ITC INFOTECH

Business-friendly Solutions

WINDCHILL CUSTOMIZATION

AND

CODE SNIPPET

CREATED BY: KAUSHIK DAS

DOCUMENT CONTROL

Document History

Issue	Date	Maintainer/ Owner	Description
A.1	12th April, 2013	Kaushik Das	Add code-snippet section
A.2	15th April,2013	Kaushik Das	Add the rule algorithm part and change the code for dropdown list ,add the code for multi-valued attribute
B.1	10th May, 2013	Kaushik Das	Change the Format of the document, Add the Wizard processing section and modify all other previous sections
B.2	21st June, 2013	Kaushik Das	Add more Code snippets
B.3	09th July, 2013	Kaushik Das	Add service listener and code snippets
B.4	23rd Aug, 2013	Kaushik Das	Table Builder
B.5	25th Nov, 2013	Kaushik Das	Added more code snippet. Custom Dropdown list for workflow variable. More example and explanation in Picker section.
B.6	10th Dec,2013	Kaushik Das	Add more examples in Rule Algorithm, Data Utility, Validator section
B.7	23rd Dec,2013	Kaushik Das	Add more examples for Listener and Table Builder
B.8	09th Jan, 2014	Kaushik Das	Add Example in Table Builder, Data utility
B.9	07th Feb,2014	Kaushik Das	Add more snippets as well as fix some previous problems , Example added in TableBuilder and miscellaneous section
B.10	28th May, 2014	Kaushik Das	Add another section related to Sonar Rule, Add more requirement in Miscellaneous section
B.11	31st July,2014	Kaushik Das	Add a section named PTC Case , Example in Rule Algorithm, Miscellaneous section
B.12	17th Sept, 2014	Kaushik Das	Add more example in Miscellaneous section, Validator and Table Builder
B.13	1st Dec, 2015	Kaushik Das	Add more example in Table Builder
C.1	7th Sept, 2018	Kaushik Das	Change the format , added advanced query spec example and tree builder

TABLE OF CONTENT

1. RULE ALGORITHM

Rule Algorithm's are used to determine different attributes value in OIR.PTC provides a set of algorithms that are available for use in object initialization rules. But there are many business case scenario where you will find that sometime OOTB RuleAlgorith's are not smart enough. That leads to creation of your own customized Rule Algorithm. To create custom algorithms, PTC provides the Rule Algorithm interface upon which all out-of-the-box algorithms have been built. For details on how to use this interface to create custom algorithms, see the Javadoc associated with this interface.

Below is a list of OOTB rule algorithm and few scenarios where you can use customized Rule Algorithm.

Based on the functionality of the Rule Algorithm they are further categorized in Test Algorithm and Branch Algorithm.

Test algorithms are use to test whether the Input is in desirable format or not. Like IfNullTest is used to check whether the input is null or not. EqualsTest is used to check whether to Objects are equal or not etc.

Algorithm	Description
wt.rule.algorithm.EqualsTest	Given two objects, determine the equality. This is an object to object comparison. Return TRUE if there is a match; otherwise, return FALSE.
wt.rule.algorithm.StringEqualsTest	Given an attribute and value determine the equality. The algorithm forces everything to be a string using toString() before the comparison. This string to string comparison is case insensitive. Return TRUE if there is a match; otherwise, return FALSE.
wt.rule.algorithm.StringRegExEqualsTest	Given an attribute and a value, with "*" in the value, determine the equality using regular expression-related concepts. This algorithm uses the java.util.regex implementation of regular expression matching. The algorithm forces the first <Arg> tag value to be a string using toString() before doing the comparison. The return is TRUE if it finds a match and the match is the whole string argument.
wt.rule.algorithm.IfNullTest	Given the value of an attribute, determine if the value is a null value. Return TRUE if the first argument is a null value; otherwise, return FALSE. To return a null value, use the wt.rule.algorithm.GetNullValue algorithm.

Branch Algorithms

Algorithm	Description
wt.rule.algorithm.BooleanBranch	Takes a list of three objects. The first object contained in a Value tag is expected to be an object that returns a Boolean that tells BooleanBranch algorithm which of the other two objects, each contained in an Arg tag, to return.
wt.rule.algorithm.CaseBranch	Is similar to BooleanBranch, except that the branch occurs when a case is TRUE. There is an Arg element for each case plus one additional Arg element that identifies the default if no cases are TRUE.

Apart from the Rules listed above there are many more like AndTest, IndexOf, LatIndexOf, IfNotNullTest, Substring, StringConstant, OrTest etc. All OOTB RuleAlgorithm's are present inside wt.rule.algorithm package.

Few scenario where we can use OOTB Rule Algorithm

```
<AttrValue id="lifeCycle.id" algorithm="com.ptc.core.foundation.lifecycle.server.impl.LifeCycleTemplateAttributeAlgorithm">
    <Value algorithm="wt.rule.algorithm.BooleanBranch">
        <Value algorithm="wt.rule.algorithm.StringEqualsTest">
            <Attr id="endItem"/>
            <Arg>true</Arg>
        </Value>
        <Arg>Approval</Arg>
        <Arg>Agreement</Arg>
    </Value>
    </AttrValue>
```

Depending upon the attributes value the lifecycle will be selected

```
<AttrValue id="number" algorithm="com.ptc.windchill.enterprise.revisionControlled.server.impl.NumberGenerator">
   <Arg>{GEN:wt.enterprise.SequenceGenerator:WTPARTID_seq:20:1}</Arg>
</AttrValue>
```

11111111111111111161

1>If the part is end item then it will follow Approval lifecycle or Agreement.
2>The number will be of 20 digit and will get padded by '1' instead of '0'

1.1 Requirement I : Number generation based on container

You want to generate the number for WTDocument in such a way so that the name given by the user and the context name should append before the OOTB sequence.

❖ Step 1

Create your own rule algorithm ext.rule.ContextVal

❖ Step 2

Change the OIR for WTDocument and upload it.

```
<! -- set the number to a generated number -->
<AttrValue id="number"
algorithm="com.ptc.windchill.enterprise.revisionControlled.server.impl.NumberGenerator">
        <Value algorithm="ext.rule.ContextVal">
                <Attr id="name"/>
        </Value>
        <Arg>{GEN:wt.enterprise.SequenceGenerator:WTDOCUMENTID_seq:10:0}</Arg>
</AttrValue>
```

Add the bold lines in OOTB OIR

1.2 Requirement II : Append current year in the number

User wants to append the current year before the name to generate the number of Object.

i.e. Suppose WTDocument's name is PR00012 then the number will be 14PR00012 if the creation year is 2014.

❖ Step 1

Custom Rule Algorithm named YearRuleAlgorithm

❖ Step 2

Change the OIR for WTDocument and upload it.

```
<!-- set the number to a generated number -->
        <AttrValue id="number"
        algorithm="com.ptc.windchill.enterprise.revisionControlled.server.impl.NumberGenerator">
                <Value algorithm="
                        ext.test.rule.YearRuleAlgorithm">
                        <Attr id="name"/>
                </Value>

        </AttrValue>
```

1.3 Requirement III : View Based Lifecycle

Depending upon the view the life cycle state will get assigned. (Use View.id as parameter for the RuleAlgorithm)

❖ Step 1

Create a class named ViewRuleAlgorithm.

❖ Step 2

```
<!-- set the lifecycle -->
<AttrValue id="lifeCycle.id"
   algorithm="com.ptc.core.foundation.lifecycle.server.impl.LifeCycleTemplateAttributeAlgorithm">
      <Value algorithm=" ext.test.rule.ViewRuleAlgorithm ">
         <Attr id="view.id"/>
      </Value>
</AttrValue>
```

Add the bold lines in OOTB OIR

1.4 Requirement IV : Document Number Auto-Generation based on container

In Project whenever a WTDocument or any subtype of WTDocument is created the number will be Total no of WTDocument in that particular project + Project Number i.e. Project Number is 000082 the first documents number will be 0001-000082 next 0002-000082 and so on.

Note

1>The OIR is only applicable on project context and for type WTDocument.
2>Project Number field is mandatory at the time of Project Generation.

❖ Step 1

Create class ext.test.rule.GettingSequence

❖ Step 2

```
<!-- set the number to a generated number -->
<AttrValue id="number"
algorithm="com.ptc.windchill.enterprise.revisionControlled.server.impl.NumberGenerator">
      <Value algorithm="ext.test.rule.GettingSequence"/>
</AttrValue>
```

1.5 Requirement V : Add product name in CR Number

How to add product name to Change Request number by OIR.

❖ Steps

Title	How to add product name to Change Request number by OIR
Descripti On	How to add product name to **Change Request** number by **OIR?**
Applies To	Windchill PDMLink 10.1 all datecodes, 10.2 all datecodes
Cause	
Resolution	1. Click **Site > Utilities > Object Initialization Rules Administration or Org > Utilities** 2. Download **Change Request rule.xml** and changed as below ---------- … --<! set the number to a generated number --> <AttrValue id="number" algorithm="com.ptc.windchill.enterprise.revisionControlled.server.impl.NumberGenerator"> <Arg>CR-</Arg> <Attr id="containerName"/> <!-- Internal name --> <Arg>{GEN:wt.enterprise.SequenceGenerator:WTCHANGEISSUEID_seq:5:0}</Arg> </AttrValue> … ---------- 3. Edit and upload the updated rule.xml

1.6 Requirement VI : Folder selection based on attribtue value

Based on IBA value the folder will be decided automatically. For example I assume the IBA internal name is “Value” and the vaule of the IBA is “1,2,4,5” and the corresponding folders name are “Folder1”,” Folder 2” and so on.The IBA is attached with Document.

❖ Steps

Create the IBA and the folders as specified above.

