Eclipse Scout

Release Notes

Scout Team

Version 10.0

Table of Contents

About This Release . . . 2
- Service Releases . . . 2
- Obtaining the Latest Version . . . 2
- Demo Applications . . . 3

IntelliJ IDEA Support . . . 4

New Web Tool Stack . . . 5

Widget Enhancements . . . 6
- Table: Visualize TableRows as Tiles . . . 6
- GroupBox: New MenuBar Position TITLE . . . 6
- TabBox: New Tab Style . . . 6
- TileGrid Groups: Show Loading State per Group . . . 7
- RadioButton and CheckBox: Wrap Text . . . 7
- FormField: Support for HTML Labels . . . 8
- Button: Support for HTML and Binary Resources . . . 8
- LabelField: Support for App Links . . . 8
- ImageField: Support for File Upload . . . 8
- Mode Selector: Alternative Display Style . . . 8
- Popup: New Properties Closable, Movable, Resizable . . . 8
- Page: Tile Overview . . . 9
- Enabled & InheritAccessibility . . . 9
- Consistent Parent . . . 10
- PropertyChange Shortcut (JS) . . . 10

NLS File Watcher . . . 11

This document is referring to a past Scout release. Please click here for the recent version.
Looking for something else? Visit https://eclipsescout.github.io for all Scout related documentation.

About This Release

The latest version of this release is: 10.0.45.

You can see the detailed change log on GitHub.

Coming from an older Scout version? Check out the Migration Guide!

Service Releases

Scout 10.0 will continue to be maintained for a while and a new build may be released from time to time. Beside bug fixes, these releases may even contain some minor features.

Obtaining the Latest Version

Scout Runtime for Java

Scout Runtime artifacts for Java are distributed using Maven Central:

- 10.0.45 on *Maven Central*
- 10.0.45 on *mvnrepository.com*

Usage example in the parent POM of your Scout application:

```
<dependency>
    <groupId>org.eclipse.scout.rt</groupId>
    <artifactId>org.eclipse.scout.rt</artifactId>
    <version>10.0.45</version>
    <type>pom</type>
    <scope>import</scope>
</dependency>
```

Scout Runtime for JavaScript

Scout Runtime artifacts for JavaScript are distributed using npm:

- Scout Core Runtime
- All official Scout JavaScript packages

IDE Tooling (Scout SDK)

In addition to the Eclipse IDE Scout 10 now officially supports IntelliJ IDEA as well

Please find more details about the new plugin and how to install it in the corresponding section below.

Eclipse

You can download the complete Eclipse IDE with Scout SDK included (EPP) here:
Eclipse for Scout Developers

To install the Scout SDK into your existing Eclipse IDE, use this update site:
http://download.eclipse.org/scout/releases/10.0/

IntelliJ IDEA

You can download the Scout plugin for IntelliJ IDEA from the JetBrains Plugin Repository or you can use the plugins client built into IntelliJ IDEA. Please refer to the IntelliJ Help on how to install and manage plugins.

Demo Applications

The demo applications for this version can be found on the features/version/10.0.45 branch of our docs repository on GitHub.

If you just want to play around with them without looking at the source code, you can always use the deployed versions:

- https://scout.bsi-software.com/contacts/
- https://scout.bsi-software.com/widgets/
- https://scout.bsi-software.com/jswidgets/

IntelliJ IDEA Support

We are happy to announce Scout support for IntelliJ IDEA which combines the power of IntelliJ and Scout and gives you more flexibility in choosing your favourite development environment.

However as this is the very first version of the IntelliJ plugin not all features that are available in Eclipse are available in IntelliJ yet. We are working towards closing the gaps and bring your favourite features to IntelliJ as well. The features supported by the latest version can bee seen in the `What's New` section of the plugin description. More features will come in future releases as soon as possible.

New Web Tool Stack

Scout updated the web tool stack from a custom implementation to a standard Node.js, Webpack and Babel based setup. This was required as some of the frameworks used until now (e.g. to minimize CSS and JavaScript code or to run JavaScript tests) cannot handle modern code and are no longer supported (end-of-life).

