Eclipse Scout
Migration Guide

Scout Team

Version 7.0

Table of Contents

About This Document

This document describes all relevant changes **from Eclipse Scout 6.1 to Eclipse Scout 7.0**. If existing code has to be migrated, instructions are provided here.

If you are upgrading from version 6.0, please also read the migration guide for the 6.1 (*Oxygen preview*) release:
https://eclipsescout.github.io/6.1/migration-guide.html

Service Release Migration

The following changes were made after the initial 7.0 release (Eclipse Oxygen release). Additionally follow these instructions when updating to a *service release*.

Oxygen.1 (7.0.100) Release expected on September 27, 2017

Attention: The here described functionality has not yet been released and is part of an upcoming release.

- Graphic Utility Naming (since 7.0.100)
- Focus and RequestFocus merged (since 7.0.100)
- ISmartField(2) (since 7.0.100)
- Table Column Index now optional (since 7.0.100)
- AbstractLifecycle and FormLifecycle simplified (since 7.0.100)
- GroupBox: mainBox flag now set automatically (since 7.0.100)

API Changes (Java)

New Scout Modules for Jackson-based REST Services

The following new Scout modules have been added to support REST services with Jackson as marshaller:

- `org.eclipse.scout.rt.rest`
- `org.eclipse.scout.rt.rest.test`
- `org.eclipse.scout.rt.jackson`
- `org.eclipse.scout.rt.jackson.test`

In order to use REST services based on the JAX-RS Jersey implementation, the following section must be added to the project `web.xml`:

```
<!-- JAX-RS Jersey Servlet -->
<servlet>
  <servlet-name>api</servlet-name>
  <servlet-class>org.glassfish.jersey.servlet.ServletContainer</servlet-class>
  <init-param>
    <param-name>javax.ws.rs.Application</param-name>
    <param-value>org.eclipse.scout.rt.rest.RestApplication</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
```

The `RestApplication` class searches for all implementations of `IRestResource` and exposes them as REST services.

Removed Properties "enabledProcessing" on IButton and IAction

When a user performs two consecutive clicks on a button or a menu within a very short period of time, the action is executed twice. In some cases, this is an unwanted behavior. A new property "preventDoubleClick" was added to IButton and IMenu to disable it (see release notes).

In older Scout versions, two flags (`m_enabledProcessing` and `m_enabledProcessingAction`, respectively) were use to accomplish a similar objective. Whenever the click/action handler was started, the flag was set to `false`, and set back to `true` at the end of the execution. A second call to the method had no effect. However, this logic no longer works since the introduction of the HTML UI. The new UI strictly allows *only one* UI event to work on the client model simultaneously (subsequent click events are hold back by the *UiSession* lock). Consequently, the "enabledProcessing" flag is never `false`, and no clicks are prevented.

Because this construct no longer works, the corresponding properties and methods were removed, namely:

- ~~`AbstractButton: m_enabledProcessing`~~
- ~~`IButton/AbstractButton: isEnabledProcessing()`~~
- ~~`IButton/AbstractButton: setEnabledProcessing()`~~
- ~~`IButton/AbstractButton: disarm()`~~
- ~~`ButtonEvent: TYPE_DISARM`~~
- ~~`AbstractAction: m_enabledProcessingAction`~~
- ~~`IAction/AbstractAction: isEnabledProcessingAction`~~

Migration: Delete any code that references any of the removed methods. There is not replacement, as the whole concept became disfunctional with the HTML UI. To prevent accidental double clicks, use the new `getConfiguredPreventDefault()` property. If for very rare and special cases such a flag is required, it has to be implemented manually by overriding `AbstractButton.doClick()` and `AbstractAction.doAction()`.