Use the below OIR in Product or Organization level based on the requirement.

```
<AttributeValues objType="wt.doc.WTDocument">
<!-- set the folder -->
<AttrValue id="folder.id" algorithm="com.ptc.core.foundation.folder.server.impl.FolderPathAttributeAlgorithm">
    <Value algorithm="wt.rule.algorithm.CaseBranch">
        <Value algorithm="wt.rule.algorithm.StringEqualsTest">
            <Attr id="Value"/>
            <Arg>1</Arg>
        </Value>
        <Arg>/Default/Folder1</Arg>
        <Value algorithm="wt.rule.algorithm.StringEqualsTest">
            <Attr id="Value"/>
            <Arg>2</Arg>
    </Value>
    <Arg>/Default/Folder2</Arg>
    <Value algorithm="wt.rule.algorithm.StringEqualsTest">
            <Attr id="Value"/>
            <Arg>4</Arg>
    </Value>
    <Arg>/Default/Folder4</Arg>
    <Value algorithm="wt.rule.algorithm.StringEqualsTest">
            <Attr id="Value"/>
            <Arg>5</Arg>
    </Value>
    <Arg>/Default/Folder5</Arg>
    <Arg>/Default</Arg>
    </Value>
</AttrValue>
<AttrConstraint id="folder.id" algorithm="com.ptc.core.rule.server.impl.GatherAttributeConstraints">
<Value algorithm="com.ptc.core.rule.server.impl.GetServerAssignedConstraint"/>
<Value algorithm="com.ptc.core.rule.server.impl.GetImmutableConstraint"/> </AttrConstraint>
</AttributeValues>
```

2. DATA UTILITY

Overview

Most of Windchill pages are constructed using GUI component which are table, tree or attributes panel, etc.
Sometimes, we need to change part of UI Elements on GUI component like following.

- Need to change one item column style on the attribute panel.
- Want to change from select box to check box design on create wizard.
- Want to show calculated value of special attributes on info page

What is UI Element?

Any visual element displayed on a Pages.

What is GUI component?

Object that carries information about a UI Element that is necessary to appropriately render the element on a page. It also includes the default renderer class that knows how to generate the appropriate display, based on the information in the GUI component.

What is renderer?

A Renderer is an object that knows how to generate HTML for a GUI Component. All UI Component has been controlled by user interface renderer, so it is difficult to change partial user interface directly. In this time, if you want to change UI Element, generally we can use Data Utility. Data utilities are the glue that takes raw query data and converts it into the model data that a component can be built out of. In the case of tables, the data utilities are responsible for constructing particular cell values. Generally Data Utility is using Component Descriptor, Component Model and Component Bean for change UI Component.

GUI Components

This component will generally be constructed in the Method Server and then returned to the Servlet Container for rendering. The GUI Component interface has one method that needs to be implemented:

public void draw(Writer out, RenderingContext renderContext) throws RenderingException; Following is OOTB lists of GUI Components

1. AttributeDisplayComponentCheckBox
2. LocationInputComponent
3. ComboBox
4. AttributeGuiComponent
5. DateDisplayComponent
6. AttributeInputComponent
7. DateInputComponent
8. AttributeInputCompositeComponent
9. EnumInputComponent
10. BooleanInputComponent
11. UrlInputComponent
12. NumberInputComponent
13. IconComponent
14. NumericDisplayComponent
15. Label
16. NumericInputComponent
17. PushButton
18. TextArea
19. RadioButton
20. TextBox
21. RevisionInputComponent
22. TextDisplayComponent
23. StringInputComponent
24. PickerInputComponent

Create Custom Data Utility Class

The data utility interface (com.ptc.core.components.descriptor.DataUtility) has three core methods:

- The getDataValue method gets the value that should be placed within a particular table cell. Typically, the object that is returned by this method should be an instance of GUIComponent. This interface has been retrofitted to NmString, NmDate.

Object getDataValue(String component_id, Object datum, ModelContext mc) throws WTException;

- The setModelData method allows the data utility to prefetch data for all the row objects for a particular column that will be rendered. The method is called before getDataValue is called for any row, and is supplied a List of all the objects that will be processed.

void setModelData(String component_id, List objects, ModelContext mc) throws WTException;

- The getLabel method allows the data utility to populate the descriptor with the correct label. There is a default implementation in the AbstractDataUtility for getting a label from a common ui rbinfo file.

String getLabel(String component_id, ModelContext mc) throws WTException;

When using DataUtility, we have to use a abstract super class AbstractDataUtility generally. AbstractDataUtility is the child class of DataUtility interface.

2.1 Requirement I: Create Dropdown list

To create Dropdownlist as input parameter for your IBA

❖ Step 1:Create class ext.datautility.DropDownUtility

To add javascript function along with the DropDown list for certain event like "onChange" or "onClick" then use "com.ptc.core.components.rendering.guicomponents.ComboBox" instead of StringInputComponent.

```
if (modelContext.getDescriptorMode().equals(ComponentMode.CREATE)) {
                    if (object instanceof AttributeInputCompositeComponent) {
                    ArrayList<String> displayList = new ArrayList<String>();
                    displayList.add("Yes");

                    displayList.add("No");
                    ComboBox cb = new ComboBox();
                    cb.setInternalValues(displayList);
                    cb.setValues(displayList);

                    cb.setColumnName(AttributeDataUtilityHelper.getColumnName(componentId,
                    datum,
                    modelContext)); //cb.addJsAction(<javascript event> , <javascript function
                    name>)

                    cb.addJsAction("onSubmit", "delValue()");
            }

        }
```

Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM

❖ Step 2:Register your data utility at site.xconf

Customized data utility class cannot use directly on GUI Component. All GUI Component can read service Identification which registered in MethodServer services cache – generally registered in "service.properties".

```
<Service context="default" name="com.ptc.core.components.descriptor.DataUtility"
targetFile="codebase/com/ptc/core/components/components.dataUtilities.properties">
        <Option cardinality="singleton" order="0" overridable="true"
                requestor="java.lang.Object"

                selector="<Internal name of the IBA>"
                serviceClass="<Fully qualified name of your data utility class>"/>

        </Service>
        eg:
        <Service context="default"
          name="com.ptc.core.components.descriptor.DataUtility"
          targetFile="codebase/com/ptc/core/components/components.dataUtilities.proper
          ties">
                <Option cardinality="singleton" order="0" overridable="true"
                requestor="java.lang.Object"
                selector="myIba"
                serviceClass="ext.datautility.DropDownUtility"/>
        </Service>
```

2.2 Requirement II: Cration of Document Picker

Want to create a document picker as input parameter of IBA.

❖ Step 1

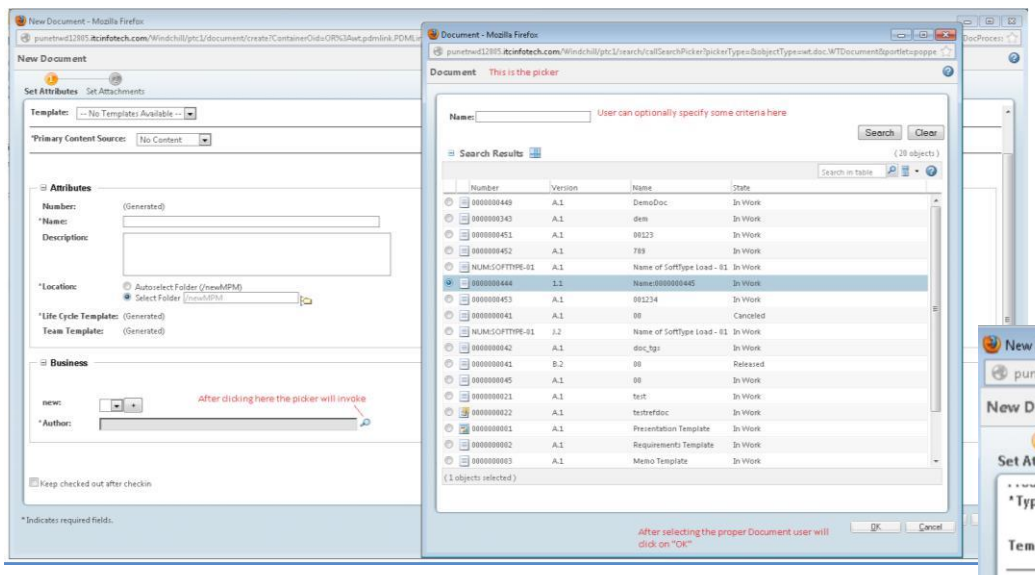
Create class ext.datautility.DocumentPickerUtility

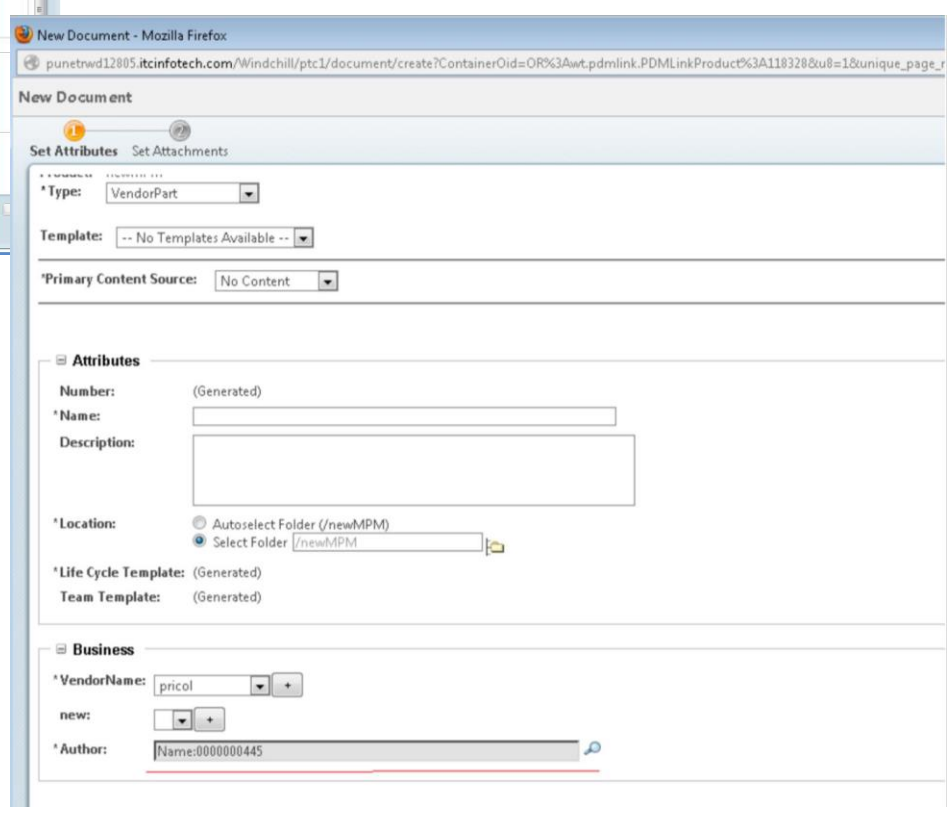
Open Type and Attribute Manager ☐ go to "Create New "Layout and "Edit "Layout ☐ Find your IBA ☐ Open Group Attribute Properties and set the DataUtilityId as 'myPicker' for both the layout.

