Dynamic Forms in UI Configuration

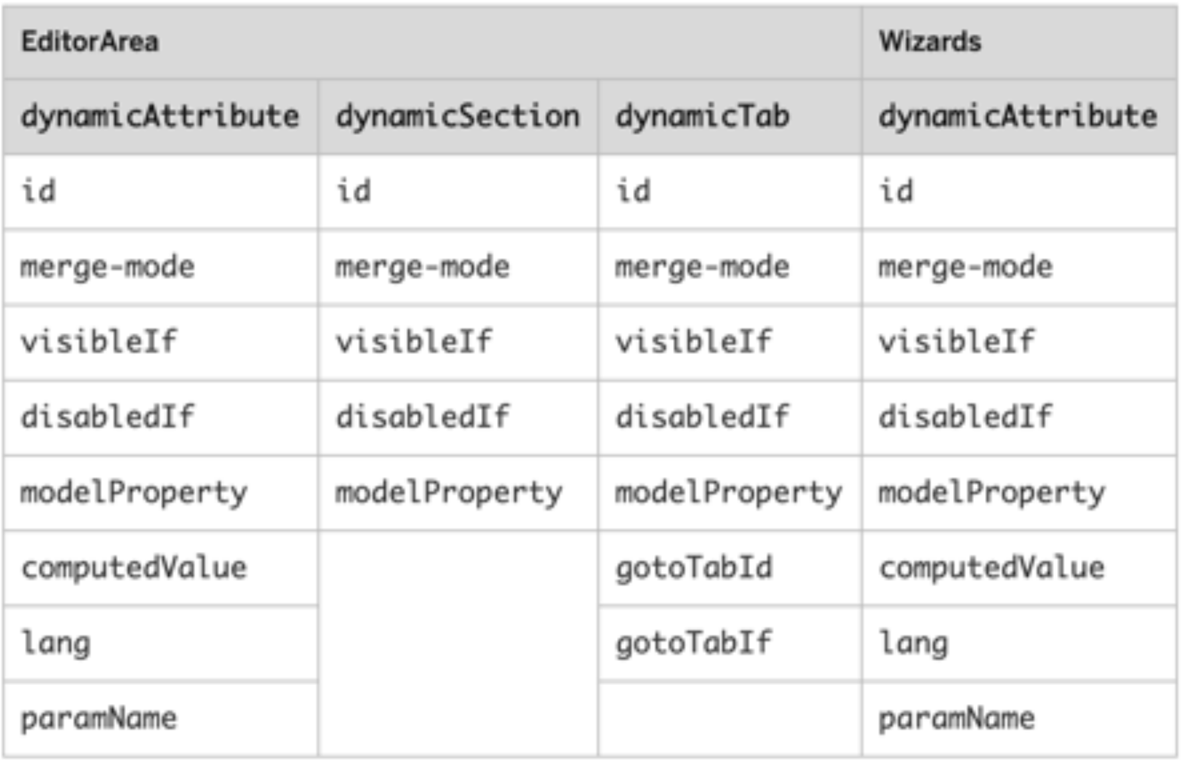
- The dynamic forms functionality was introduced to provide more flexibility in the Backoffice Application interface.
- Dynamic forms can be configured in the Editor Area widget and in wizards to allow some **custom logic being** added to the UI/UX in the Backoffice.

Overview

- Dynamic Forms functionality enable performing dynamic actions using scripting languages on **attributes**, **sections** and **tabs** when the chosen model property changes.
- In this document, we present the examples of how the dynamic forms can be implemented in the configuration of the Editor Area widget and in wizards.

Configuration

To implement dynamic actions in the user interface, depending on the effect we want to achieve, we need to define any (or all) of the following elements: dynamicAttribute, dynamicSection or dynamicTab in the configuration files. The available attributes for dynamic elements are presented in the following table:

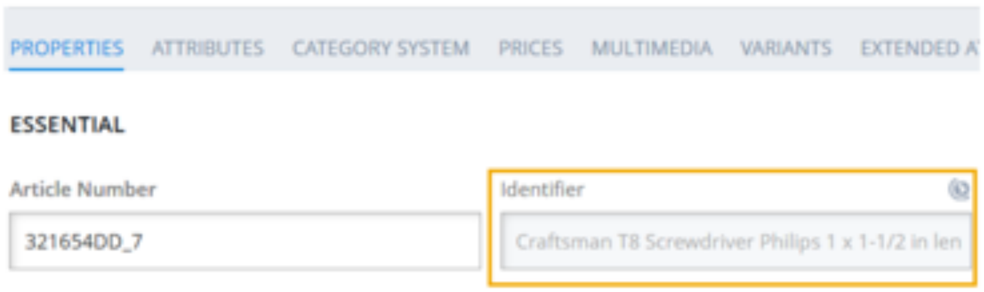


EditorArea			**Wizards**
dynamicAttribute	**dynamicSection**	**dynamicTab**	**dynamicAttribute**
id	id	id	id
merge-mode	merge-mode	merge-mode	merge-mode
visibleIf	visibleIf	visibleIf	visibleIf
disabledIf	disabledIf	disabledIf	disabledIf
modelProperty	modelProperty	modelProperty	modelProperty
computedValue		gotoTabId	computedValue
lang		gotoTabIf	lang
paramName			paramName