Thanks to the power of the new tool stack it is now possible to write modern JavaScript code which will be transpiled at build time so that old browsers can execute the code. Furthermore it is easier to integrate and update 3rd party JavaScript frameworks available in the huge npm registry.

The old Scout JavaScript code has already been migrated to a modern EcmaScript 6+ code level and uses these new features. Because of this your project specific JavaScript code must be migrated to the new level as well. Please refer to the Technical Guide for details about the new tool stack and to the Migration Guide on how to update your project.

Widget Enhancements

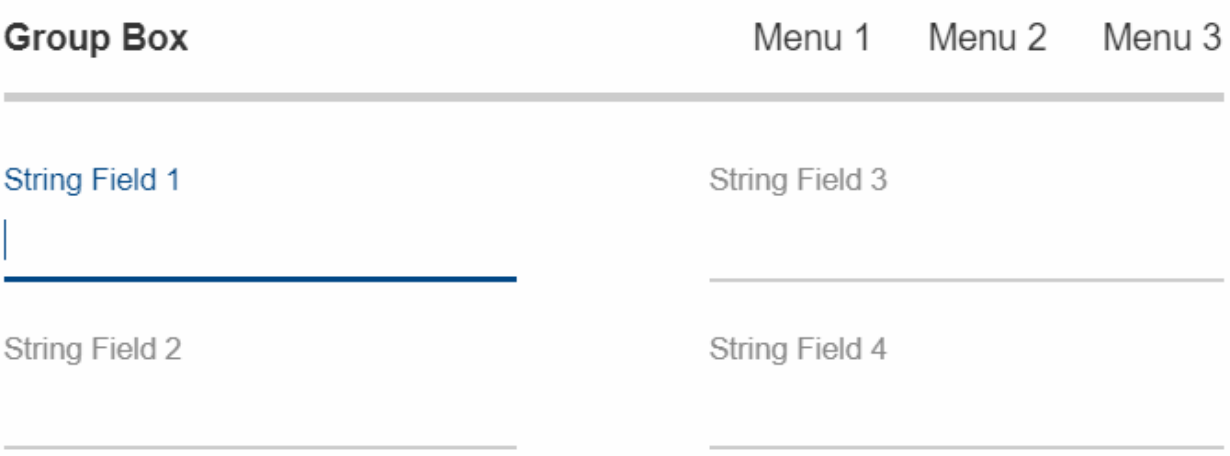
Table: Visualize TableRows as Tiles

The table has now a new way to visualize it's data: the tile mode. The property `tileMode` in `AbstractTable` controls which visualization is used. To use this feature, your table needs to implement `execCreateTile(ITableRow row)`.

When in tile mode, the Table uses a `TileGrid` to visualize it's tiles. All relevant events (selection, action, ...) and properties (selectable, filter, ...) are synchronized between the table and it's internal tileGrid.

GroupBox: New MenuBar Position TITLE

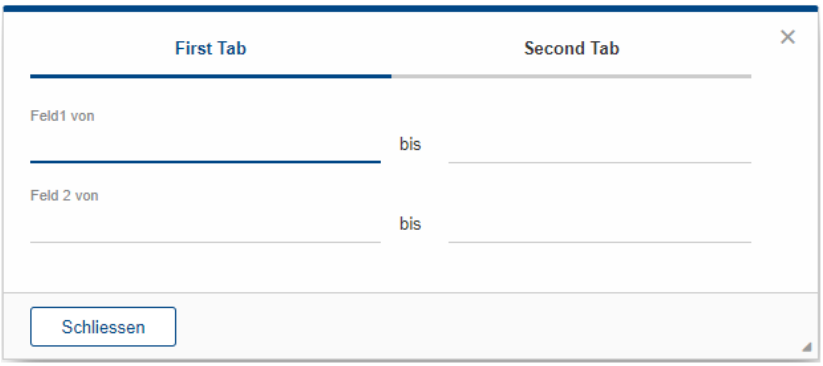
In addition to the existing menu-bar position TOP and BOTTOM, the GroupBox now supports TITLE. The menu-bar is placed in the header of the group-box right of the title-label DIV. Scout Classic: see property `menuBarPosition` in `AbstractGroupBox` and constant `IGroupBox.MENU_BAR_POSITION_TITLE`. Scout JS: see property `menuBarPosition` in `GroupBox.js` and constant `scout.GroupBox.MenuBarPosition.TITLE`.