At the time of registering your data utility in the site.xconf use the ID 'myPicker' as a selector instead of the name of your IBA

Register your data utility and run xconfmanager -p and restart server.

❖ Screen-Shot:





2.3 Requirement III: Part Picker with Custom Search Criteria

Want to create a part picker with custom search criteria.

❖ Step 1

Create class TestItemPicker

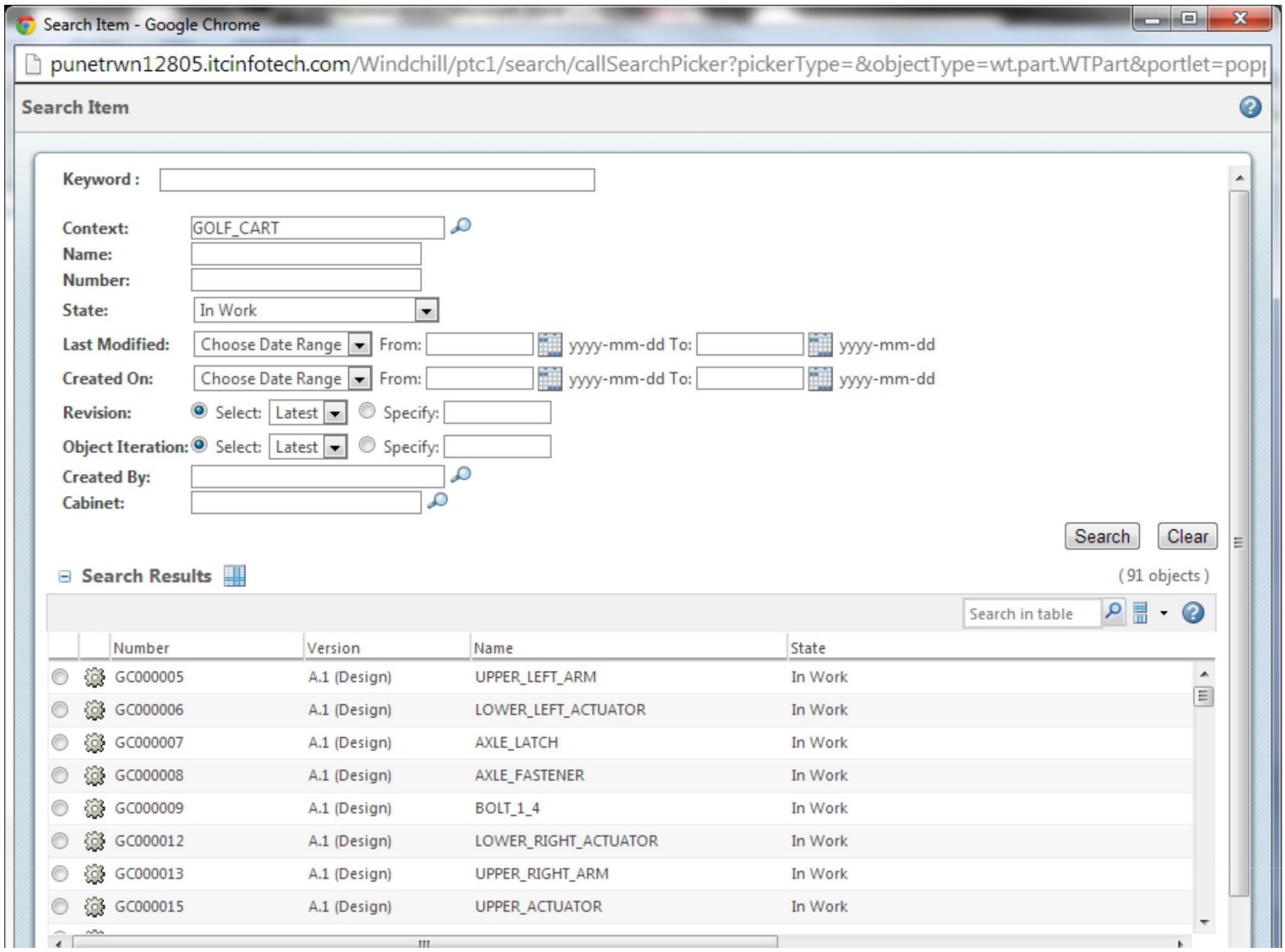
❖ Step 2

Register data utility and define search criteria.
To define Search criteria
Open <Windhcill_home>\codebase\pickerAttributes.xml and add your own search criteria with the same componentID mentioned in the dataUtility. (In this example it is TestPicker)
Sample of search criteria used in this example

```
<ComponentID id="TestPicker">
<ObjectType id="wt.part.WTPart">
<SearchCriteriaAttributes>
<Attributes>
<Name>contextRef</Name>
<DisplayName>CONTEXT_LABEL</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes><Attributes>
<Name>name</Name>
<DisplayName>NAME_LABEL</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes><Attributes>
<Name>number</Name>
<DisplayName>NUMBER_LABEL</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes><Attributes>
<Name>state.state</Name>
<DisplayName>STATE_LABEL</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes><Attributes>
<Name>thePersistInfo.modifyStamp</Name>
<DisplayName>LAST_UPDATED_ATTRIBUTE</DisplayNam
e> <IsSearchable>true</IsSearchable>
</Attributes><Attributes>
<Name>thePersistInfo.createStamp</Name>
<DisplayName>CREATED_ATTRIBUTE</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes>
<Attributes>
<Name>versionInfo.identifier.versionId</Name>
<DisplayName>VERSION_LABEL</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes>
<Attributes>
</Attributes>
<Attributes>
<Name>iterationInfo.identifier.iterationId</Name>
<DisplayName>ITEM_ITERATION_LABEL</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes>
<Attributes>
<Name>iterationInfo.creator</Name>
<DisplayName/>
<IsSearchable>true</IsSearchable>
</Attributes>
<Attributes>
<Name>folderingInfo.cabinet</Name>
<DisplayName>CABINET_TABLEVIEW_LABEL</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes>
<Attributes>
<Name>containerInfo.ownerRef</Name>
<DisplayName>OWNER_LABEL</DisplayName>
<IsSearchable>true</IsSearchable>
</Attributes>
</SearchCriteriaAttributes></ObjectType></ComponentID>
```

Use the same componentID in data utility

❖ Screenshots:



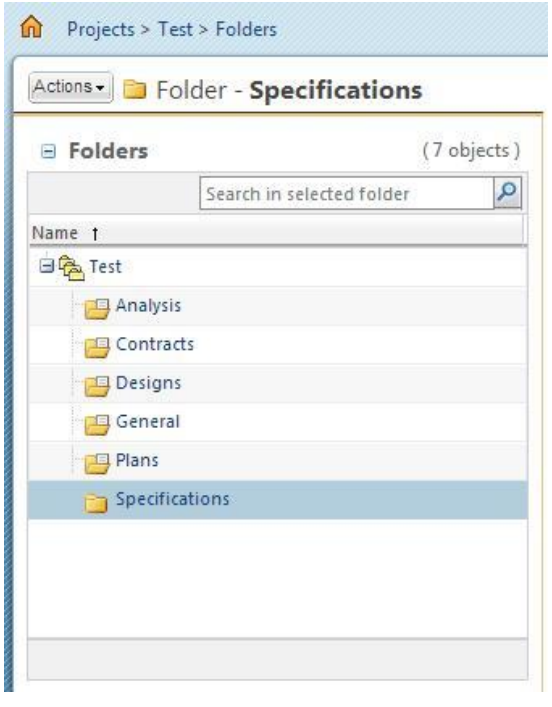
❖ **Property of PickerRenderConfigs**

PickerRenderConfigs.*PICKER_ID*	All picker must have a unique Picker Id.The value is of String datatype. It's a mandatory property for any picker
PickerRenderConfigs.*OBJECT_TYPE*	The Object type to pick. The object type can be any OOTB hard type as well as soft type.Ex.WCTYPE|wt.doc.WTDocument|com.itcinfotech.MySubType It's a mandatory property for any picker
PickerRenderConfigs.*PICKER_TITLE*	The picker title which is displayed in the top-left corner of the picker.
PickerRenderConfigs.*PICKER_ATTRIBUTES*	This property value will decide which attribute of the picked object to store. If your picked object is WTPart then the picker attribute can be name,number etc.
PickerRenderConfigs.*READ_ONLY_TEXTBOX*	Property to decide whether the textbox contains the picked attribute value is read-only or not. The value of the property is of Boolean datatype. i.e true,false
PickerRenderConfigs.*COMPONENT_ID*	The component id which is used to define custom search criteria in pickerAttribute.xml
PickerRenderConfigs.PICKER_TYPE	To decide whether the picker type is search or picker.
PickerRenderConfigs.PICKER_CALLBACK	Specify the picker callback function. It's an optional property.

For a full list of PickerRenderConfigs property refer to Windchill Customization guide.

2.4 Requirement IV : Change Folder icon based on presence of content

In navigator (Top Left) if any folder is empty the icon should be different.



❖ Step 1

Create class FolderedDataUtility

❖ Step 2

Open site.xconf and add the following line.

```
<Service context="default" name="com.ptc.core.components.descriptor.DataUtility"
targetFile="codebase/com/ptc/windchill/enterprise/enterprise.dataUtilities.properties">
        <Option cardinality="duplicate" order="0" overridable="true"

        requestor="java.lang.Object"
        selector="folderName"
        serviceClass="ext.test.datautility.FolderedDataUtility"/>

</Service>
```

❖ Step 3

Run xconfmanager -p from Windchill shell and Start / Restart method server.

