5.5 and r5.5 SP1. There are differences in inheritance engine behavior compared to earlier releases that will impact some customers. In the following sections three different cases are described.

Case 1: Multiple Inheritance Call Resolution

In some cases inherited function calls may be resolved differently in the action diagram with the result that a different call is created in the generated code. In a typical model, this only affects a very small percentage of functions (if any). These cases can be identified through testing of the generated application, comparisons of generated source, or comparisons of action diagram code. If you require further assistance in identifying such problems, contact CA Technical Support.

You can resolve these problems by adding the appropriate replacement triples to return the function to its required behavior. Ideally such triples can be entered in your *standards layer* model, thus minimizing the amount of changes required.

Background

CA Plex r3.5 introduced support for multiple function inheritance by changing the behavior of the FNC is a FNC verb. As a result, even existing CA Plex functions could inherit from more than one function without keying any extra triples. Another consequence of this was to allow two or more different inherited Calls triples to be Visible in the Object Properties corresponding to the same original action diagram call. Only one call is appropriate in any inheriting action diagram and CA Plex determines which of the available functions should actually be called when it is loaded.

Prior to r5.0, CA Plex did not explicitly handle such action diagram calls. As a result, the resolution of the call did not always follow the usual rules of the inheritance engine. At r5.0 and later a mechanism was put in place to handle such calls. This mechanism has been refined a number of times, most recently at r5.5.

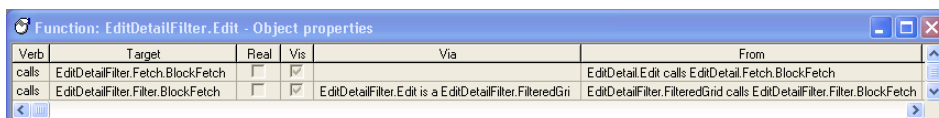
Example 1: The Filter Pattern

A common example of multiple inherited action diagram call can be seen in instances of the Filter.FilteredGrid function. Consider these triples:

```

EditDetailFilter is a ENT STORAGE/RelationalTable
EditDetailFilter is a ENT FOUNDATI/EditDetail
EditDetailFilter is a ENT FOUNDATI/Filter
EditDetailFilter.Edit replaces FNC UIBASIC/Grid
                        ...by FNC EditDetailFilter.FilteredGrid
    
```

When you look at the calls for the EditDetailFilter.Edit you will see calls to two different BlockFetch functions:



Verb	Target	Real	Vis	Via	From
calls	EditDetailFilter.Fetch.BlockFetch	<input type="checkbox"/>	<input checked="" type="checkbox"/>		EditDetail.Edit calls EditDetail.Fetch.BlockFetch
calls	EditDetailFilter.Filter.BlockFetch	<input type="checkbox"/>	<input checked="" type="checkbox"/>	EditDetailFilter.Edit is a EditDetailFilter.FilteredGrid	EditDetailFilter.FilteredGrid calls EditDetailFilter.Filter.BlockFetch

Note that both calls are visible (the Vis column is checked). You can see this same behavior in r5.0 and r5.1. So which BlockFetch is actually called in the action diagram? This determination is made at action diagram load time. The result is sensitive to changes in the inheritance engine in recent CA Plex releases. The intention is that the Filter.BlockFetch should be called. To get this result at r5.5, CA has entered an additional replacement triple on the FOUNDATION pattern library:

```
FOUNDATI/Filter.FilteredGrid replaces FNC UIBASIC/UIBasic.Grid.BlockFetch
...by FNC FOUNDATI/Filter.Filter.BlockFetch
```

Previously, the BlockFetch function was not explicitly replaced. Instead the replacement was made only on the scoping view:

```
FOUNDATI/Filter.FilteredGrid replaces VW UIBASIC/UIBasic.Grid
...by VW FOUNDATI/Filter.Filter
```