TabBox: New Tab Style

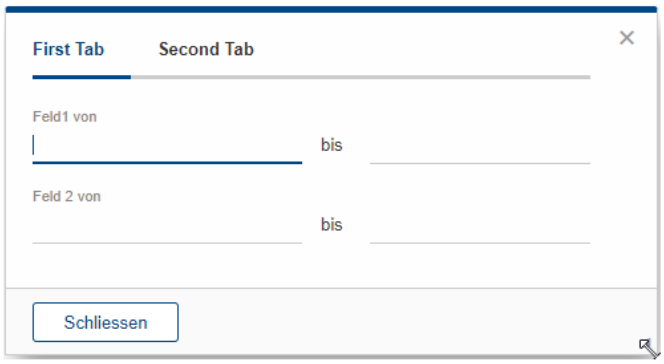
The tab box has a new property `ITabBox.PROP_TAB_AREA_STYLE` with currently two possible tab styles.

- `ITabBox.TAB_AREA_STYLE_DEFAULT`: Default appearance
- `ITabBox.TAB_AREA_STYLE_SPREAD_EVEN`: This style will spread all the tabs over the available space. When using this style, all the menu will be pushed into the ellipse menu.



TAB_AREA_STYLE_SPREAD_EVEN

TAB_AREA_STYLE_DEFAULT

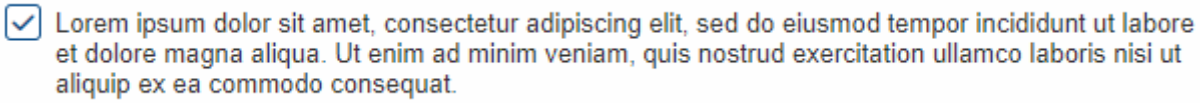


TileGrid Groups: Show Loading State per Group

Each Group in a TileGrid can now display a loading indicator in the group header individually. This is useful if each displayed group loads data from an individual data source. Scout Classic: call `AbstractGroup#setLoading(boolean)`, Scout JS: call `Group.js#setLoading(boolean)`. Note: it is still possible to set the loading state on the TileGrid, to indicate the whole grid (and not an individual group) is loading data.

RadioButton and CheckBox: Wrap Text

Both widgets `RadioButton` and `CheckBox` (aka BooleanField), now support the `wrapText` property. This means a radio button or a check box can have a label with a long text on multiple lines. In order to see the wrapped text, the field must have a `gridH` > 1 or set the `gridUseUiHeight` property to true.



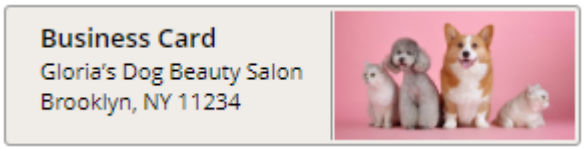
FormField: Support for HTML Labels

The `FormField` has a new property `labelHtmlEnabled`. When set to true, the label is rendered as HTML instead of plain text. Since the label is also used to render buttons, check-boxes and radio-buttons these widgets are now HTML capable too.

Button: Support for HTML and Binary Resources

Since `Button` is a FormField too, you can now use HTML in the label-part of the button. You can even reference binary resources in your HTML. Simply call the method `AbstractButton#setAttachments(Collection<? extends BinaryResource>)` and define a reference, say an image URL, in your label HTML:

```
<img src="binaryResource:business-card.jpg" />
```



LabelField: Support for App Links

The `LabelField` now supports app links. In order to use app links in a label field, the property `htmlEnabled` must be set to true. This property affects the value-part of the LabelField, whereas `labelHtmlEnabled` inherited from FormField affects the label-part.

ImageField: Support for File Upload

The `ImageField` has a new property `uploadEnabled`. When set to true, the field opens the native file chooser and performs a file upload.