Added HTTP Abstraction Layer: Google HTTP Client

If your application previously used a custom (not provided by Scout) implementation of the `org.eclipse.scout.rt.shared.servicetunnel.http.HttpServiceTunnel` class a migration might be necessary. For the service tunnel and a few other HTTP usages a new HTTP abstraction layer (Google HTTP Client 1.22) was introduced to support different low-level HTTP libraries (previously just the `java.net.HttpURLConnection` was used). The new default low-level HTTP library is the Apache HTTP Client 4.5.3, more details are available in the release notes. As the `java.net` connections are not used anymore by default the MultiSessionCookieStore (for `java.net` connections) is not installed by default (could be installed manually).

For the `HttpServiceTunnel`class the method `URLConnection createURLConnection(ServiceTunnelRequest call, byte[] callData)` has been replaced by `HttpResponse executeRequest(ServiceTunnelRequest call, byte[] callData)`, also several methods (`addCustomHeaders`, `addSignatureHeader`, `addCorrelationId`, `createAuthToken` and `interceptHttpResponse`) signatures have changed their `URLConnection` parameter to either a `HttpRequest` or `HttpResponse` parameter. The same migration applies for subclasses of the `org.eclipse.scout.rt.server.commons.servlet.HttpProxy` class.

IGroupBox: Changed Behavior of "getGridColumnCount"

The function `getGridColumnCount()` returned the calculated column count of the grid. This has been changed so that it returns the configured one which is not necessarily the same.

Example: the default of the property `gridColumnCount` is -1 which will typically result in 2 columns.

So in case you used this method, you should consider calling `getFieldGrid().getGridColumnCount()` instead.

ScoutTexts

Removed support for session scope specific `ScoutTexts` instances because support of scoped services was removed in Scout without OSGi (version >= 5.0).

Changes:

- `ScoutTexts` is now an application-scoped bean
- `ScoutTexts.CURRENT` thread local removed without replacement (use `BEANS.get(ScoutTexts.class)` instead)

Deprecations (methods will be removed in P-release):

- `ScoutTexts.get` → `TEXTS.get`
- `ScoutTexts.getInstance()` → `BEANS.get(ScoutTexts.class)`
- `ISession.getTexts()` → `BEANS.get(ScoutTexts.class)`

- `AbstractSqlService.getConfiguredNlsProvider()` → method is not used anymore (returns `null`)
- `AbstractSqlService.getNlsProvider()` → method is not used anymore (returns `null`)

ISmartField(2) (since 7.0.100)

The following methods have been removed from `ISmartField(2)`:

- `String getBrowseNewText()`
- `setBrowseNewText(String s)`
- `void doBrowseNew(String newText)`

The following methods have been removed from `AbstractSmartField(2)`:

- `protected ILookupRow<VALUE> execBrowseNew(String searchText)`
- `protected String getConfiguredBrowseNewText()`

Note: with 7.0 these methods still exist on the old smart field `ISmartField` but are marked as deprecated.

Deprecated property `focusable` from IFormField, AbstractFormField

Since the new Html UI was introduced with Scout 5.2 this property had no effect on the UI anymore. Instead the UI uses sensible defaults for each field type. For instance: a LabelField is never focusable, a normal StringField is always focusable, as long as it is enabled. Since the property was rarely used, we deprecated all methods from the interface/class FormField. The following methods/properties will be removed with Scout 7.1:

- boolean IFormField#isFocusable()
- void IFormField#setFocusable(boolean f)
- boolean IFormField#PROP_FOCUSABLE
- boolean AbstractFormField#getConfiguredFocusable()
- class AbstractNonFocusableButton
- class AbstractNonFocusableRadioButton

You should no longer use these methods. Since there will be no replacement in later Scout releases, you should delete code that uses these methods/properties.

API Changes (JavaScript)

render()

The parameter `$parent` has been removed from the `_render` method because `this.$parent` is

available for every widget. There is no need to have a parameter `$parent` which points to the same variable. Use `this.$parent` instead.