❖ Note

It's not advisable to change or modify site.xconf. Instead of that it's always a good practice to make one custom xconf file.

1. Create one xconf file. Like test.xconf

2. Put the file in any path under <WINDCHILL_HOME>.

3. For Example I put test.xconf in <WINDHCILL_HOME>/codebase/ext/test/test.xconf

4. Open declarations.xconf and enter your xconf file's path there <ConfigurationRef xlink:href="codebase/ext/test/test.xconf"/>

5. Open test.xconf and enter below lines to register the FolderedDataUtility.

```
<Service context="default" name="com.ptc.core.components.descriptor.DataUtility"
targetFile="codebase/com/ptc/windchill/enterprise/enterprise.dataUtilities.properties">
        <Option cardinality="duplicate" order="0" overridable="true" requestor="java.lang.Object"
selector="folderName"
                serviceClass="ext.test.datautility.FolderedDataUtility"/>
</Service>
```

6. In Windchill shell run xconfmanager -p and start / restart windchill.

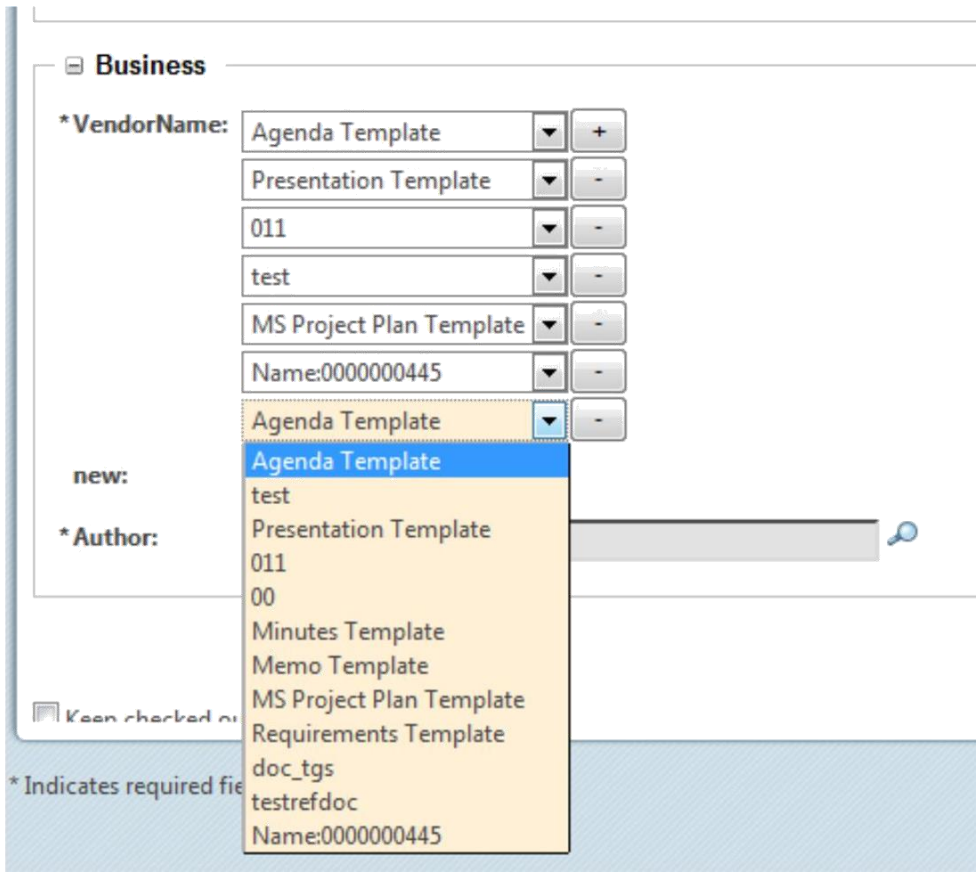
2.5 Requirement V : Multi valued Dropdown list

Want to create a multi-valued dropdown list as an input parameter for an IBA

The easiest way to do this is through UI but recently I got a requirement which force me to create Multi Valued Dropdown list programmatically because the values of the drop down lists are dynamic and it's really involved a lots of coding. So, requesting everyone if anyone knows any easier method please share with me.

Below is an example which creates a Multi Valued dropdown list and the value in that Drop Down list will be the Documents name present in the windchill.

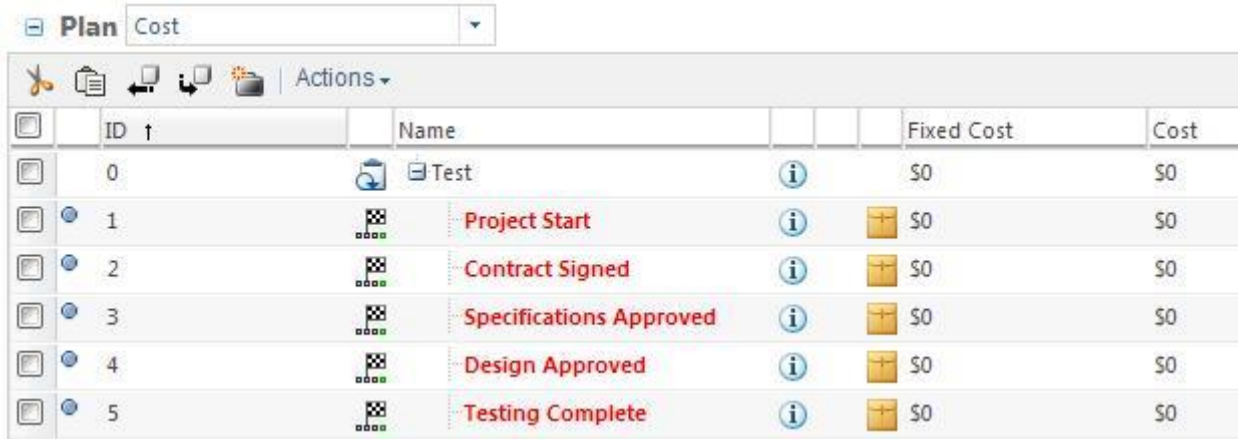
Finally it will look like this.



Create data utility class DepartmentDataUtility and register it.

2.6 Requirement VI : Highlight activity name in Plan table based on pre-defined criteria

Highlight Activity name in Plan table if the activity is Critical or the due date is passed.



❖ Step 1

Create class ColorNameDataUtility.java

❖ Step 2

Register the data utility and start / restart server after compiling the xconf file using xconfmanager -p

```
<Service context="default" name="com.ptc.core.components.descriptor.DataUtility"
targetFile="codebase/com/ptc/windchill/enterprise/enterprise.dataUtilities.properties">
        <Option cardinality="duplicate" order="0" overridable="true"
        requestor="java.lang.Object"
        selector="name"
        serviceClass="ext.test.datautility.ColorNameDataUtility"/>
</Servide>
```

2.7 Requirment VII : Create Radio Button

User want radio button's as input parameter for an IBA.

Snapshot in attribute panel of Wizard

In details (View Information) tab

❖ Step 1:

Create class **RadioButtonDataUtility.java**

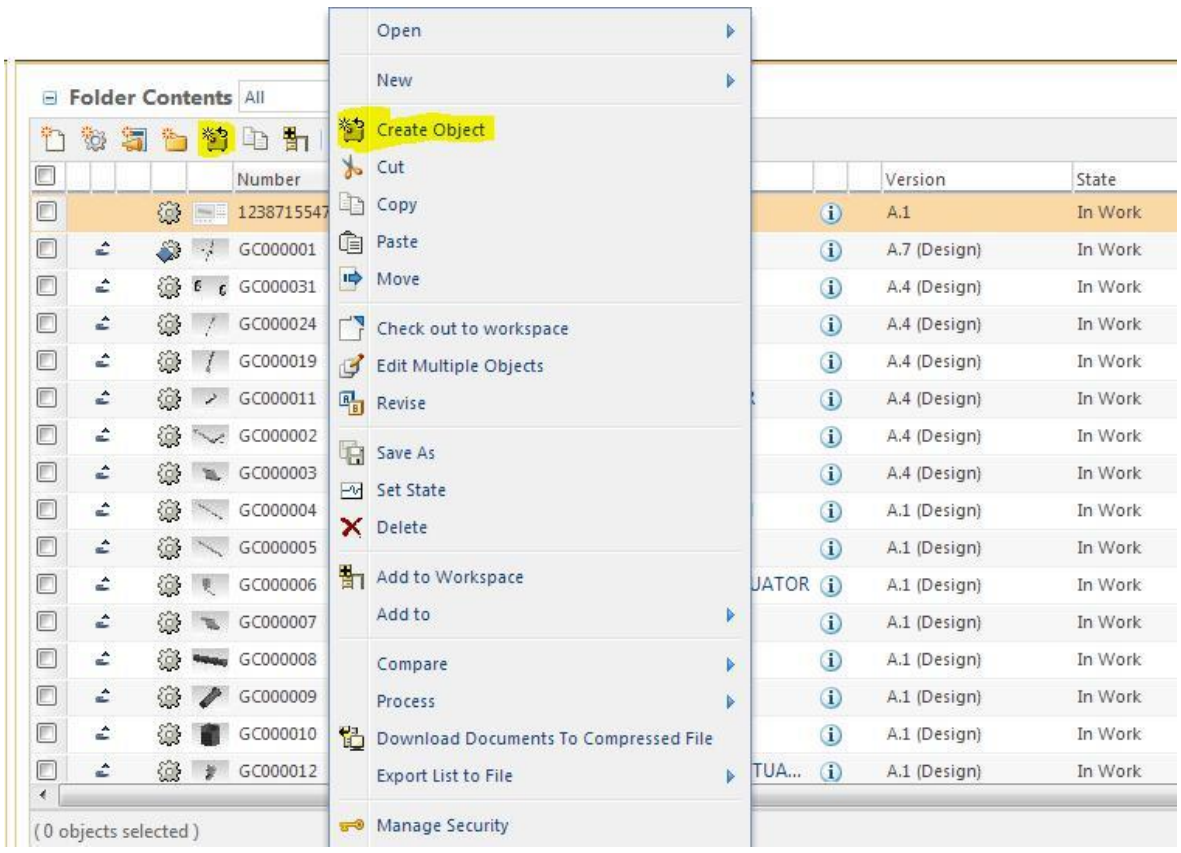
❖ Step 2:
Register it in xconf

❖ Step 3:
Restart the server.

