

# CA DataMinder

## Account Import XML Schema Guide

Release 14.5



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 may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Documentation is confidential and proprietary information of CA and may not be disclosed by you or used for any purpose other than as may be permitted in (i) a separate agreement between you and CA governing your use of the CA software to which the Documentation relates; or (ii) a separate confidentiality agreement between you and CA.

Notwithstanding the foregoing, 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 © 2013 CA. All rights reserved. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

## CA Technologies Product References

This document references the following CA Technologies products:

- CA DataMinder

## 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](mailto: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>.



# Contents

---

|                                             |           |
|---------------------------------------------|-----------|
| <b>Chapter 1: Introduction</b>              | <b>7</b>  |
| <b>Chapter 2: Example XML Data File</b>     | <b>9</b>  |
| Schema Notes.....                           | 11        |
| <b>Chapter 3: Account Import XML Schema</b> | <b>13</b> |
| Supported XML Schema .....                  | 13        |
| <b>Appendix A: Accessibility Features</b>   | <b>23</b> |
| Display .....                               | 23        |
| Sound .....                                 | 24        |
| Keyboard .....                              | 24        |
| Mouse .....                                 | 25        |



# Chapter 1: Introduction

---

This guide summarizes the schema for XML data files used by CA DataMinder for Account Import operations.

Account Import enables administrators to import user details into CA DataMinder from an external Lightweight Directory Access Protocol (LDAP) directory or a source file. Account Import can import new users and groups into the existing CA DataMinder user hierarchy, or it can reorganize existing users to synchronize them with an external hierarchy. It can also import user attributes such as e-mail addresses and employee IDs.



## Chapter 2: Example XML Data File

---

This section shows an example XML file used in an Account Import operation.

**Note:** See the Schema Notes for details about the <root>, <hierarchy> and <users> sections.

```
<?xml version='1.0' encoding='UTF-8'?>
<accountimport version='4.7' format='hierarchical' preserveuniquegroups='True'
add_db='true' xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'>
  <root>
    <group name='Unipraxis'>
      <group name='Directors' />
      <group name='Development'>
        <group name='Senior Software Engineers' />
        <group name='Quality Assurance' />
        <group name='Software Engineers'>
          <user policyexempt='true'>
            <name>UNIPRAXIS\srimmel</name>
            <fullname>Spencer Rimmel</fullname>
            <role>Manager</role>
            <reportname>srimmel</reportname>
            <securitymodel description='Policy (All Events, Restricted Triggers) and
Management Group (Standard, Self-Exclude)'>PA,MDX</securitymodel>
            <policyroles>
              <policyrole>PCI Compliance Policies</policyrole>
            </policyroles>
            <mgmtgroups>
              <group>
                <element>Unipraxis</element>
                <element>Development</element>
                <element>Software Engineers</element>
              </group>
              <group>
                <element>Unipraxis</element>
                <element>Development</element>
                <element>Senior Software Engineers</element>
              </group>
            </mgmtgroups>
          </user>
        </group>
      </group>
    </group>
  </root>
</accountimport>
```

```
<attributes>
  <attr index='1' xsi:type='IndexedAttribute'>
    <value>Development</value>
  </attr>
  <attr index='4' xsi:type='IndexedAttribute'>
    <value>Taunton</value>
  </attr>
  <attr index='7' xsi:type='IndexedAttribute'>
    <value></value>
  </attr>
  <attr xsi:type='EmailAttribute'>
    <value>software.developer@unipraxis.com</value>
  </attr>
</attributes>
</user>
</group>
</group>
</group>

<hierarchy relativeTo='Development'>
  <group name='Quality Assurance'>
    <user policyexempt='false'>
      <name>UNIPRAXIS\fschaeffer</name>
      <fullname>Frank Schaeffer</fullname>
      <role>User</role>
      <attributes>
        <attr index='1' xsi:type='IndexedAttribute'>
          <value>Development</value>
        </attr>
        <attr index='4' xsi:type='IndexedAttribute'>
          <value>Taunton</value>
        </attr>
        <attr xsi:type='EmailAttribute'>
          <value>qa.engineer@unipraxis.com</value>
        </attr>
      </attributes>
    </user>
  </group>
</hierarchy>
```

```
<users>
  <user>
    <name>UNIPRAXIS\lsteel</name>
    <fullname>Lynda Steel</fullname>
    <role>User</role>
    <group isRelative='true'>
      <element>directors</element>
    </group>
  </user>
</users>
</root>
</accountimport>
```