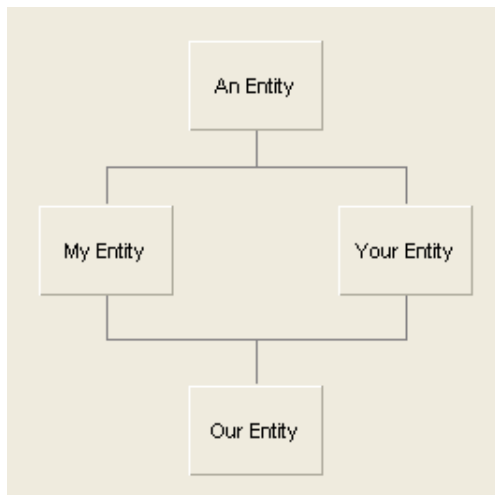
With the revised algorithm used in CA Plex r5.5, this was not sufficient to force the required function replacement to occur. By adding an additional, explicit replacement on the BlockFetch itself, CA has been able to retain the required behavior in r5.5. Similar actions may be necessary in your own functions that exhibit this behavior.

Example 2: Diamond Inheritance

Consider the following set of triples:



This type of scenario is known as *diamond inheritance* due to the shape of the resulting inheritance hierarchy as shown by the following diagram.



The inheritance hierarchy starts at *An Entity*, diverges, and then the two branches are brought back together at *Our Entity*. This situation is quite complex but the examples in real customer models are often even more complex with further branches and layers of inheritance involved.

If you view the Object Properties for the function *Our Entity.Our Function* then you can see the two calls triples, both visible. There are two important points to note:

- There is only one actual action diagram Call explicitly entered in this set of functions but two possible calls were resolved in Object Properties
- This scenario can be reproduced in any CA Plex release from 3.5 onwards

Object properties - Function: Our Entity.Our Function			
Verb	Target	Real	Vis
calls	Our Entity.Our Validation Function	<input type="checkbox"/>	<input checked="" type="checkbox"/>
replaces	Our Entity.A Validation Function	<input type="checkbox"/>	<input checked="" type="checkbox"/>
...by	Our Entity.Our Validation Function	<input type="checkbox"/>	<input checked="" type="checkbox"/>
is a	Our Entity.My Function	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
calls	Our Entity.Your Validation Function	<input type="checkbox"/>	<input checked="" type="checkbox"/>
is a	Our Entity.A Function	<input type="checkbox"/>	<input checked="" type="checkbox"/>
replaces	Our Entity.Our Validation Function	<input type="checkbox"/>	<input checked="" type="checkbox"/>
...by	Our Entity.Your Validation Function	<input type="checkbox"/>	<input checked="" type="checkbox"/>
is a	Our Entity.Your Function	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
replaces	Our Entity.My Validation Function	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
...by	Our Entity.Our Validation Function	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

The only way to determine which function is actually called is to examine the action diagram code. This example again shows the changes in the inheritance engine over recent releases: At r5.0, Our Entity.Your Validation Function is actually called, but in all other releases including r5.5 it is Our Entity.Our Validation.

Case 2: Changes to Sequence of Parameters

In some cases the sequence of inherited parameters on a generated function call may change compared to earlier releases. In many cases, such changes have no negative consequences since all functions concerned are regenerated in the course of upgrading to the new release. However, there are cases where such changes may be significant:

- If you are exploiting the backwards compatibility of existing RPG and Java functions, then this change may require that additional functions be regenerated
- If you have hand-coded programs that integrate with programs generated by CA Plex, then those hand-coded programs may be modified to take account of the revised parameter interface

These cases can be identified by testing the generated application, comparisons of generated source, or comparisons of action diagram code. If you require further assistance in identifying such problems, contact CA Technical Support.

Case 3: Changes to the Sequence of Events, Subroutines and Collection Points

The sequence of inherited Event constructs and Subroutines can be altered compared to previous releases. Changes to the sequence of Subroutines have no negative consequences for the generated application. In general, this is also true for Event constructs. However, the sequence of Events may be significant if you referenced the same logical event on multiple Event constructs or used an unqualified Event construct. For example:

Events Handler

Event

```
//unqualified event triggered for every event
```

```
Go Sub Generic Event Processing
```

Event Delete

```
Go Sub Delete Processing
```

Event Delete

```
Go Sub More Delete Processing
```

If the previous sequence of Event constructs is different, then the behavior of the generated application will change. In practice, this functionality is rarely exploited and it is even rarer for the inheritance engine changes to cause significant differences in the sequence. As in other cases, such problems can be identified through testing of the generated application, comparisons of generated source or comparisons of action diagram code.