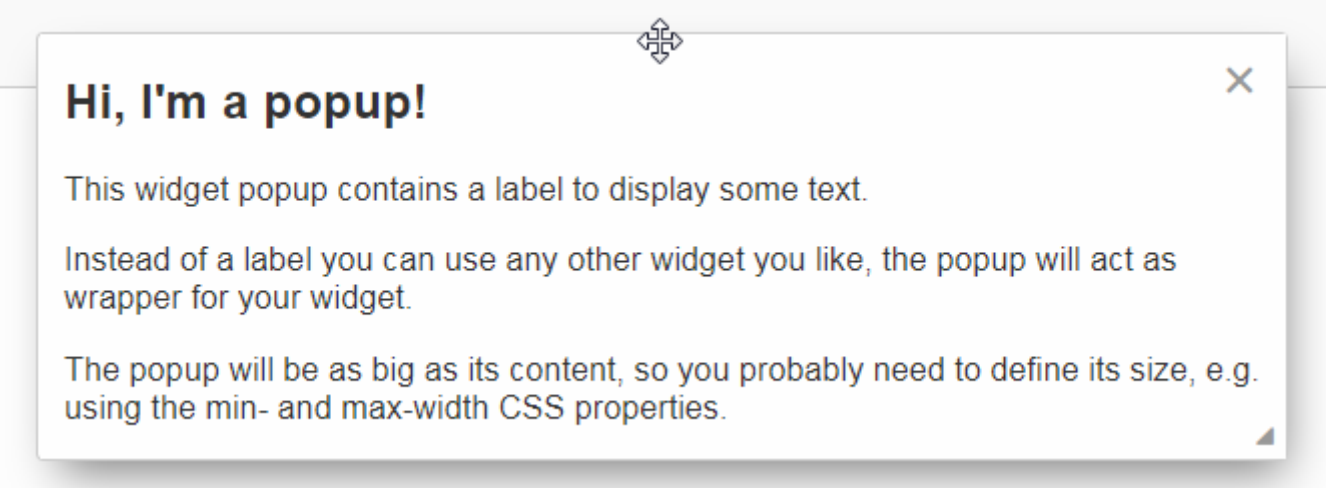
Mode Selector: Alternative Display Style

The mode selector now has a different appearance for the two field styles `IFormField.FIELD_STYLE_CLASSIC` and `IFormField.FIELD_STYLE_ALTERNATIVE`.