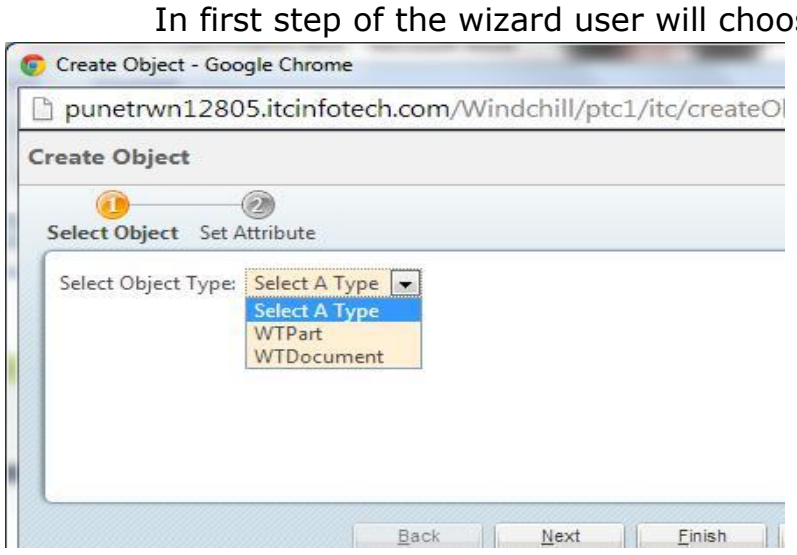
3. WIZARD AND PROCESSOR

3.1 Requirement I : Creare two step custom wizard to creare WTPart / WTDocument

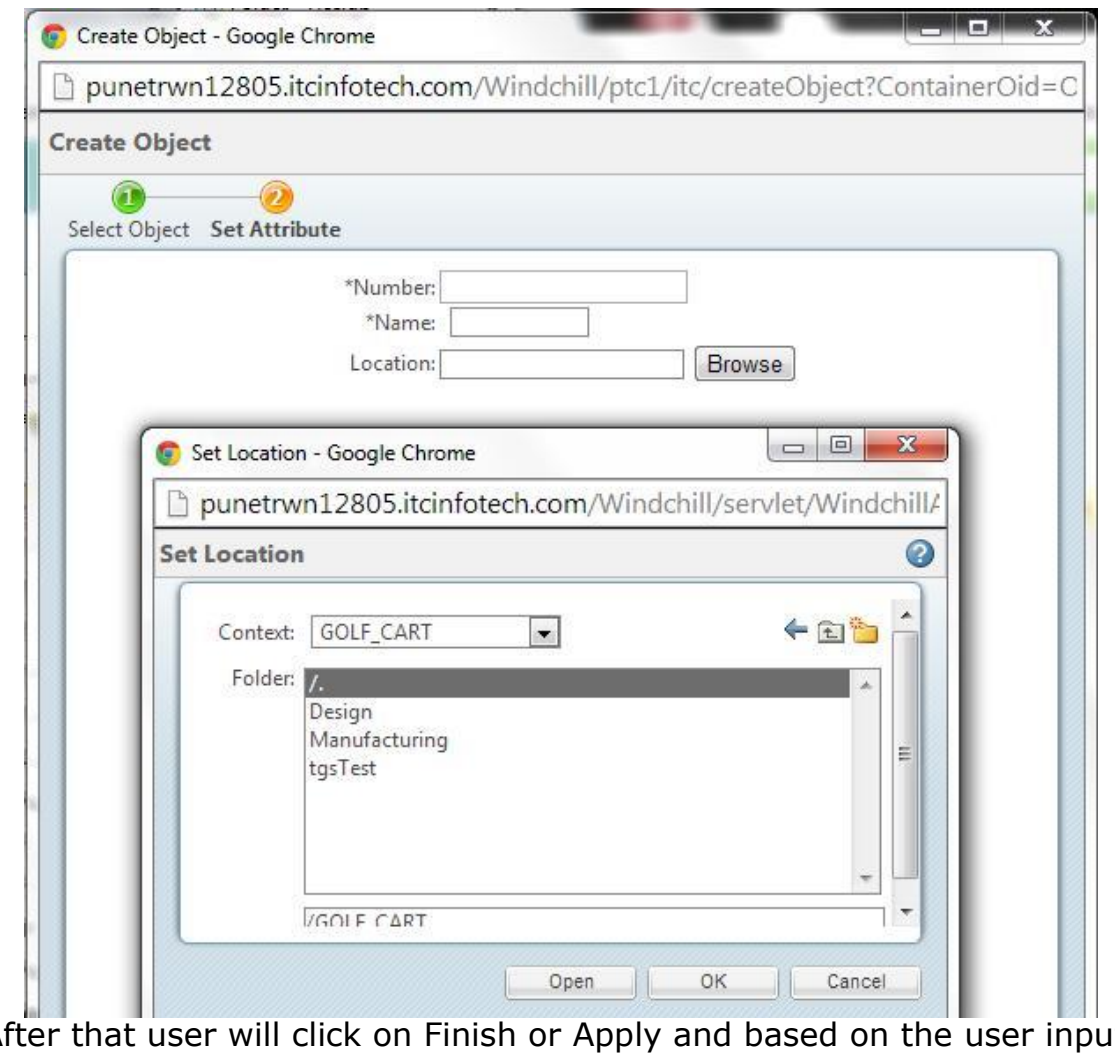
We have to create a simple two step wizard to create WTPart or WTDocument in Windchill.
There will be a Custom Action after clicking which the wizard will pop-up.



In first step of the wizard user will choose what object to create “WTPart” or “WTDocument”



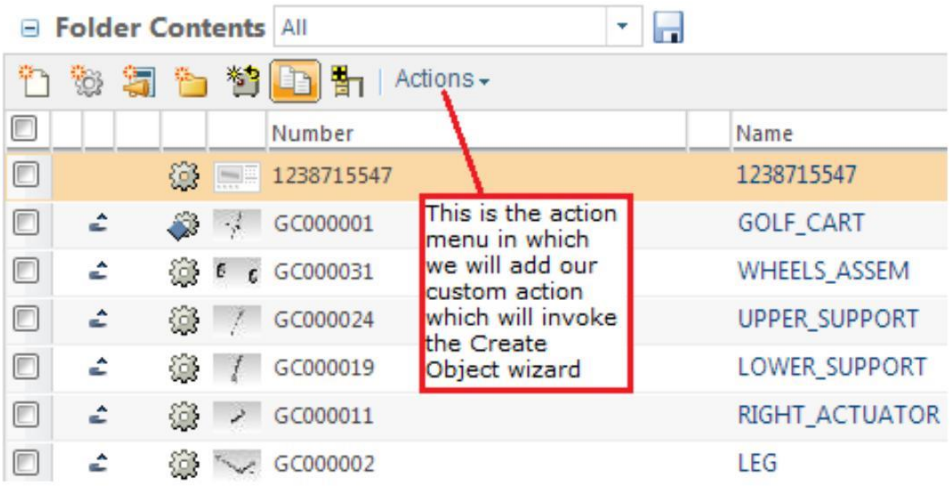
In next step user will give three attribute’s value. (Name, Number and Location to create)

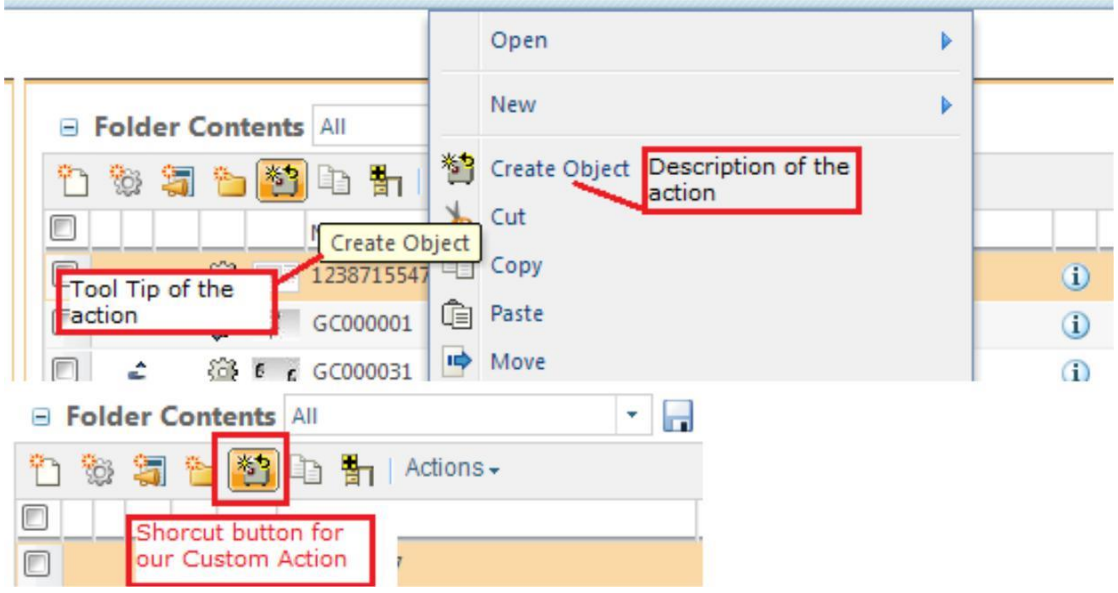


After that user will click on Finish or Apply and based on the user input corresponding object will be created.

❖ **Step 1:**

To create the custom actions we have to write a ResourceBundle file. In resource bundle file there will be entry for three different action.(In the example they are "Create Object" after clicking which the wizard will invoke, "Select Object" the first step of the wizard, "Set Attribute" the second step of the wizard)





For this example the ResourceBundle file is "CustomActionResource.java" which is in "ext.test.resource" package.