In principle, similar considerations apply to the sequence of inherited code blocks in Collection Points (Pre and Post Points). Testing by CA has not revealed examples of such results but you should be aware that the possibility exists.

Upgrading from CA Plex r5.0

In addition to the upgrade requirements documented in the earlier sections, review the following sections when you are upgrading from r5.0.

MFC Native Controls in Windows Clients

In r5.0 and later, CA Plex supports two sets of GUI controls in Windows clients. By default, WinC applications use MFC controls for all types of controls, except the grid. The alternative set of GUI controls is called Winwidgets, which was the default before r5.0.

There are advantages and disadvantages associated with each set of controls. Winwidgets controls have been used in all releases of CA Plex since 1.0. Consequently, they represent a mature technology that has been implemented successfully for many years by many CA Plex customers. When upgrading from an earlier release of CA Plex, it is often simpler to use the Winwidgets controls since this is likely to minimize any backwards compatibility problems.

MFC controls provide a range of advantages including:

- Compatibility with third-party testing tools
- Windows standard look and feel
- Access to the MFC API for low-level control

Backwards Compatibility Options

With some exceptions (see the following section), all the functionality previously available with Winwidgets controls should also be available with the MFC controls. However, to safeguard against unexpected upgrade issues, run-time options are available that enable you to revert to the Winwidgets controls.

A separate option is available for each type of control:

```
[NativeControls]
;GUI controls either display as MFC controls (1) or Winwidgets (0).
;These settings have no effect on the WinC grid which is always Winwidgets
ListBox=1
SpinButton=1
SingleLineEdit=1
SingleLineEditNumeric=1
MultiLineEdit=1
ComboBox=1
RadioButton=1
CheckBox=1
PushButton=1
Statics=1
```

Set the required option to 0 if you want to revert the control concerned to Winwidgets. The MultilineEdit setting is an exception—Setting this value to 0 invokes an earlier implementation of the native control as Winwidgets do not provide a multiline edit control.

Note: Report any undocumented upgrade issues to CA Technical Support.

Upgrade Issues with MFC Controls

Known upgrade issues are listed below. Report any undocumented upgrade issues to CA Technical Support if they cause a problem in your application.

- **Disabled text color.** When MFC controls are disabled, the text displays in the Windows standard color (typically, light gray). The Text Color property is ignored when controls are disabled. Note that for edit controls you can use the combination of properties

`Mode=Read-only`

and

`Disabled=No`

to define a read-only edit control that does support the Text Color property.

- **Combo box size.** Unlike other control types, the size of the edit control portion of a combo box cannot be changed directly; it is determined by the font size of the text within the control. You can change the size of the edit portion at design time but it is ignored at run time. Instead, changing the size of the edit portion at design time, changes the size of the drop-down list at run time.
- **Transparency.** MFC controls do not fully support transparency. This may change the appearance of panels in cases where controls overlap.
- **z-order.** The z-order of MFC controls is reversed compared to the default z-order of Winwidgets controls. This may impact some panel designs that rely upon overlapping controls.

Note: To allow the z-order of design time controls to more closely match the run time, set the Clip Control property to Yes for each control concerned. This is useful when working with overlapping controls such as a frame within a frame.

Calling Java Functions

As of CA Plex r4.5, when calling Java functions from the command line or from hand-coded Java, the function name must be prefixed with its package (`package.func`).

Note that at CA Plex r5.0 Service Pack 1 and later, there is a `PackageList.properties` file entry that can be used to provide a list of packages that are searched if no package is specified within the call. Typically, a function's package is generated directly into the code for each function call. However, this may not be the case in a dynamic partitioning scenario. For example, at generation time the target of a function call could be a Windows function (in which case no package will be included in the generated call). At run time, if the target function is switched to Java then the `PackageList` can be used to locate the function.

Upgrading from CA Plex r4.5

This section explains upgrade requirements related to r4.5.

Inheritance and Property Resolution Changes

CA Plex r5.0 and later includes enhancements to the inheritance engine that may change the properties of inherited objects in some circumstances.

Replacing the Target of an Inheritance Triple

Consider the following example:

A is a B

B is a C

A replaces C by D

When **B is a C** arrives on A, A no longer inherits from C, as B did, but it now inherits from D.