A. Attribute

1.Dynamic Attribute

The following figure shows the result of this configuration:



Article Number: 321654DD_7

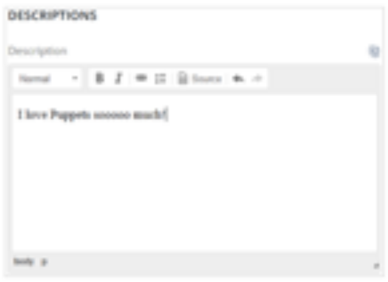
Identifier: Craftsman T8 Screwdriver Philips 1 x 1-1/2 in len

Figure: Disabled Identifier field upon changing the Article Number (code) to have the length greater than 5.

```
<context type="Product" component="editorAreaDynamicForms" merge-by="module">
        <df:dynamicForms xmlns:df="http://www.hybris.com/cockpitng/component/dynamicForms">
                <df:attribute id="uniqueId" triggeredOn="code" qualifier="name"
disabledIf="code.length() &gt; 5" />
        </df:dynamicForms>
</context>
```

2.Dynamic lang Attribute

As a result of the above configuration in the cockpit-config.xml file, upon changing the Identifier's name of a Product, we get the Description field for Colombian Spanish and German languages dynamically updated to display the 'I love Puppets soooooo much!' where 'Puppets' is the

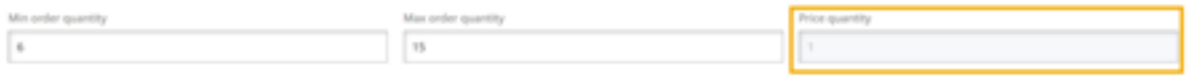


I love Puppets soooooo much!

Figure: Dynamic lang attribute upon changing the Identifier's name.

```
<context type="Product" component="editorAreaDynamicForms" merge-by="module">
        <df:dynamicForms xmlns:df="http://www.hybris.com/cockpitng/component/dynamicForms">
                <df:attribute id="uniqueId" qualifier="description" lang="de,es_CO"
triggeredOn="name" computedValue="'I love ' + name[new java.util.Locale('en')] + ' sooooo much!
'"/>
        </df:dynamicForms>
</context>
```

3.triggeredOn="*" Attribute
This example depicts triggeredOn="*", meaning that dynamicForm's action is triggered by any model change:



Min order quantity	Max order quantity	Price quantity
6	15	1

Figure: Field disabled when the specified values are entered.

```
<context type="Product" component="editorAreaDynamicForms" merge-by="module">
        <df:dynamicForms xmlns:df="http://www.hybris.com/cockpitng/component/dynamicForms">
                <df:attribute id="uniqueId" triggeredOn="*" qualifier="priceQuantity"
disabledIf="(minOrderQuantity &gt; 5 || maxOrderQuantity &lt; 15)" />
        </df:dynamicForms>
</context>
```

As a result, the Price Quantity field gets disabled if any of the specified fields meets the criteria: the Minimum Order Quantity value is greater than 5 or the Maximum Order Quantity value is smaller than 15:

4.paramName Attribute

Dynamic attribute, both in Editor Area and Wizards, can have paramName attribute specified, that makes the computed value to be added as a param to editors of a specified qualifier.

This feature can be handful when there is a dependency between editors, for example, to filter the values available in one editor based on values in another editor. In this case, we would use the feature of reference editors.

Let's suppose that we have the following wizard configuration

```
<context merge-by="type" parent="AbstractConstraint" type="AttributeConstraint"
component="create-wizard">
<wz:flow xmlns:wz="http://www.hybris.com/cockpitng/config/wizard-config"
id="AttributeConstraintWizard" title="create.title(ctx.TYPE_CODE)">
  <wz:prepare id="itemPrepare">
    <wz:initialize property="attrDescCtx" type="java.util.HashMap"/>
    <wz:assign property="attrDescCtx.composedType" value="null"/>
  </wz:prepare>
  <wz:step id="step1" label="flow.allmanadatory">
    <wz:content id="step1.content">
      <wz:property position="49" qualifier="descriptorEnclosingType"
type="Reference(ComposedType)" validate="false"/>
      <wz:property position="50" qualifier="newObject.descriptor"/>
      <wz:property position="200" qualifier="newObject.languages"/>
    </wz:content>
  </wz:step>
</wz:flow>
</context>
```

In this case, we would like to filter the available descriptors, based of the chosen composed type (descriptorEnclosingType). To achieve this dependency, we can define the following rule:

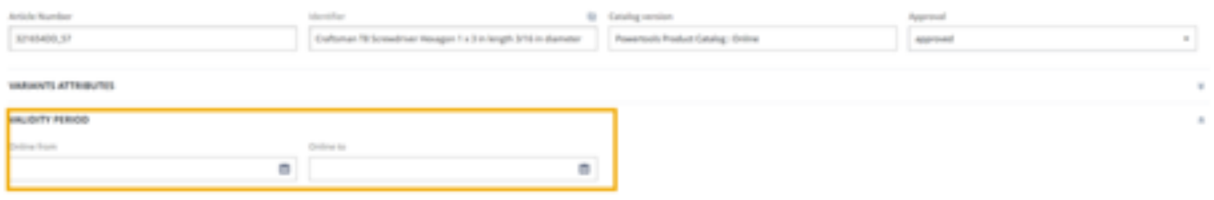
```
<context type="AttributeConstraint" component="configurableFlowDynamicForms"
module="platformbackoffice">
        <df:dynamicForms modelProperty="newObject">
                <df:attribute id="dynamicDescriptor" disabledIf="descriptorEnclosingType==null"
paramName="referenceSearchCondition_enclosingType"
computedValue="descriptorEnclosingType!=null ? descriptorEnclosingType.pk.toString():null"
modelProperty="*" qualifier="newObject.descriptor" triggeredOn="descriptorEnclosingType"/>
        </df:dynamicForms>
</context>
```

With the above configuration, the descriptor property in a wizard would be disabled until the composed type is chosen, and the list of available references would be limited to descriptors which are contained by the chosen type.

B.Dynamic Section

Adding a specific dynamic section.

The following snippet shows how to get the section being added when the specified condition is fulfilled.As a result, the Validity Period section appeared:


Article Number: 32165400_57
Identifier: Craftsman T8 Screwdriver Hexagon 1 x 3 in length 3/16 in diameter
Catalog version: Powertools Product Catalog : Online
Approval: approved

VARIANTS ATTRIBUTES

VALIDITY PERIOD
Online from
Online to

Figure: Validity Period section enabled upon code change.

```
<context type="Product" component="editorAreaDynamicForms" merge-by="module">
  <df:dynamicForms xmlns:df="http://www.hybris.com/cockpitng/component/dynamicForms">
    <df:section id="uniqueId" triggeredOn="code" qualifier="hmc.section.catalog_and_validity"
visibleIf="code.length() &gt; 5" />
  </df:dynamicForms>
</context>
```

The above example is similar to the previous one, but it uses <section> tag instead of <attribute> and it uses visibleIf="..." action instead of disabledIf="...".

C.Dynamic Tab

The below snippet shows how to configure the action of switching tabs upon a certain change:

```
<context type="Product" component="editorAreaDynamicForms" merge-by="module">
  <df:dynamicForms xmlns:df="http://www.hybris.com/cockpitng/component/dynamicForms">
    <df:tab id="uniqueId" triggeredOn="code" gotoTabIf="code.length() &gt; 5"
gotoTabId="hmc.tab.product.multimedia"  />
  </df:dynamicForms>
</context>
```

This time, upon changing the Article Number, by using <tab> tag and our action gotoTabIf="...", the selected tab would be switched to the tab defined in the gotoTabId="..." attribute. Note that in this example we are not declaring the qualifier="..." attribute, because we are not amending any elements, but only changing the selected tab.

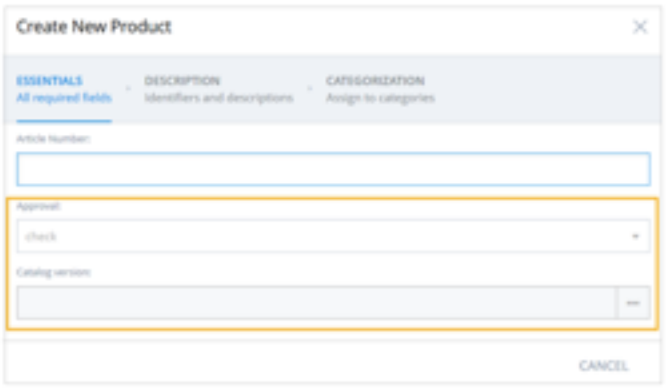
D.Dynamic Wizard

1.It is possible to set the dynamic forms for wizards configuration.
2.Contrary to the EditorArea, in wizards by default only the **df:attribute** is supported (no tabs or sections). The following snippet shows the example configuration for wizards:

```
<context type="Product"  component="configurableFlowDynamicForms" merge-by="module">
          <df:dynamicForms xmlns:df="http://www.hybris.com/cockpitng/component/dynamicForms"
modelProperty="newProduct">
                    <df:attribute id="a" disabledIf="code==null || code.length() &lt; 2"
qualifier="catalogVersion" triggeredOn="*" />
                    <df:attribute id="b" disabledIf="code==null || code.length() &lt; 2"
qualifier="approvalStatus" triggeredOn="*" />
          </df:dynamicForms>
</context>
```

3.This time, instead of modifying behavior of editorArea, by setting component="configurableFlowDynamicForms" we declare that we would be defining actions for create-wizard and we wish to disable catalogVersion and approvalStatus fields if the code is null or its length is shorter than 2 characters.
4.Additionally, model-property= has to be set to newProduct. Please notice that in the case of configurableFlows, we have to explicitly cover null scenarios like in code==null.

See the result of this configuration in the following figure:



Article Number:

Approval: check

Catalog version:

Figure: Catalog Version and Approval Status disabled as the Code does not have enough characters.