Next we will make some entry in "custom-actions.xml" and "custom-actionModel.xml".
Path of these files are "<Windchill_Home>\codebase\config\actions"

After changing the custom-actionModel.xml our file will look like this, based on the windchill version the custom-actionModel will change.

```
<model name="folderbrowser_toolbar_actions">
<description>Folder browser toolbar actions menu for all Folders.</description>
<action name="createObject" type="itc" shortcut="true"/> <-- To generate a shortcut button in table toolbar use shortcut=" true" -->
<submodel name="folderbrowser_toolbar_open_submenu" />
<action name="separator" type="separator" />
<submodel name="folderbrowser_toolbar_new_submenu" />
<action name="separator" type="separator" />
<action name="list_cut" type="object" />
<action name="list_copy" type="object" shortcut="true" />
<action name="pasteAsCopy" type="saveas" />
<action name="fbpaste" type="object" />
<action name="CONTAINERMOVE" type="pdmObject" />
<action name="separator" type="separator" />
<action name="multiObjectCheckIn" type="wip" />
<action name="multiObjectCheckOut" type="wip" />
<action name="PAGERELOADINGMULTICHECKOUT" type="folder" />
<action name="multiObjectUndoCheckout" type="wip" />
<action name="editMultiObjects" type="object" />
<action name="MULTIREVISEITEMS_FROMFOLDERS" type="pdmObject" />
<action name="route" type="workflow" />
<action name="separator" type="separator" />
<action name="WFMULTISAVEAS" type="folder" />
<action name="MULTIRENAME" type="folder" />
<action name="SETSTATE_FROMFOLDERS" type="pdmObject" />
<action name="export" type="object" />
<action name="cadmultiexport" type="object" />
<action name="list_delete" type="object" />
<action name="batchPrint" type="wvs" />
<action name="separator" type="separator" />
<action name="WFADDTOWORKSPACE" type="folder" shortcut="true" />
<submodel name="folderbrowser_toolbar_addto_submenu" />
<action name="separator" type="separator" />
<action name="sendToPDM" type="sandbox" />
<action name="convertToShareTB" type="sandbox" />
<action name="sandboxCheckoutShareMultiSelect" type="object" />
<action name="sandboxUndoCheckout" type="object" />
<action name="manageIdentityConflicts" type="sandbox" />
<action name="SBUpdatePrj" type="sandbox" />
<action name="updateShareMultiSelect" type="sandbox" />
<action name="removeShareTB" type="object" />
<action name="separator" type="separator" />
<submodel name="folderbrowser_toolbar_compare_submenu" />
<submodel name="folderbrowser_toolbar_process_submenu" />
<action name="downloadDocumentsToCompressedFile" type="document" />
<action name="createImportJob" type="ixb"/>
<submodel name="folderbrowser_toolbar_exportlisttofile_submenu" />
<submodel name="folderbrowser_toolbar_requirements_submenu" />
<action name="importFromIntegrity" type="integrityRM" />
<action name="separator" type="separator" />
<action name="multiObjManageSecurity" type="accessPermission"/>
<action name="MULTIEDITSECURITYLABELS" type="object"/>
<action name="createSubscription" type="subscription" />
<action name="separator" type="separator"/>
<action name="track_new_work_table" type="resourceAssignment" /><
<action name="new_plan_activity_table" type="planActivity" /><
<includeFilter name="ActionFilterOnProject2State" />
</model>
```

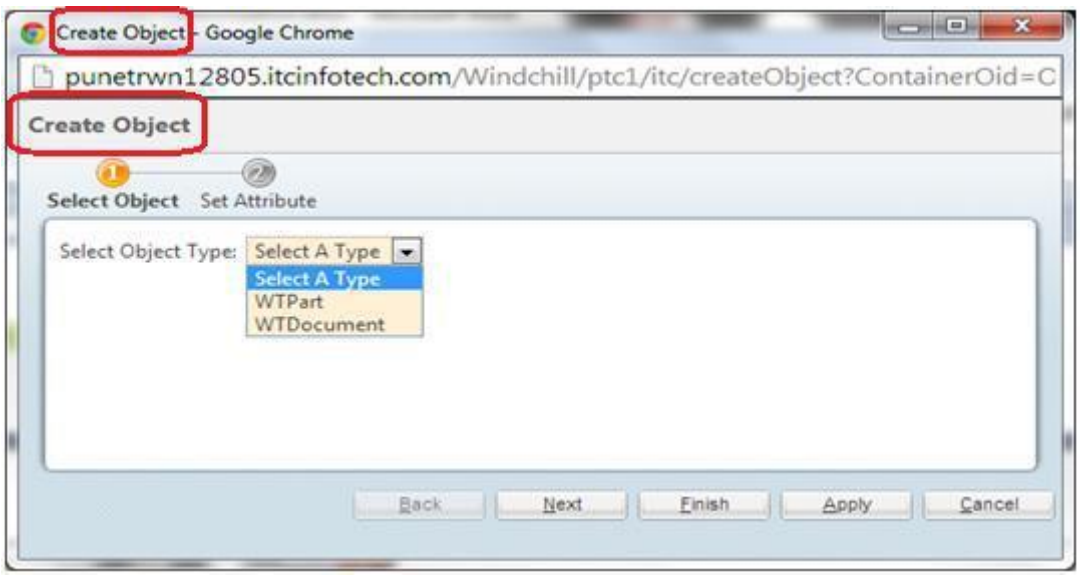
Only add the bold section. Based on windchill version the ootb entry will be different. The example is from Windchill 11.0

Next we will make some changes in custom-actions.xml.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE listofactions SYSTEM "actions.dtd">
<listofactions>
<!-- The action type is "itc" which is used in resource bundle file also -->
<objecttype name="itc"
      resourceBundle="ext.test.resource.CustomActionResource"> <!--
      Wizard to create document -->
      <action name="createObject" type="itc" >
            <command url="/netmarkets/jsp/itc/createObject.jsp"
class="ext.test.processor.SelectedObjectFormProcessor" windowType="popup"/>
<!-- Processor class for our custom wizard -->
      </action>
      <action name="selectObject" type="itc" >
            <command url="/netmarkets/jsp/itc/selectObject.jsp"
      windowType="wizard_step" /> </action>
      <action name="setAttribute" type="itc" >
            <command url="/netmarkets/jsp/itc/setAttribute.jsp"
      windowType="wizard_step" /> </action>
</objecttype>
```

❖ **Step 2:**

In next step we will create three jsp. For wizard the action name and JSP name should be identical.
(All the JSP used in this example is in <Windchill_Home>/codebase/netmarkets/jsp/itc)



"createObject.jsp"

```
<%@ taglib prefix="jca"
uri="http://www.ptc.com/windchill/taglib/components" %> <%@include
file="/netmarkets/jsp/components/beginWizard.jspf"%> <%@include
file="/netmarkets/jsp/components/includeWizBean.jspf"%>
 <jca:wizard title="Create Object">
            <jca:wizardStep action="selectObject" type="itc"/>
            <jca:wizardStep action="setAttribute" type="itc"/>
 </jca:wizard>

<%@include file="/netmarkets/jsp/util/end.jspf"%>
```

selectObject.jsp

```
<html>
<body>
Select Object Type: <select name="Type">
      <option >Select A Type</option>
  <option >WTPart</option>
  <option >WTDocument</option>
</select>
</body>
</html>
```

setAttribute.jsp

```
<%@ include file="/netmarkets/jsp/components/beginWizard.jspf" %>
<%@ taglib prefix="w" uri="http://www.ptc.com/windchill/taglib/wrappers"%>
<table>
	<tr>
		<td scope="row" width="200” align="right">
			*Number:
		</td>
		<td align="left">
			<input type="textbox" name="null___number___textbox" id="number" />
		</td>
	</tr>
	<tr>
		<td scope="row" width="200” align="right">
			*Name:
		</td>
		<td align="left"> 
			<w: textBox name="name" id="name" maxlength="30" size="10" />
		</td>
		 <! -- Both input type textbox and w:textBox will create a text box for user input.
				One is using normal HTML tag another is from Windchill Tag Library that’s the
				only difference -->
	</tr>
	<tr>
		<td scope="row" width="200” align="right">
			Location:
		</td>
		<td align="left">
			<input type="text" name="loc_displayLocation" id="loc_displayLocation">
			<button onclick="ITC.launchFolderPicker(event)" type="button">
				Browse
			</button>
		</td>
	</tr>
</table>
<input type="hidden" name="null___loc_folder___textbox" id="loc_folder" />
<input type="hidden" name="null___loc_container___textbox" id="loc_container" />
<script type="text/javascript">
<!--
//namespace for paste example to avoid colliding with any globally defined functions --> ITC = {};
ITC.launchFolderPicker = function(event) {
var url = getBaseHref() +
'servlet/WindchillAuthGW/wt.enterprise.URLProcessor/invokeAction?action=cadxBrowseLocations&containerVisi
bilityMask=PDMLink&accessPermission=modify&displayHotlinks=false&displayCreateFolder=true';
launchBrowseFolders(url, $('loc_displayLocation'), ITC.locationCallback);
Event.stop(event);
return false;
}

ITC.locationCallback = function(contextValues) {
var containerOID = contextValues.containerOID;
var folderOID = contextValues.folderOID;// 'OR:wt.pdmlink.PDMLINKProduct:123' // 'OR:wt.folder.SubFolder:123'
$('loc_folder').value = folderOID;
$('loc_container').value = containerOID;
}
</script>
<%@include file="/netmarkets/jsp/util/end.jspf"%>
```

In above implementation user can choose only the Product and the Folder inside it.

If you don’t want to restrict user for only “Product” then remove this portion

“**&containerVisibilityMask=PDMLink**” from var url. If you want to restrict for Project then change it to “**&containerVisibilityMask=ProjectLink**”.

❖ **Step 3:**

Now for our wizard we need a processor class. Processor class used in this example.