In previous releases, anything inherited from D, which is in contention with that inherited from B, defers to the version inherited from B. In r5.0, anything contentious inherited from D takes precedence over things from B which arose from its inheritance from C. In other words **A is a D** takes precedence over **A is a B** and **B is a C**. This result is now consistent with the general rules of inheritance, where *later* triples take precedence over *earlier* triples.

Triples for the Same Property That Have Been Entered at More Than One Level

It is possible to duplicate triples for an object by entering a triple, changing configuration to an earlier level and entering the same property. For example:

A **type** Character (Level 3)

A **type** Numeric (Level 1)

With the configuration set to the later level (Level 3 in the example) both triples would show in the Model Editor prior to r5.0 and are passed on to the inheritance engine. The last triple in sequence (as seen in the Model Editor) takes precedence. So in the previous example, A would be Numeric.

In CA Plex r5.0, the *later hides earlier* principle is applied to the levels and only **A type** Character is seen in the Model Editor and passed on for inheritance.

Meta-Variables in RPG Internal Functions and OBASE/Set Current Date and Time

At r5.0, a fix was added so that internal RPG functions now have their own meta-variable space, and no longer share the meta-variable space of their calling functions. This change is in accordance with the published functionality regarding the scope of meta-variables (which states that a meta-variable's state only persists on calls to functions of type Meta) and is consistent with other generators.

This fix highlighted a place within the class libraries where meta-variable state information was expected to pass from an external function to an internal function. It was calling the OBASE/Set current date and time function in the OBASE variants AS400 5250 and Windows/AS400.

Note: OBASE/Server Set current date and time is a new function used with RPG400. It is recommended that you call this function instead of the OBASE/Set current date and time function with OBASE set in the Windows/AS400 variant and OBDATETIME set in the Windows client variant. This is exemplified in the call change in OBASE/Audited entity.Set audit fields.

To assist customers in tackling any upgrade issues associated with this fix, the following Plex.ini file option can be used in CA Plex r5.1:

```
[RPG Generator]
Share Meta-Variables With Internal Functions=1
```

An entry of 1 reverts to pre-r5.0 behavior. If no entry is present, the default setting of 0 is used, providing the same behavior as r5.0.

Use of Single Quote Character in Java Messages and Source Code

The Java Generator at CA Plex r4.5 and earlier required two single quote characters (") to be used in messages and source code and required a single quote to be used at run time. For example, this was required when embedding parameters in the source code for use with the Exec SQL statement. This behavior was inconsistent with other generators.

This problem was corrected in CA Plex r5.0. Now only a single quote character (') is required. However, any existing messages or source code that used the previous workaround need to be edited when upgrading to r5.0.

Use of Backslash Character in Values for Java

The Java Generator at CA Plex r4.5 and earlier required that two backslash characters (\\) be used in values whenever you required a single backslash character to be used at run time. This behavior was inconsistent with other generators.

This problem was corrected in CA Plex r5.0. Now only a single backslash character (\) is required. However, any existing values using the previous workaround will need to be edited when upgrading to r5.0.

Upgrading from CA Plex r4.0

This section explains upgrade requirements related to r4.0.

Replacement and Scoped Objects

CA Plex r4.5 introduced a change to the resolution of replacement triples attached to scoping objects to provide better control over replacement and performance improvements. This fix is not enabled by default. It requires a setting to be added to the Plex.ini file because it can cause significant backwards compatibility issues when it is enabled. For a full discussion, search the online help for the topic Replacement and scoped objects.

Upgrading from CA Plex r3.5

In addition to the instructions in Upgrading from CA Plex r4.0, review the following sections.

Change of Behavior with For Update Option

In r4.0, the behavior of the For Update option on Position and Fetch action diagram statements was changed to support pessimistic concurrency. It is possible that these changes may significantly change the behavior of ODBC-based applications (including NT BackOffice applications). For more information about backwards compatibility options, see the topic Row Locking in SQL Implementations in the online help.

Change in CONCAT Operator in C++ Code

The documented behavior of the CONCAT operator is that trailing blanks are removed from field values. In practice, it was possible to retain trailing blanks in C++ functions in cases where the values concerned were not displayed on panels. This inconsistency was corrected in CA Plex r4.0 in that trailing blanks are now removed in all cases. The recommended technique for including blanks in concatenated strings is to use the Format Message statement with the blanks embedded in the message.