**More information:**

[Schema Notes](#) (see page 11)

## Schema Notes

**<accountimport> element**

The <accountimport> element denotes which version of CA DataMinder the account import operation will synchronize with. If this tag includes preserveuniquegroups='true', then if a group with a unique name exists in the xml file and the same group name is also unique in the CA DataMinder database but in a different location, then a <movegroup> command is performed to move the group from the current CA DataMinder location to the new location specified in the xml file, leaving only one group with that name in the CA DataMinder database; if this tag includes add\_db='true', then the current CA DataMinder group hierarchy is added to the <root> element (see below) when the XML file is imported. If the XML file does not contain a <root> element, then one will be added.

**<root> element**

The <root> element creates the entire hierarchy of groups and users. There can only be one <root> element per file. If your groups and users are contained in separate databases, you may want to create your hierarchy using the <hierarchy> and <user> elements - see below.

### <hierarchy> element

The <hierarchy> element represents a subsection of the group structure. The relativeTo attribute value is the name of the group in the <root> section where you want to insert this subsection. Any group name specified as a relativeTo attribute value must already exist and be unique within the <root> section. In the following example, the <root> section *must* contain a unique group called Development; a QualityAssurance group is then created within it:

```
<hierarchy relativeTo='Development'>
  <group name='QualityAssurance' />
</hierarchy>
```

The relativeTo attribute is optional, but if it is not included, the hierarchy is inserted at the top of the <root> section, that is, it will be identical to the <root> section. If you think you need a <hierarchy> section without a relativeTo attribute, it may be better to add this directly into the <root> section, as the XML file can then be processed quicker.

### <user> elements

<user> elements in the <users> section are added to the <root> section based on their <group> element. The <group> element can specify either the complete group name or a relative group name. For example, the following element adds Spencer Rimmel to the Unipraxis/Directors group, while the *policyexempt* attribute exempts him from policy.

```
<users>
  <user policyexempt='true'>
    <name>SpencerRimmel</name>
    <group>
      <element>Unipraxis</element>
      <element>Directors</element>
    </group>
  </user>
</users>
```

The next example adds Lynda Steel to the same Directors group. As with the <hierarchy> element's relativeTo attribute, if the isRelative attribute is set to true, the first <element> value must specify a group that already exists and is unique in the <root> section. In this example, no *policyexempt* attribute is specified, so the attribute defaults to 'false'. That is, Lynda Steel is not exempt from policy.

```
<users>
  <user>
    <name>Lynda_Steel</name>
    <group isRelative='true'>
      <element>Directors</element>
    </group>
  </user>
</users>
```

# Chapter 3: Account Import XML Schema

---

This is the XML schema that can be used to validate the format of XML files you want to import.

