Upgrading to Servoy 7.x.x

In This Chapter

- Upgrading Installations
  - Upgrade Servoy 6.1.x Installations to Servoy 7.x
  - Upgrading Servoy Application Server
  - Upgrading Servoy Developer
- Upgrading Existing Solutions
  - Behavior Changes
    - Changes That Might Break Existing Code
  - Deprecated API
    - Designtime API
    - Runtime API

Upgrading Installations

In order to provide the best possible security, Servoy 7 comes with a renewed Code Signing certificate, that utilizes OCSP. As Java 5 is End Of Live (see http://www.oracle.com/technetwork/java/eol-135779.html) it does not support OCSP, which makes Java 6 the new minimum Java version for Servoy 7.

Upgrade Servoy 6.1.x Installations to Servoy 7.x

As of November 8th 2013, the Servoy 6.1.x branch is merged into the 7.x branch, as outlined in the Servoy Release Policy update post on the Servoy Blog. The specific details of upgrading Servoy 6.1.x Developer or Servoy 6.1.x Application Server installations to 7.3 are detailed in the Servoy 6.1.6 to Servoy 7.3 upgrade on Servoy Talk.

Upgrading Servoy Application Server

Installations running an older version of Servoy can't be upgraded, thus a new installation is required. See Installing the Application Server for more information.

Upgrading Servoy Developer

Existing Servoy 6.1.x installation can be upgraded to Servoy 7.0.x. In order to do so, open the Available Update Sites preferences and edit the URL of the **Servoy update site** entry:

- Open **Window** > **Preferences** > **Install/Update** > **Available Software Sites**
- Find the entry named **Servoy update site** and select the row
- Click **Edit**
- Change the Location field from `https://www.servoy.com/developer/61x_updates` to `https://www.servoy.com/developer/70x_updates`
- Close all dialogs
- Go to **Help** > **Check for updates** to trigger the update
- After the update has completed, restart Servoy Developer
- when Servoy Developer is launched again, it will prompt for an update of the underlying Servoy Application Server.

Servoy Developer installations of older Servoy version cannot be upgraded, so a new installation with the Servoy 7 installer is required. See Installing Developer for more information.

Upgrading Existing Solutions

Upgrading existing Solutions to Servoy 7 is as easy as opening Servoy Developer 7 and getting the Solutions into the workspace or opening an existing workspace in Servoy 7. Note that once solutions are edited using Servoy 7, they cannot be used in earlier versions of Servoy anymore.

Once imported, the Solutions can be tested with Servoy 7. Servoy 7 introduces a few minor behavior changes, some of which could potentially break existing code. It should be investigated if the upgraded solutions are affected by these changes. The behavior changes are listed and discussed in the paragraph #Behavior Changes.

Behavior Changes

This paragraph describes the behavior changes that are introduced in Servoy 6.0. There are three categories of behavior changes:

1. Changes that might break existing code
2. Real behavior changes
3. Changes that affect the display of the UI

Changes That Might Break Existing Code

- As of Servoy 7 Forms that are in ClientDesign mode will retain their Client Design mode status if they are hidden from the UI and then shown again
- As of Servoy 7 when a column has a Column Converter attached, the output of the converter will be used for the SQL layer of Servoy in FindMode or with Table/FoundSet filters. Prior to Servoy 7, the supplied values in FindMode by the user or through scripting or the values used in Table /FoundSet filters would not be passed through the column Converter before being used in the SQL layer
- As of Servoy 7 when altering inherited elements their z-index will remain the same, unless the z-index is explicitly altered by the developer. Prior to Servoy 7, altering any property of an inherited element would cause the z-index to change.

Deprecated API

Servoy 7 deprecates parts of the existing API, as it has become obsolete or has been replaced by better alternatives. This section lists the deprecated methods or properties with their replacement where applicable.

Designtime API

The list of deprecated properties in the Designtime API is limited. Except for the deprecation of the `rowBGColorCalculation` on Forms and Portals, the deprecation of the properties require no action on behalf of the developer, as Servoy Developer will automatically handle the changes.

Object	Deprecated method or property	Replacement	Comment
Field	text	titleText	To better align name with actual function
Label, Button, Image	rotation	textRotation	To better align name with actual function

Runtime API

The majority of the deprecations in the Runtime API are related to the new windowing API and the refactored HTTP plugin.

Object	Deprecated method or property	Replacement	Comment
globals	allmethods	solutionModel.getGlobalMethods("scopeName")	an exact replacement would be: solutionModel.getGlobalMethods("scopeName").map(function (jsMethod) { return jsMethod.getName() })
globals	allrelations	solutionModel.getRelations(null)	an exact replacement would be: solutionModel.getRelations(null).map(function (jsRelation) { return jsRelation.name })
globals	allvariables	solutionModel.getGlobalVariables("scopeName")	an exact replacement would be: solutionModel.getGlobalVariables("scopeName").map(function (jsVariable) { return jsVariable.name })
forms	allnames	solutionModel.getForms()	an exact replacement would be: solutionModel.getForms().map(function (jsForm) { return jsForm. name })
RuntimeForm	alldataproviders	foundset.alldataproviders	
RuntimeForm	allnames	elements.allnames	
RuntimeForm	allrelations	solutionModel.getRelations(dataSource) where dataSource is provided by solution model's JSForm.dataSource or controller. getDataSource()	an exact replacement would be: solutionModel.getRelations (controller.getDataSource()).map(function (jsRelation) { return jsRelation.name })
RuntimeForm	allvariables	solutionModel - JSForm.getVariables(Boolean) or JSForm. getVariables()	an exact replacement would be: 'solutionModel.getForm(controller.getName()).getVariables().map (function (jsVariable) { return jsVariable.name })
ELEMENT_TYPES	LIST_BOX	LISTBOX	Alignment of names used
JSField	LIST_BOX	LISTBOX	Alignment of names used
ALIGNMENT	ACCORDION_PANEL	JSTabPanel.ACCORDION_PANEL	
ALIGNMENT	SPLIT_HORIZONTAL	JSTabPanel.SPLIT_HORIZONTAL	
ALIGNMENT	SPLIT_VERTICAL	JSTabPanel.SPLIT_VERTICAL	
JSPart, JSRelationItem, JSTab	getDesignTimeProperty putDesignTimeProperty, removeDesignTimeProperty		Not relevant
databaseManager	hasNewRecords (JSRecord)	JSRecord.isNew()	

database Manager	hasRecordChanges(JSRecord)	JSRecord.hasChangedData()	
database Manager	revertEditedRecords(JSRecord)	JSRecord.revertChanges()	
I18N	setI18NMessagesFilter	Use TableFilters instead	