Avoiding Run-Time Level Checks When Accessing the System i Field Values File

The YOBDDL program resides within the CA Plex library on the System i. When building the System i Field Values File within CA Plex, a copy of the YOBDDL program is created in the System i Object Library. This copy is called YOBVALSV. A copy is not made if YOBVALSV already exists.

The YOBDDL program was modified for r4.0 of CA Plex. For this reason, the old versions of YOBVALSV need to be deleted from each of your System i Object Libraries. Enter the following command on the System i:

```
DLTPGM PGM(Object-Library/YOBVALSV)
```

The new version of YOBVALSV needs to be placed in to each of the System i Object Libraries. This can be done by either:

- Rebuilding the System i Field Values File
- Creating a duplicate object by entering the following command on the System i:

```
CRTDUPOBJ OBJ(YOBDDL) FROMLIB(PLEX)  
OBJTYPE(*PGM)  
TOLIB(Object-Library) NEWOBJ(YOBVALSV)
```

Failure to perform these steps results in a level-check at run time when attempting to access the field values selection list.

Dynamic Application Partitioning APIs

If you have used the `GetLocationInformation` and `SetLocationInformation` APIs, note that the structure `ObLocationInfo` was changed for CA Plex r4.0. If you have created source code objects that implement these APIs, you must edit the code to avoid compile errors.

The following three fields are removed:

```
ObLongFld m_fDataConv
ObLongFld m_fObTran
ObCharFld m_ObTranDLL
```

and replaced with the following single field:

```
ObLongFld m_iDataConv
```

See the `Odap.mdl` sample model for examples of the required source code.

Linker Options in Upgraded Local Models

CA Plex r4.0 introduced a new feature called custom C++ build options. The linker option `/OPT:NOWIN98` is used by default. Without this option the size of compiled DLLs is significantly increased. If your local model was created before r4.0 of CA Plex, this option does not appear by default and should be added manually. For more information, see the online help topic `Customizing C++ Builds`.

Loading the Run-Time Property DLL

The source code required to load the run-time property DLL was changed in CA Plex r4.0. If your application uses the `SetProperty` API, change the source code that loads the run-time property DLL, as follows:

```
#ifdef _DEBUG
    ObPropertyAPI::SetValue(Ob600Prpd.dll,
                           OB_ATOM_ATOM_INSTALL, 0, 0)
#else
    ObPropertyAPI::SetValue(Ob600Prp.dll,
                           OB_ATOM_ATOM_INSTALL, 0, 0)
#endif
```

User Data (in BSUPPORT) and Business Contacts (in BCONTACT) Patterns

The functions with panels were moved from the container Services to the container UI in CA Plex r4.0. If you have previously used one of these patterns in CA Plex r3.5, you must rescope some of these functions inherited from that pattern. After extracting from the new version of the library, you can see your existing panel functions in the Object Browser under Services as real (in yellow), and new functions with the same names under UI (not yet real, in gray). Rescope the existing panel functions by dragging them from the Services function and dropping them onto the UI function. (Note that the UI function must be made real before doing this.)

Upgrading from CA Plex r3.1 and r3.0

In addition to the instructions in Upgrading from CA Plex r4.0 and Upgrading from CA Plex r3.5, note that CA Plex r3.1 was the last release of CA Plex that supported the creation of 16-bit Windows applications.

Upgrading from CA Plex 2.51 and Earlier

You cannot upgrade CA Plex 2.51 or earlier without first upgrading to CA Plex r3.0. Follow the upgrade instructions published with CA Plex r3.0.

Chapter 3: System Requirements

The following requirements comprise the minimum hardware and software requirements for developing an application with CA Plex.

This section contains the following topics:

- [PC Development Environment](#) (see page 41)
- [Platform-Specific System Requirements](#) (see page 41)
- [Windows C++ Client \(WinC Generator\)](#) (see page 42)
- [System i \(System i 5250 and System i Client/Server Generators\)](#) (see page 42)
- [C# Server Generator](#) (see page 43)
- [C# Client Generator](#) (see page 43)
- [Java Generator](#) (see page 43)
- [EJB Option](#) (see page 44)
- [Windows C++ Server \(WinNTC Generator\)](#) (see page 44)
- [Deployment System Requirements](#) (see page 44)

PC Development Environment

This specification is a recommendation rather than a strict requirement.