## Supported XML Schema

```
<?xml version="1.0" ?>
<!DOCTYPE xsd:schema>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            elementFormDefault="qualified"
            xml:lang="en">
  <xsd:annotation>
    <xsd:documentation>
      CA DataMinder Account Import XML data format Version 2
      This version is used to validate the hierarchical XML.
      Copyright 2012 CA.
      All rights reserved.
    </xsd:documentation>
  </xsd:annotation>

  <!-- Main element - accountimport -->
  <xsd:element name="accountimport">
    <xsd:annotation>
      <xsd:documentation>
        Main element of the document.
        Mandatory
        One instance only
        Contains 'root' and/or 'hierarchy' and/or 'users' elements
      </xsd:documentation>
    </xsd:annotation>
  </xsd:element>
</xsd:schema>
```

```
<xsd:complexType>
  <xsd:choice minOccurs="1" maxOccurs="unbounded">
    <!-- First level sub-element - root -->
    <xsd:element name="root" type="root" minOccurs="0" maxOccurs="1" />
    <!-- First level sub-element - users -->
    <xsd:element name="users" type="users" minOccurs="0"
      maxOccurs="unbounded" />
    <!-- First level sub-element - hierarchy -->
    <xsd:element name="hierarchy" type="hierarchy" minOccurs="0"
      maxOccurs="unbounded" />
  </xsd:choice>

  <!-- Version attribute -->
  <!-- Currently only permits 4.0 and 4.7 -->
  <xsd:attribute name="version" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:double">
        <xsd:enumeration value="4.0" />
        <xsd:enumeration value="4.7" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>

  <!-- Currently only permits "hierarchical" -->
  <xsd:attribute name="format" use="required">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="hierarchical" />
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:attribute>

  <!-- Add_db attribute -->
  <xsd:attribute name="add_db" type="xsd:boolean" default="false" />
</xsd:complexType>
</xsd:element>
```

```

<!-- Types within first level of accountimport -->
<xsd:complexType name="root">
  <xsd:choice minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="user" type="HierarchyUser" minOccurs="0"
      maxOccurs="unbounded" />
    <xsd:element name="group" type="HierarchyGroup" minOccurs="0"
      maxOccurs="unbounded" />
  </xsd:choice>
</xsd:complexType>
<xsd:complexType name="users">
  <xsd:sequence>
    <xsd:element name="user" type="FlatUser" minOccurs="0"
      maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

<!-- Hierarchy type -->
<xsd:complexType name="hierarchy">
  <xsd:choice minOccurs="1" maxOccurs="unbounded">
    <xsd:element name="user" type="HierarchyUser" minOccurs="0"
      maxOccurs="unbounded" />
    <xsd:element name="group" type="HierarchyGroup" minOccurs="0"
      maxOccurs="unbounded" />
  </xsd:choice>

  <!-- relativeTo attribute -->
  <xsd:attribute name="relativeTo" type="xsd:string" />
</xsd:complexType>

<!-- HierarchyUser type -->
<xsd:complexType name="HierarchyUser">
  <xsd:annotation>
    <xsd:documentation>
      User sub-element that may be found in 'root' or 'hierarchy' elements
      Zero or more instances allowed.
      Sub-elements described below
    </xsd:documentation>
  </xsd:annotation>

```

```
<xsd:all>
  <xsd:annotation>
    <xsd:documentation>
      Sub-elements of 'HierarchyUser' are expected as follows
      'name' - mandatory, one instance only
      'fullname' - optional, one instance only
      'role' - mandatory, one instance only
      'reportname' - optional, one instance only
      'mgmtgroups' - optional, one instance only
      'attributes' - optional, one instance only
      'securitymodel' - optional, one instance only
      'policyroles' - optional, one instance only
    </xsd:documentation>
  </xsd:annotation>

  <!-- user's name, full name, role, management groups and attributes -->
  <xsd:element name="name" type="xsd:string" />
  <xsd:element name="fullname" type="xsd:string" minOccurs="0" />
  <xsd:element name="role" type="Role" default="User" />
  <xsd:element name="reportname" type="xsd:string" minOccurs="0" />
  <xsd:element name="mgmtgroups" type="MgmtGroups" minOccurs="0" />
  <xsd:element name="attributes" type="Attributes" minOccurs="0" />
  <xsd:element name="securitymodel" type="SecurityModel" minOccurs="0" />
  <xsd:element name="policyroles" type="PolicyRoles" minOccurs="0" />
</xsd:all>

<xsd:attribute name="policyexempt" type="xsd:boolean" default="false"/>

</xsd:complexType>
```

```

<!-- FlatUser type -->
<xsd:complexType name="FlatUser">
  <xsd:annotation>
    <xsd:documentation>
      User sub-element that may be found
      in 'users' elements.
      Zero or more instances allowed.
      Sub-elements described below.
    </xsd:documentation>
  </xsd:annotation>

  <xsd:all>
    <xsd:annotation>
      <xsd:documentation>
        Sub-elements of 'FlatUser' are expected as follows
        'name' - mandatory, one instance only
        'fullname' - optional, one instance only
        'group' - mandatory, one instance only
        'role' - mandatory, one instance only
        'reportname' - optional, one instance only
        'mgmtgroups' - optional, one instance only
        'attributes' - optional, one instance only
        'securitymodel' - optional, one instance only
        'policyroles' - optional, one instance only
      </xsd:documentation>
    </xsd:annotation>

    <!-- the user's name, full name, group, role,
         management groups and attributes -->
    <xsd:element name="name" type="xsd:string" />
    <xsd:element name="fullname" type="xsd:string" minOccurs="0" />
    <xsd:element name="group" type="FlatGroup" />
    <xsd:element name="role" type="Role" default="User"/>
    <xsd:element name="reportname" type="xsd:string" minOccurs="0" />
    <xsd:element name="mgmtgroups" type="MgmtGroups" minOccurs="0" />
    <xsd:element name="attributes" type="Attributes" minOccurs="0" />
    <xsd:element name="securitymodel" type="SecurityModel" minOccurs="0" />
    <xsd:element name="policyroles" type="PolicyRoles" minOccurs="0" />

  </xsd:all>

  <xsd:attribute name="policyexempt" type="xsd:boolean" default="false"/>
</xsd:complexType>

<!-- AbsoluteHierarchyGroup type -->
<xsd:complexType name="AbsoluteHierarchyGroup">
  <xsd:choice minOccurs="0" maxOccurs="unbounded">