```
package ext.test.processor;

import java.util.HashMap;
import java.util.List;

import javax.servlet.http.HttpServletRequest;

import wt.doc.WTDocument;
import wt.fc.ObjectReference;
import wt.fc.Persistable;
import wt.fc.PersistenceHelper;

import wt.fc.ReferenceFactory;
import wt.folder.Folder;
import wt.folder.FolderEntry;

import wt.folder.FolderHelper;
import wt.inf.container.WTContainer;
import wt.part.WTPart;
import wt.util.WTException;

import wt.util.WTMessage;
import wt.util.WTPropertyVetoException;

import com.ptc.core.components.beans.ObjectBean;

import com.ptc.core.components.forms.DefaultObjectFormProcessor;
import com.ptc.core.components.forms.FormResult;
import com.ptc.netmarkets.util.beans.NmCommandBean;

/**
*       Processor class for custom wizard.
*       @author 'true' 12805 kaushik.das@itcinfotech.com
      * @version 'true' 1.0
 */
public class SelectedObjectFormProcessor extends DefaultObjectFormProcessor {
      /**
            * Overridden method of {@link DefaultObjectFormProcessor}.
            * @param commandBean
            *        the {@link NmCommandBean} object
       *
            * @param list
            *        list's of {@link ObjectBean}
            * @exception WTException
            *           throws {@link WTException}
            * @return object of {@link
            FormResult} */

      @Override
      public FormResult doOperation(NmCommandBean commandBean,
                  List<ObjectBean> list) throws WTException {
            /*
                  * Fetch the data given by the user. If you know the exact component ID
                  * then you may use this method
             */
            HttpServletRequest request = commandBean.getRequest();
            String type = request.getParameter("Type");
            /*
             *       This is another method to fetch user input
             */
            @SuppressWarnings("rawtypes")
            HashMap map = commandBean.getText();
            Object[] keys = map.keySet().toArray();
            String name = null;
            String number = null;
            String containerOid = null;
            String folderOid = null;
            for (Object oneSet : keys) {
                  if (((String) oneSet).contains("name")) {

                        name = ((String) map.get(oneSet)).toUpperCase();
                  } else if (((String) oneSet).contains("loc_folder")) {
                        folderOid = (String) map.get(oneSet);
                  } else if (((String) oneSet).contains("loc_container")) {
```

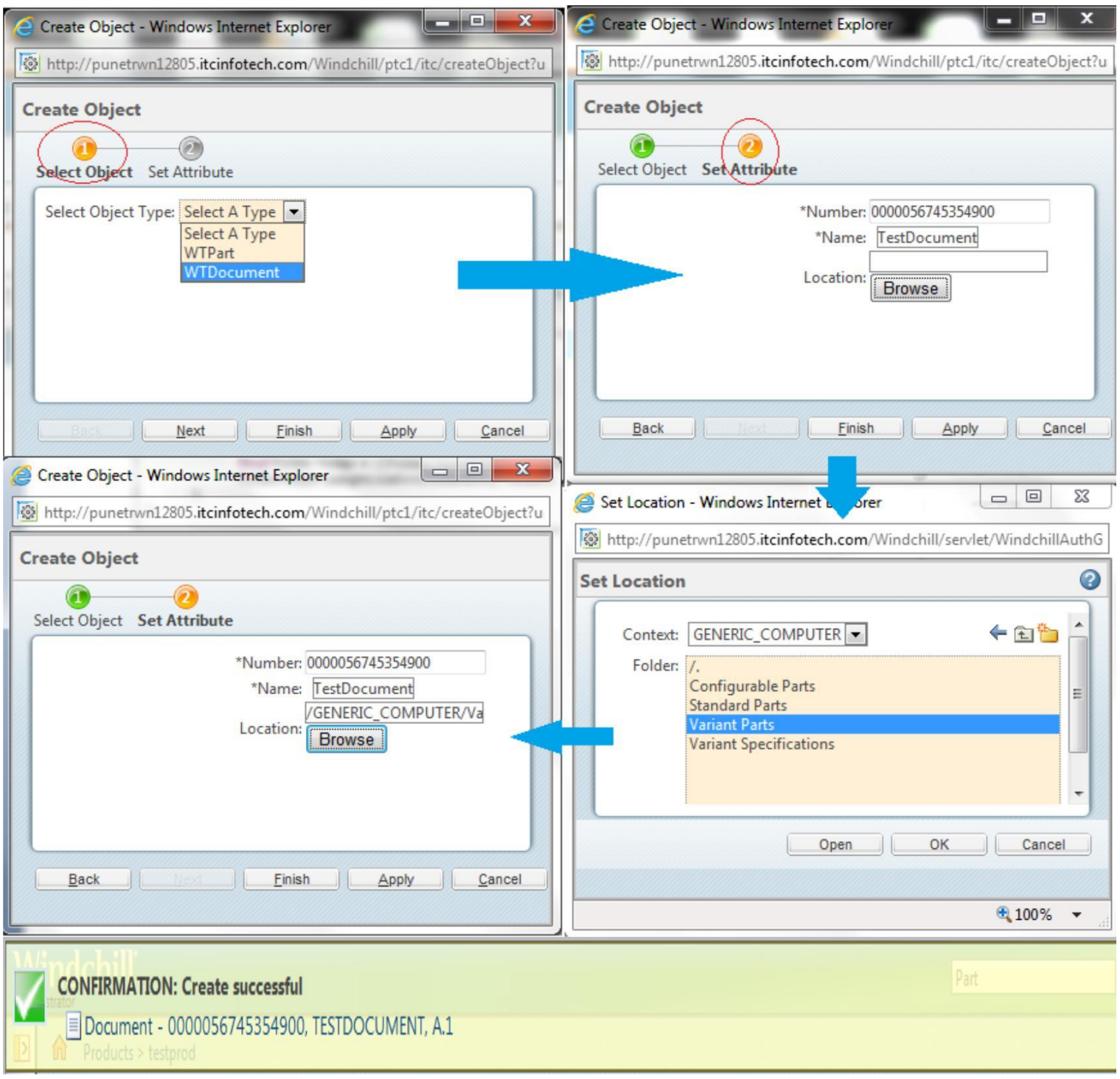
```
        if (name == null || number == null) {
                        /*
                         * Checking whether user provide the name or Number if no then show
                         * proper error message
                         */
                        throw new WTException("Provide value for Name and
                Number"); } else if (folderOid == null || containerOid == null) {
                        /* Checking whether user select the location or not */
                        throw new WTException(
                                        "Select the location properly by clicking on Browse button kindly don't copy paste it");
                }
                Persistable per = null;
                if (type.toLowerCase().contains("part")) {
                        per = createPart(name, number, containerOid, folderOid);
                } else if (type.toLowerCase().contains("document")) {
                        per = createDocument(name, number, containerOid, folderOid);
                } else {
                        throw new WTException("Select a valid object type");
                }
                ObjectBean ob = ObjectBean.newInstance();
                ob.setObject(per);
                list.add(ob);
                /* Adding the Persistable object in the List */
                return super.doOperation(commandBean, list);
        }

        /*
         * <--- Short note about inline message --> You can develop inline message
*         to capture success or failure of certain user actions. You can also use
         * this to display warning or any informational message. However, this
         * cannot be used for any kind of message that requires user input.
*         In our wizard the inline message is like Confirmation: Create Successful
         * followed by the Persistable object's name as hyperlink along with the
         * appropriate icon
         */

        /**
         * Overridden method of {@link DefaultObjectFormProcessor}. Used for inline
*         messaging.
         *
         * @return {@link
         WTMessage} */

        public WTMessage getSuccessMessageTitle() {
                return new WTMessage("com.ptc.core.ui.successMessagesRB",
                                "OBJECT_CREATE_SUCCESSFUL_TITLE", (Object[]) null);
        }

        /**
*         Method used to create WTDocument as per the user input.
         * @param name
*                 name of the {@link WTDocument} given by the user
         * @param number
         *                 of the {@link WTDocument} given by the user
         *
*         @param containerOid
*                 OID of {@link WTContainer} given by the user
*         @param folderOid
*                 OID of the {@link Folder} select by the user
         * @return the object of {@link Persistable}
         *
         * @throws WTException
         *                 throws {@link WTException}
         */
        private Persistable createDocument(String name, String number,
                        String containerOid, String folderOid) throws WTException {
                WTDocument doc = WTDocument.newWTDocument();
                try {
                        /*
                         * Setting the name and number of the WTDocument as per the user
                         * input
                         */
                        doc.setName(name);
                        doc.setNumber(number);
                        // doc.setDescription("My Description"); //method to set description
```

```
            doc.setContainer((WTContainer) fromReferenceToObject(containerOid));
                                    /* Setting the container as well as folder as selected by the user */
                                    assignFolder(doc, folderOid);
                                    /* Storing the WTDocument in DB */
                                    doc = (WTDocument) PersistenceHelper.manager.save(doc);
                            } catch (WTPropertyVetoException
                                    e) { e.printStackTrace();
                            } catch (WTException e) {
//                                      TODO Auto-generated catch
block e.printStackTrace();
                            }
                            /*
                             * Returning the WTDocument object. As WTDocument implements Persistable

                             * so we can use Persistable's object to hold the data.
                             */
                            return doc;
                    }

                    /**
*                      Create {@link WTPart} as per the input given by the user.
                     * @param name
                     *          name of the {@link WTPart}
                     *
                     * @param number
*                               number of the {@link WTPart}
*                      @param containerOid
*                               OID of the {@link WTContainer} select by the user
*                      @param folderOid
                     *          OID of the {@link Folder} given by the user
                     *
*                      @return {@link Persistable} object
                     * @throws WTException
                     *           throws {@link WTException}
                     */

                    private Persistable createPart(String name, String number,
                                    String containerOid, String folderOid) throws WTException {
                            WTPart part = WTPart.newWTPart();
                            try {
                                    part.setName(name);
                                    part.setNumber(number);

                                    part.setContainer((WTContainer)
                                    fromReferenceToObject(containerOid)); assignFolder(part, folderOid);

                                    /*
                                     * Storing the WTPart after creating and assigning different
                                     * attribute's value as per user input

                                     */
                                    part = (WTPart) PersistenceHelper.manager.save(part);

                            } catch (WTPropertyVetoException
                                    e) { e.printStackTrace();

                            } catch (WTException e) {

//                                      TODO Auto-generated catch
block e.printStackTrace();
                            }
                            return part;
                    }