Also `$parent` is now optional when calling widget.render(). The `$parent` may be resolved using `this.parent`. No need to always write `widget.render(this.$container)` anymore, instead just write `widget.render()` if the `$container` of the `parent` should be used as `$parent`.

Promises

With jQuery 3 the promise API is now Promises/A+ compliant. This means you may need to adjust your code if you use promises.

We noticed the following effects:

- If a rejection is catched using a fail handler, the fail handler has to return a rejected promise as well, otherwise the next success handler would be called instead of the next fail handler.
- Every callback is now executed asynchronously. This is especially relevant for the tests.
- Catch has been added → replace fail(null, func) for better readability.

See also https://jquery.com/upgrade-guide/3.0/ for details.

Removed addClassSVG, removeClassSVG, attrSVG, removeAttrSVG

These functions are now supported by jQuery directly. Just use `addClass`, `removeClass`, `attr` and `removeAttr`.

Property Change Event

The property change event has been simplified.

The event had 3 properties:

- ~~`newProperties`~~
- ~~`oldProperties`~~
- ~~`changedProperties`~~

This was added to be able to react to multiple property change events at once. Since 6.1, bulk property changes don't exist anymore, so there is no need for these properties anymore.

Now, with 7.0, the property change event has the following properties:

- `propertyName`
- `oldValue`
- `newValue`

This makes handling the event easier. Check your propertyChange event handlers and adjust them accordingly.

Logical Grid Validation

Automatic Grid Data Validation has been introduced. This means there is no need to manually create a Logical Grid (e.g. `VerticalSmartGroupBoxBodyGrid` or `HorizontalGroupBoxBodyGrid` and validate it anymore, this will be done by the `LogicalGridLayout` itself. Also, check your JSON files, remove any explicit x, y grid definitions because they will be calculated by the `LogicalGrid`. Make sure to always use the property `gridDataHints` instead of `gridData`.

Event Naming

The naming of the events has been harmonized to conform with the event naming guide. This is only relevant, if you attached listeners using JavaScript or if you do some kind of load testing using the events in the requests.

The following changes have been made:

- Rename `doAction` to `action`
- Rename `linkPageWithRow` to `pageRowLink`
- Rename `initPage` to `pageInit`
- Rename `exportToClipboard` to `clipboardExport`
- Rename `parseerror` to `parseError`
- Rename `selectionChanged` to `selectionChange`
- Rename `callAction` to `action`
- Remove `insertText`
- Rename `displayTextChanged` to `acceptInput`
- Rename `popupopen` to `popupOpen`
- Rename `locationChanged` to `locationChange`
- Rename `sessionready` to `sessionReady`
- Rename `desktopcreated` to `desktopReady`
- Rename `positionChanged` to `positionChange`
- Rename `scrollstart` to `scrollStart`
- Rename `scrollend` to `scrollEnd`
- Rename `clicked` to `click`
- Rename `modelChanged` to `modelChange`
- Rename `selectionChanged` to `selectionChange`
- Rename `viewRangeChanged` to `viewRangeChange`
- Rename `formActivated` to `formActivate`
- Rename `historyEntryActivated` to `historyEntryActivate`
- Rename `viewAdded` to `viewAdd`

- Rename `viewRemoved` to `viewRemove`
- Rename `viewActivated` to `viewActivate`
- Rename `viewDeactivated` to `viewDeactivate`
- Rename `tabClicked` to `click`
- Rename `tabSelected` to `tabSelect`
- Rename `nodeClicked` to `nodeClick`
- Rename `rowClicked` to `rowClick`
- Rename `rowsSorted` to `sort`
- Remove `sortRows`
- Rename `rowsGrouped` to `group`
- Remove `groupRows`
- Rename `exportToClipboard` to `clipboardExport`
- Rename `rowsFiltered` to `filter`
- Rename `addFilter` to `filterAdded`
- Rename `removeFilter` to `filterRemoved`
- Rename `filterResetted` to `filterReset`
- Remove `groupingChanged`

Graphic Utility Naming (since 7.0.100)

The naming of the functions of `scout.graphics` and `scout.HtmlComponent` has been harmonized. Also, they now consistently use an `options` parameter.

