

participants

Metawidgets along with config

Inspector - with config

layout - for the widget

simple use case SwingMetawidget- setToInspect(object)

# Processing pipeline

1. setToInspect is called to set the Object for inspection. The user typically calls this method, either directly or through

some intermediate language (e.g. using a JSP attribute).

2. internally, buildWidgets is called to begin the process. It first calls Inspector.inspect to return inspection results,

then InspectionResultProcessor.processInspectionResult to process (i.e. sort) them.

3. buildWidgets calls WidgetBuilder.buildWidget to choose a suitable widget for the top-level element of the inspection result (based on its @type and other attributes). If WidgetBuilder.buildWidget returns such a widget,

skips to 6.

4. if WidgetBuilder.buildWidget returns null for the top-level element, calls buildCompoundWidget to iterate over

each child of the top-level element.

5. for each child, calls WidgetBuilder.buildWidget and adds the returned widget to the Metawidget. If

WidgetBuilder.buildWidget returns null for a child, creates a nested Metawidget.

6. the created widgets are passed through a series of WidgetProcessors. These can apply binding mechanisms, validators

and so on.

7. as a final step, the created widgets are passed to a Layout.

JpaInspector

jspInspectionResultProcessor

comesAfterInspectionResultProcessor

overriddenWidgetBuilder

readOnlyWidgetBuilder

springWidgetBuilder

HtmlWidgetBuilder

WidgetProcessor

Layouts

**void** layoutWidget(W widget,String elementName,Map<String,String> attributes,C container,M metawidget);

StaticMetawidget