Code Writing Features

This section describes useful code writing features in Servoy Script Editor.

Code Completion

Servoy supports code completion. As users type into the Script Editor, all known, non-private functions are displayed in a drop-down select box that becomes filtered as more characters are typed. A description box also shows when an item is selected (see below).

 alignment: Form < alignment> alinames: Array < String> core_dlg_generic: Form < core_dlg_generic> customers: Form < customers> customers_new1: Form < customers_new1> customers_new3: Form < customers_new3> customers_new4: Form < customers_new4> customers_new: Form < customers_new2> acustomers_new: Form < customers_new3> acustomers_new: Form < customers_new3> acustomers_new3 customers_new3 	Form based on datasource: db:/example_data/customers
distribute: Form <distribute> Distribution: Form<distribution></distribution></distribute>	Press 'Tab' from proposal table or click for foc

• To insert a selected item into the code, hit Enter.

Code Hints

To see information relevant to a particular code item, users can hover over the relevant parts of the code. This is true for both the code list shown in the Solution Explorer, as well as code in the Script Editor view. Servoy will provide an item description if it exists, for both out-of-the box functions as well as functions defined by the user via JSDoc (see Support for JSDoc__section below).

Dynamic Code Validation

Code is dynamically validated at design-time. The success of the validation process is partially dependent on comprehensive JSDoc commenting, which enables Servoy to provide more information regarding code errors and other warnings. Warnings are flagged in the vertical ruler and outline bar, and details are provided in the Problems view at the bottom of the Servoy Workbench (see image below).

custome	rs_new1.js 🖉 globals.js 🕱 🎯 iServoy 📲 Compare globals.js C "1 🗖	• 🗆				
2//	globals.record_status += " - SEAKLH IEKM: '" + globals.nav_search + "'"	2				
278	globals.nav_search = ''					
279	}					
280						
281	globals.record_status += ''					
282						
283	if(cr == 1) //current record 1 - so hide the "previous" button					
284	{					
<u>&</u> 285	<pre>forms[frm].elements.btn_prev.visible = false</pre>					
286	}					
287	else					
288	{					
289	<pre>forms[frm].elements.btn_prev.visible = true</pre>					
290	}					
291	iffer follow to //www.et is lost one on hide the "mout" butter					
<pre>292 if(cr == fs cr == tc) //current is last one - so hide the "next" button 293 {</pre>						
A295	{ forms[frm].elements.btn_next.visible = false					
295	}					
296	else					
297	{					
A298	forms[frm].elements.btn_next.visible = true					
299						
300	}					
301	else					
302	{					
303	globals.record_status = ' <html>No Records'</html>					
<pre>A304 forms[frm].elements.btn_next.visible = false</pre>						
305	<pre>forms[frm].elements.btn_prev.visible = false</pre>					
306	}					
307 }						
308		U				
309 /**		^				
310 * 0	<pre>@properties={typeid:24,uuid:"4728bb29-9b80-4695-8a71-434b8dd97298"}</pre>	¥ 🗖				
e						
Problems		• 🗆				
	warnings, 0 others	т.				
Description		Ту				
	ings (60 items)	- 0				
▲ Element "tabless_140" in form "frm_nav_main" frm_nav_main.frm /servoy_sample_pdf_forms servoy_sample_Fo						
A Element in form "customers_new3" is outside customers_new3.frm /servoy_sample_pdf_forms servoy_sample_Fd						
A Element in form "customers new3" is outside customers new3.frm /servov sample pdf form servov sample Ed						

As previously mentioned, the type of warnings/annotations that Servoy shows can be customized using the **Preferences** menu (found in the Script Editor context menu).

Script Templates

Servoy allows users to create templates for frequently-used scripts and JSDoc comments. These scripts are then enabled by Servoy for use in code completion.

Templates can be created by accessing **Preferences** from the contextual menu in Script Editor (users may have to unfold the menu tree on the left of the Preferences window to see the Templates option, see below).

type filter text General JavaScript Editor Folding Templates Create, edit or Name @author	remove template Context JSDoc JSDoc	es: Description author name	Auto Ins	⇔ ∗ ⇒∗ ▼
▼JavaScript Create, edit or ▼Editor Name	Context JSDoc	Description	Auto Ins	
VEditor Name	Context JSDoc	Description	Auto Ins	
	JSDoc		Automs	
l lolang			on	New
Hovers 🗹 @param	JSDUC	parameter declaration	on	
Mark Occurrences	JSDoc	function return type dec	on	Edit
Syntax Coloring	JSDoc	thrown exception declar	on	
Templates @ @type	JSDoc	variable type declaration		Remove
Typing do	JavaScript	do-while	on	Kennove
M for	JavaScript	Simple loop	on	
M function	JavaScript	method definition	on	(Restore Removed)
✓ function	JavaScript	simple function with con		
✓ ifelse	JavaScript	if - else	on	(Devent to Default)
switch	JavaScript	switch	on	Revert to Default
☑ try	JavaScript	try catch	on	
while	JavaScript	while loop	on	Import
	Jurusenpe	e.e.e.e		
				Export
Preview:				
		Resto	re Default	Apply
?		\subset	Cancel	ОК

Support for JSDoc

Servoy features support for regular comments and JSDoc comments. Importantly, disciplined use of JSDoc comments when creating functions and variables ensures more complete information for Servoy in refactoring and code completion.

JSDoc tags can be created for an existing function by accessing the context menu **Script Editor > Source > Generate Element Comment**) or using the keyboard shortcut ALT+SHIFT+J (*alt+cmd+J*).



If business logic is added from Form Editor, JSDoc comment templates are automatically generated based on the context. Users can complete the comment information as needed in Script Editor.

When creating JSDoc comments, code completion is also available for the following supported tags (type '@' first, then Control-Space):

```
@AllowToRunInFind
@author {userName}
@constructor
@deprecated
@param {Type} name
@private
@return {Type}
@see
@since
@throws {Type}
@type {Type}
@version
```

A complete list of other supported tags in JSDoc is provided in the Programming Guide.

Spellchecker for Comments

Servoy features SpellChecker support for comments in the Script Editor. This feature works with singleline, multiline and JSDoc comments.