- 1 GHz 32-bit (x86) or 64-bit (x64) processor
- Windows 8, Windows 7, Windows XP (SP2 or SP3), Windows Vista SP1, Windows Server 2008, or Windows Server 2012

Note: The Server Core version of Windows is not supported for development or deployment of Plex applications.

- At least 1 GB RAM
- 40 GB hard drive or larger; at least 4 GB of free space is recommended
- A CD-ROM drive
- A 17-inch color monitor capable of running at 1024 x 768 resolution or greater

Customers developing Windows C++ applications should also review the requirements for Visual Studio 2005 at <http://www.microsoft.com>.

Platform-Specific System Requirements

The system requirements depend on the platforms on which you intend to deploy your generated applications. The platform requirements change regularly as vendors release new versions of operating systems, compilers, and databases.

Windows C++ Client (WinC Generator)

Following are the requirements for Windows C++ Client:

- Microsoft Visual Studio 2005 Standard, Professional or Architect edition
 - Notes:**
 - Visual Studio Express edition is not compatible with Plex C++ Builds because it does not include the MFC libraries
 - Visual Studio 2005 SP1 is also required (available for download <http://www.microsoft.com>)
 - Visual Studio 2008 and 2010 are NOT compatible with CA Plex C++ builds. Multiple releases of Visual Studio may be installed and run side-by-side on the same PC.
- CA Plex WinC Run-Time System
- ODBC Driver and Database (if client-side data access is required)
- Deployment platforms:
 - Microsoft Windows XP SP2 or SP3
 - Microsoft Windows Vista SP1
 - Microsoft Windows 7
 - Microsoft Windows 8

System i (System i 5250 and System i Client/Server Generators)

Following are the requirements for System i 5250 and System i Client/Server Generator:

- TCP/IP
- CA Plex System i libraries
- RPG III or RPG IV compiler
- V5R4, 6.1 or 7.1
- CA License key for 5250 generator (if required)

C# Server Generator

Following are the requirements for C# Server generator:

- Microsoft .NET Framework Version 4.0 (can be installed from the CA Plex installation CD). Both 32-bit and 64-bit .NET Frameworks are supported.
- DBMS (Microsoft SQL Server 2008, Oracle 11g, and IBM DB2 9.5) and OLE DB Provider.
- ODBC driver for database builds, not required at run time

Note: Microsoft Visual Studio is *not* required for C# development. Optionally, you may find Visual Studio useful for debugging and other development activities.

C# Client Generator

Following are the requirements for C# Client generator:

- Microsoft .NET Framework Version 4.0 (can be installed from the CA Plex installation CD). Both 32-bit and 64-bit .NET Frameworks are supported.
- CA License Key for .NET Client generator.

Java Generator

Following are the requirements for Java generator:

- Sun J2SE Development Kit 6.0. Both 32-bit and 64-bit Java Runtime Environments are supported.
- Apache ANT 1.8.2 (automatically installed with CA Plex)
- Third-party DBMS, such as, Oracle, Microsoft SQL Server, or any third-party DBMS and JDBC driver
- ODBC driver for database builds, not required at run time

EJB Option

In addition to the requirements for the Java generator, following are the requirements for the EJB option:

- Java EE Application Server, such as, Oracle GlassFish, Oracle WebLogic, JBoss, or IBM WebSphere
 - Java EE 6 SDK
- Note:** The application must be packed by J2EE 1.3.
- CA License Key for EJB Connector generation.

Windows C++ Server (WinNTC Generator)

Following are the requirements for Windows C++ Server:

- CA Plex Windows Application Server.
- Microsoft Visual Studio 2005 Standard or Professional edition. For more information about Visual Studio, see the Windows C++ Client (WinC Generator) section in this guide.
- DBMS (Microsoft SQL Server 2008, Oracle 11g, or IBM DB2 9.5). The latest available service packs are generally recommended.
- Microsoft Windows Server 2003, Microsoft Windows Server 2008, or Microsoft Server 2012.

Note: The Windows C++ Server generator is provided only for maintenance of existing WinNTC applications. CA strongly recommends the use of C# or Java generators for the development of new applications for the Windows Server platform.