```

```
<xsd:element name="user" type="HierarchyUser" minOccurs="0"
  maxOccurs="unbounded" />
<xsd:element name="group" type="AbsoluteHierarchyGroup" minOccurs="0"
  maxOccurs="unbounded" />
</xsd:choice>
<xsd:attribute name="name" type="xsd:string" />
</xsd:complexType>

<!-- HierarchyGroup type -->
<xsd:complexType name="HierarchyGroup">
  <xsd:complexContent>
    <xsd:extension base="AbsoluteHierarchyGroup">
      <xsd:attribute name="isRelative" type="xsd:boolean" default="false" />
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- Role type -->
<xsd:simpleType name="Role">
  <xsd:annotation>
    <xsd:documentation>
      The role that a user has, e.g. Manager, Policy Reviewer  

Values are configurable and cannot be validated by this schema
    </xsd:documentation>
  </xsd:annotation>
  <xsd:restriction base="xsd:string" />
</xsd:simpleType>

<!-- MgmtGroups type -->
<xsd:complexType name="MgmtGroups">
  <xsd:annotation>
    <xsd:documentation>
      Contains one or more 'group' subelements
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="group" type="FlatGroup" minOccurs="0"
      maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>
```

```
<!-- Attributes type -->
<xsd:complexType name="Attributes">
  <xsd:annotation>
    <xsd:documentation>
      Contains one or more 'attr' subelements
      'attr' elements can be one of three formats.
      See below for description.
      'BaseAttribute' is the base format. The other formats extend it.
      If no value is specified, the value of the attribute is deleted.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="attr" type="BaseAttribute" maxOccurs="unbounded" />
  </xsd:sequence>
</xsd:complexType>

<!-- BaseAttribute type -->
<xsd:complexType name="BaseAttribute" abstract="true" >
  <xsd:annotation>
    <xsd:documentation>
      The basic 'attr' element
      No attributes
      One or more 'value' elements
      Abstract, so cannot be used directly in an XML instance.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="value" type="xsd:string" minOccurs="0"
      maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

```
<!-- EmailAttribute type -->
<xsd:complexType name="EmailAttribute">
  <xsd:annotation>
    <xsd:documentation>
      The e-mail 'attr' element
      Used to represent e-mail addresses associated with the user.
      Identical to the base attribute type
      No attributes
      One or more 'value' elements, each containing an e-mail address
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="BaseAttribute"/>
  </xsd:complexContent>
</xsd:complexType>

<!-- NamedAttribute type -->
<xsd:complexType name="NamedAttribute">
  <xsd:annotation>
    <xsd:documentation>
      A named 'attr' element
      The name will be matched against the list of
      CA DataMinder property value names and converted to the
      appropriate index.
      Any attribute with a name that does not match
      an CA DataMinder property value name will be ignored.
      One mandatory attribute (name), contains the
      name of the attribute. One or more 'value' elements,
      each containing a value for the attribute
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="BaseAttribute">
      <xsd:attribute name="name" type="xsd:string" use="required" />
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
```

```

<!-- IndexedAttribute type -->
<xsd:complexType name="IndexedAttribute">
  <xsd:annotation>
    <xsd:documentation>
      An indexed 'attr' element
      The index corresponds to one of the internal
      CA DataMinder attributes. One mandatory attribute (index),
      contains a value greater than or equal to 1
      One optional element (displayname), the value of
      which will be the name of the corresponding internal
      CA DataMinder attribute. There purely to make the XML more
      'user friendly' to human readers.
      One or more 'value' elements, each containing a value
      for the attribute.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:complexContent>
    <xsd:extension base="BaseAttribute">
      <xsd:attribute name="index" use="required">
        <xsd:simpleType>
          <xsd:restriction base="xsd:integer">
            <xsd:minInclusive value="1" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:attribute>
      <xsd:attribute name="displayname" type="xsd:string" />
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