Popup: New Properties Closable, Movable, Resizable

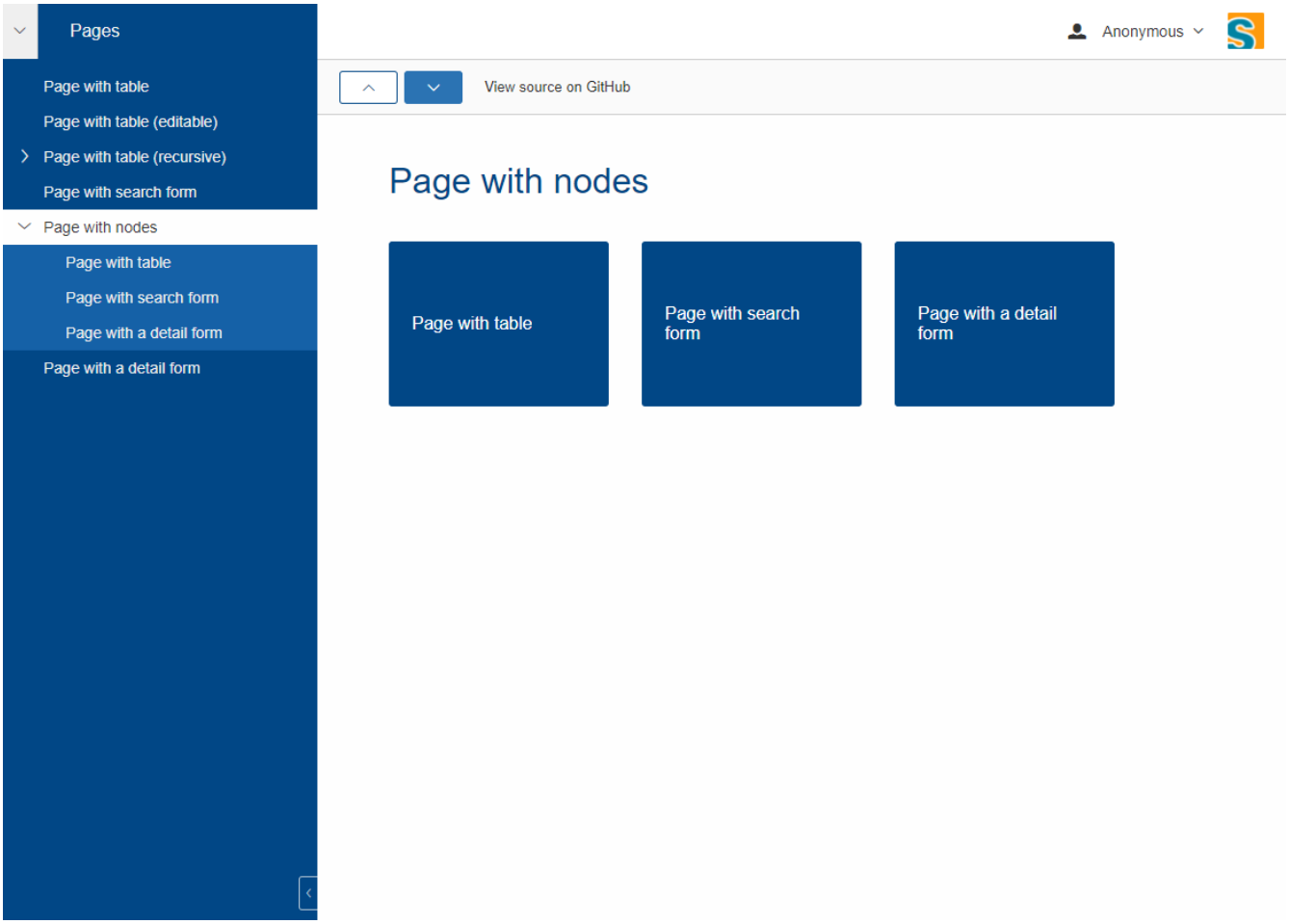
The new properties allow a user to close, move or resize a popup using the typical controls he already knows by other widgets like a `Dialog`. The properties are available on the `WidgetPopup` and

can even be turned on or off while the popup is open.



Page: Tile Overview

The tile overview known from the outline root is now available for all pages. It can can be enabled via the `AbstractPage#getConfiguredShowTileOverview()` property.



Enabled & InheritAccessibility

The `enabled` property already existing for FormFields, Actions and other Scout elements has been moved to the top level Widget class. Therefore all widgets can now be disabled as it already was in

Scout JS. Additionally Scout Classic widgets also support dynamic enabled/disabled dimensions on all widgets now as it already existed for FormFields.

The `inheritAccessibility` property that already existed for Actions (e.g. Menus) can now be used on all widgets. This is especially interesting for FormFields for which it was necessary to do that using multiple setEnabled() calls until now.

Consistent Parent

Lots of Scout widgets like `Actions`, `FormFields`, `Tiles` or `Tables` can have a parent element. Until now there have been several methods to access parent elements (e.g. `IActionNode#getParent()`, `ITile#getContainer()`, `IFormField#getParentField()`, etc.) and it was only available on the specific element (e.g. on an Action).

Now there is one `getParent()` method for all widgets. Furthermore visit methods have been added to traverse up the parent hierarchy.

PropertyChange Shortcut (JS)

It is now possible to listen for specific property changes rather than listening to all property change events and then checking for the right property manually. This can be done using the new notation `propertyChange:propertyName`, see the example below.

Listing 1. Listen for specific property changes

```
field.on('propertyChange:value', function(event) {
  // This listener is only executed when the 'value' property changes
  console.log('Property ' + event.propertyName + ' changed from ' + event.oldValue + '
to ' + event.newValue);
});
field.setValue('New Value.');
```

NLS File Watcher

The new `NlsFileWatcher` observes changes in text property files. If a change in a text property file occurs, the nls resource bundle cache will be invalidated. This means that there is no need to restart the server when working on texts and translations. The file watcher is only active if `scout.devMode=true` is set and can be disabled with `scout.dev.texts.fileWatcherEnabled=false`. By default, the file watcher is enabled in development mode.

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