The following changes have been made:

scout.graphics:

- Rename `getMargins` to `margins`
- Rename `getInsets` to `insets`
- Rename `getSize` to `size`
- Remove `getBounds`

scout.HtmlComponent:

- Rename `getMargins` to `margins`
- Rename `getInsets` to `insets`
- Rename `getSize` to `size`
- Rename `getPreferredSize` to `prefSize`
- Rename `getAvailableSize` to `availableSize`

- Remove `getBounds`

The old methods still exists but are marked as deprecated. Note that `getBounds` does not include margins anymore, and `bounds` and `offsetBounds` now take an `options` object instead of 2 boolean parameters.

Focus and RequestFocus merged (since 7.0.100)

`scout.Widget` had a function called `requestFocus`. Some concrete widgets provided a function named `focus`. Because there is no need to have two methods doing the same, these functions have been merged. `scout.Widget` now provides a function `focus`. `RequestFocus` has been deprecated.

Table Column Index now optional (since 7.0.100)

When creating a table using JavaScript and `scout.create('Table')`, specifying the index for each column has been necessary. This has changed, the column indices are now set automatically based on the order of the columns. Just remove the index settings from your JSON files.

AbstractLifecycle and FormLifecycle simplified (since 7.0.100)

The life cycle has been reworked and simplified:

- Renamed `AbstractLifecycle` to `Lifecycle`.
- Installed form life cycle by default and added delegating functions to the form to make it easier to use.
- Implemented reset.
- Added default callbacks on form which may be implemented by subclasses: `_load`, `_save`, `importData`, `exportData`
- Removed `doFinally`, because it is not needed yet.
- Renamed `disposeWidget` to `close`.
- Removed the do prefix from all the life cycle functions.
- Added system menus (`OkMenu`, `CancelMenu` etc.).

To migrate: check your forms, remove the life cycle, use the new callback functions and system menus if needed, adjust your custom life cycles.

GroupBox: mainBox flag now set automatically (since 7.0.100)

Every form needs a root group box. That box has the flag mainBox set to true. Until now, this has to be set explicitly. From now on, you can omit it. To migrate just remove the mainBox flags.

Charts: fixed typo in function name

Incorrect function name `handelToBigLabels` in AbstractGridChartRenderer.js was renamed to `handleTooBigLabels`. You must apply this renaming to your code if this function is called or overridden.

Other Changes

Maven "provided" Dependencies

In Maven dependencies with the scope provided are not transitive. This makes sense if a dependency is set to provided depending on the environment. Any artifacts that are not intended to be used in a certain environment should not have the scope provided and are therefore now transitive. We removed any current dependency `javax.servlet:javax.servlet-api` except for the one in the artifact `org.eclipse.scout.rt.server.commons`.

To migrate your project, remove any dependency to `javax.servlet:javax.servlet-api`, `javax.xml.ws:jaxws-api` or `javax.ws.rs:javax.ws.rs-api`. Then add to all artifacts with packaging type war the dependency to `javax.servlet:javax.servlet-api` with scope provided. Depending on the container, you may want also to add the depdendency `javax.xml.ws:jaxws-api` with scope provided to the war artifact.

```
<project>
  <artifactId>myproject.server.war</artifactId>
  <packaging>war</packaging>

  <dependencies>
    <dependency>
      <groupId>myproject</groupId>
      <artifactId>myproject.server</artifactId>
    </dependency>

    <!-- provided by container -->
    <dependency>
      <groupId>javax.servlet</groupId>
      <artifactId>javax.servlet-api</artifactId>
      <scope>provided</scope>
    </dependency>
  </dependencies>
</project>
```

Do you want to improve this document? Have a look at the sources on GitHub.