<!-- FlatGroup type -->
<xsd:complexType name="FlatGroup">
  <xsd:annotation>
    <xsd:documentation>
      Zero or more instances allowed.
      Contains zero or more 'element' elements, each
      representing a segment of the path. If there are no
      'element' elements, the root path is being represented.
      Optional 'isRelative' attribute, indicates if group
      is a subgroup of another group in the XML file
      Used in FlatUser and MgmtGroups types
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="element" type="xsd:string" minOccurs="0"
      maxOccurs="unbounded" />
  </xsd:sequence>

```

```
<!-- isRelative attribute -->
<xsd:attribute name="isRelative" type="xsd:boolean" default="false" />
</xsd:complexType>

<xsd:complexType name="SecurityModel">
  <xsd:annotation>
    <xsd:documentation>
      Zero or one instance allowed.
      Contains details of the security model assigned to a user.
      'description' attribute contains full description of the security model.
      The tag's value is the short code representation of the security model.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="description" type="xsd:string" />
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
<xsd:complexType name="PolicyRoles">
  <xsd:annotation>
    <xsd:documentation>
      Zero or one instance allowed.
      Contains one mandatory 'policyrole' subelement.
      The XML is configured this way because,
      although a user is currently allowed only one policy role,
      this may be changed in future to allow multiple policy roles.
    </xsd:documentation>
  </xsd:annotation>
  <xsd:sequence>
    <xsd:annotation>
      <xsd:documentation>
        One instance allowed.
        The policy role applied to the user.
      </xsd:documentation>
    </xsd:annotation>
    <xsd:element name="policyrole" type="xsd:string" />
  </xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

# Appendix A: Accessibility Features

---

CA Technologies is committed to ensuring that all customers, regardless of ability, can successfully use its products and supporting documentation to accomplish vital business tasks. This section outlines the accessibility features that are supported by CA DataMinder.

## Display

To increase visibility on your computer display, you can adjust the following options:

### **Font style, color, and size of items**

Defines font color, size, and other visual combinations.

The CA DataMinder iConsole also supports a High Visibility mode. This increases the size of text and images in the iConsole screens.

### **Screen resolution**

Defines the pixel count to enlarge objects on the screen.

### **Cursor width and blink rate**

Defines the cursor width or blink rate, which makes the cursor easier to find or minimize its blinking.

### **Icon size**

Defines the size of icons. You can make icons larger for visibility or smaller for increased screen space.

### **High contrast schemes**

Defines color combinations. You can select colors that are easier to see.

## Sound

Use sound as a visual alternative or to make computer sounds easier to hear or distinguish by adjusting the following options:

### **Volume**

Sets the computer sound up or down.

### **Text-to-Speech**

Sets the computer's hear command options and text read aloud.

### **Warnings**

Defines visual warnings.

### **Notices**

Defines the aural or visual cues when accessibility features are turned on or off.

### **Schemes**

Associates computer sounds with specific system events.

### **Captions**

Displays captions for speech and sounds.

## Keyboard

You can make the following keyboard adjustments:

### **Repeat Rate**

Defines how quickly a character repeats when a key is struck.

### **Tones**

Defines tones when pressing certain keys.

### **Sticky Keys**

Defines the modifier key, such as Shift, Ctrl, Alt, or the Windows Logo key, for shortcut key combinations. Sticky keys remain active until another key is pressed.

## Mouse

You can use the following options to make your mouse faster and easier to use:

**Click Speed**

Defines how fast to click the mouse button to make a selection.

**Click Lock**

Sets the mouse to highlight or drag without holding down the mouse button.

**Reverse Action**

Sets the reverse function controlled by the left and right mouse keys.

**Blink Rate**

Defines how fast the cursor blinks or if it blinks at all.

**Pointer Options**

Let you do the following:

- Hide the pointer while typing
- Show the location of the pointer
- Set the speed that the pointer moves on the screen
- Choose the pointer's size and color for increased visibility
- Move the pointer to a default location in a dialog box