Deployment System Requirements

As with development, the deployment system requirements vary depending on the target platform. You can find the details of currently supported deployment platforms by going to CA Support Online and clicking the Compatibility Matrix link on the CA Plex Product Home page at <http://ca.com/support>.

Chapter 4: Installation Considerations

You can find installation and upgrade procedures for this release of CA Plex in the *Getting Started*.

You can also perform a silent install of CA Plex. For more information about the silent installation of CA Plex, see the *Getting Started*.

This section contains the following topics:

[Microsoft Help Compiler Installation](#) (see page 45)

Microsoft Help Compiler Installation

Microsoft Visual Studio 2005 does not include the help compiler for creating HLP format help files.

To compile HLP files with Plex, you must download and install the help compiler separately. You can find Windows 95 Help Authoring Kit (hwcsetup.exe) available for download at <http://www.microsoft.com>. You must install the MS help compiler to its default location: C:\Program Files\Help Workshop.

The minimum requirement for successful help topic compilation is to have two files—HCRTF.EXE and HWDLL.DLL present in the folder C:\Program Files\Help Workshop.

Chapter 5: Published Fixes

You can find the complete list of published bug fixes for this product through Published Solutions on the CA Technologies support site.

Chapter 6: Known Issues

This section describes known issues, workarounds, and solutions for this release of CA Plex. For additional information, search the Knowledge Base on CA Support Online at <http://ca.com/support>.

This section contains the following topics:

[Modification of Internal Tables](#) (see page 49)

[C++ Build Performance with Visual Studio 2005](#) (see page 49)

Modification of Internal Tables

Although it is possible to edit internal tables in CA Plex internal tables to add your own object types and verbs, this is an undocumented and unsupported feature. CA does not guarantee to support models that have been modified in this way. It is strongly recommended that you do *not* modify the internal tables in CA Plex in any way.

Similarly, manual editing of any group model table files is also not supported. This includes the lib.tab file that contains sensitive data that controls aliasing of group model IDs.

C++ Build Performance with Visual Studio 2005

You may experience slow C++ build performance compared with previous releases. For tips on improving the performance of WinC and WinNTC compiles using Visual Studio 2005, see the Technical Document *TEC415593* in the Knowledge Base on CA Support Online at <http://ca.com/support>.

Chapter 7: International Support

An internationalized product is an English product that runs correctly on local language versions of the required operating system and required third-party products, and supports local language data for input and output. Internationalized products also support the ability to specify local language conventions for date, time, currency and number formats.

A translated product (sometimes referred to as a localized product) is an internationalized product that includes local language support for the product's user interface, online help and other documentation, as well as local language default settings for date, time, currency, and number formats.

In addition to the English release of this product, CA supports only those languages listed in the following table.

Language	Internationalized	Translated
Brazilian-Portuguese	Yes	No
Chinese (Simplified)	Yes	No
Chinese (Traditional)	Yes	No
Czech	Yes	No
Danish	Yes	No
Dutch	Yes	No
Finnish	Yes	No
French	Yes	No
German	Yes	No
Greek	Yes	No
Hungarian	Yes	No
Italian	Yes	No
Japanese	Yes	Yes
Korean	Yes	No
Norwegian	Yes	No
Polish	Yes	No
Russian	Yes	No
Spanish	Yes	No

Language	Internationalized	Translated
Swedish	Yes	No
Turkish	Yes	No

Limited testing has been performed on languages other than English. This limited testing may comprise testing by CA partners or customers or be based on historical testing of prior releases of CA Plex.

This certification means that this English release will run on the non-English operating systems on those machines. It is not intended to mean that the product is localized for those markets and languages.

Chapter 8: Documentation

The following guides are automatically installed with the product and available at <http://ca.com/support>:

- *Plex_Getting_Started_ENU.pdf*
- *Plex_Release_Summary_ENU.pdf*
- *Plex_Tutorial_for_System_i_ENU.pdf*
- *Plex_Tutorial_for_Windows_ENU.pdf*

To view PDF files, you must install the Adobe Reader if it is not already installed on your computer.

Note: Updated guides for this release are available under the CA brand drop-down list on the Documentation page at <http://ca.com/support>.