                    /**
*                      Return the object from the given object reference.
*                      @param objectReference
                     *          The object reference
                     *
*                      @return The persistable object from the given reference
                     * @exception WTException
                     *              throws {@link WTException}
                     */

                    private Persistable fromReferenceToObject(String objectReference)
                                    throws WTException {
                            final ReferenceFactory factory = new ReferenceFactory();
                            final ObjectReference objRef = (ObjectReference) factory.getReference(objectReference);
                            return (objRef.getObject());
                    }
```

```
                /**
*                  Assign the given object to the given folder.
*                  @param folderEntry
                   *          The object which need to be assigned typically WTPart or
                   *          WTDocument
                   *
                   * @param folderOid
*                            The folder's object id in which the object will be assigned
*                  @exception WTException
                   *              throws {@link WTException}
                   */
                  private void assignFolder(FolderEntry folderEntry, String folderOid)
                                throws WTException {
                          /*
*                            Fetching the Folder object from the given folder
object id */
                          final Folder folder = ((Folder) fromReferenceToObject(folderOid));
                          FolderHelper.assignLocation(folderEntry, folder);
                  }
          }
```

Final Step:

Start/Restart server.



Now it's time to discuss about some possible loopholes in our implementation.

- User can skip the second step of the Wizard "Set Attribute" and click on "Finish" or "Apply". So, we have to force the user to click on "Next" before submitting the wizrd by clicking "Finish" or "Apply".
  - Steps to resolve
    Open "custom-actions.xml" and add required="true" at the time of entering the data for Second Step like this

```
<action name="setAttribute" type="itc" required="true">
        <command url="/netmarkets/jsp/itc/setAttribute.jsp" windowType="wizard_step" />
</action>
```

Now if your server is up and running then to incorporate the changes without restart ,

Open Windchill shell and run the command:

windchill com.ptc.netmarkets.util.misc.NmActionServiceHelper

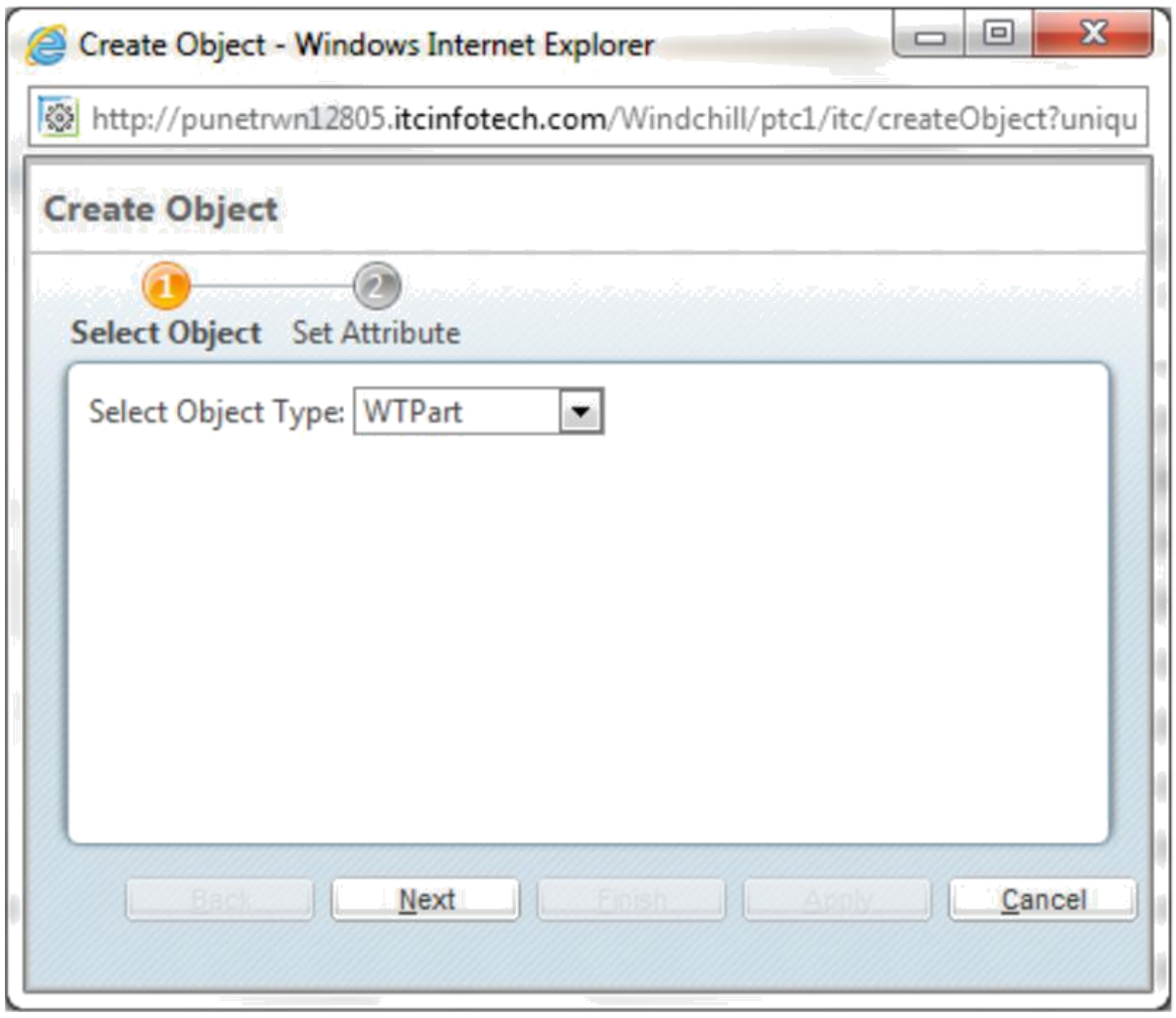
Check Method Server for possible compilation error. Sample of error log from method server

```
INFO : com.ptc.netmarkets.util.misc.StandardNmActionService.xmlInit  - Finished initializeXML()
INFO : com.ptc.netmarkets.util.misc.StandardNmActionService.xmlInit  - Started initializeXML()
INFO : wt.system.out - [Fatal Error] :17:4: The element type "command" must be terminated by the
 matching end-tag "</command>".
ERROR : com.ptc.netmarkets.util.misc.StandardNmActionService.XMLParsing  - could not read in t
he actionmodel file: config/actions/custom-actions.xml
wt.util.xml.XMLMechanismException: org.xml.sax.SAXParseException: The element
type "command" m ust be terminated by the matching end-tag "</command>".
Nested exception is: org.xml.sax.SAXParseException: The element type
"command" must be termina ted by the matching end-tag "</command>".
     at org.apache.xerces.parsers.DOMParser.parse(Unknown Source)
     at org.apache.xerces.jaxp.DocumentBuilderImpl.parse(Unknown Source)
     at javax.xml.parsers.DocumentBuilder.parse(DocumentBuilder.java:124)
     at wt.util.xml.NonValidatingXML4jAdapter.createDOMDocument(NonValidatingXML4jAdapter.j
ava:147)
     at wt.util.xml.XMLParser.createDOMDocument(XMLParser.java:193)
     at com.ptc.netmarkets.util.misc.StandardNmActionService.readFile(StandardNmActionServi
ce.java:1259)
     at com.ptc.netmarkets.util.misc.StandardNmActionService.getDOMRootsFromProperty(Standa
rdNmActionService.java:1187)
     at com.ptc.netmarkets.util.misc.StandardNmActionService.initializeXML(StandardNmAction
Service.java:3148)
     at com.ptc.netmarkets.util.misc.StandardNmActionService.reloadXML(StandardNmActionServ
ice.java:882)
     at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
     at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:39)
     at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:2
5)
     at java.lang.reflect.Method.invoke(Method.java:597)
     at wt.method.MethodResultWriter.writeExternal(MethodResultWriter.java:151)
     at wt.method.MethodResult.writeExternal(MethodResult.java:226)
     at java.io.ObjectOutputStream.writeExternalData(ObjectOutputStream.java:1426)
     at java.io.ObjectOutputStream.writeOrdinaryObject(ObjectOutputStream.java:1398)
     at java.io.ObjectOutputStream.writeObject0(ObjectOutputStream.java:1158)
     at java.io.ObjectOutputStream.writeObject(ObjectOutputStream.java:330)
     at sun.rmi.server.UnicastRef.marshalValue(UnicastRef.java:274)
     at sun.rmi.server.UnicastServerRef.dispatch(UnicastServerRef.java:313)
     at sun.rmi.transport.Transport$1.run(Transport.java:159)
     at java.security.AccessController.doPrivileged(Native Method)
     at sun.rmi.transport.Transport.serviceCall(Transport.java:155)
     at sun.rmi.transport.tcp.TCPTransport.handleMessages(TCPTransport.java:535)
```

A successful method server log

```
INFO : com.ptc.netmarkets.util.misc.StandardNmActionService.xmlInit  - Started initializeXML()

INFO : com.ptc.netmarkets.util.misc.StandardNmActionService.xmlInit  - Finished initializeXML()
```

If no other line is there in between that mean Compilation is successful. Its time to check the changes done by adding required=" true"



The Finish button will not enable until user visit the next page at least once

- To boost up the performance we can validate the user input in client side. As server side validation (From the Processor class) is always more expensive in terms of time and performance.
  For client side validation we need JavaScript.

  - Steps for Client side validation
    Add below lines in "selectObject.jsp"

```
<html>

<head>

<script>
function displayResult()
{
	var x=document.getElementById("Type").value;
	if(x == "Select a type")
		{
			alert(x + ' From the dropdown list provided');
			return false;
		}
	return true;
}
</script>

</head>

<body>

Select Object Type: <select name="Type" id="Type">
	<option>Select a type</option>
	<option>WTPart</option>
	<option>WTDocument</option>
</select>

</body>

</html>
```

  - In "setAttribute.jsp" add below javascript.

```
<script>
function checkResult()
{
var x=document.getElementById("number").value;
var y=document.getElementById("name").value;
var z=document.getElementById("loc_folder").value;
var a=document.getElementById("loc_container").value;
if(x == ""){
	alert('Kindly provide value for Number');

	return false;
				}
			else if(y == ""){
	alert('Kindly provide value for Name');
	return false;
	}
else if(z == "" || a ==""){
	alert('Kindly select the location after clicking the "Browse" button');
	return false;
	}
	return true;
}
</script>
```

- Add afterJS property in custom-actions.xml.

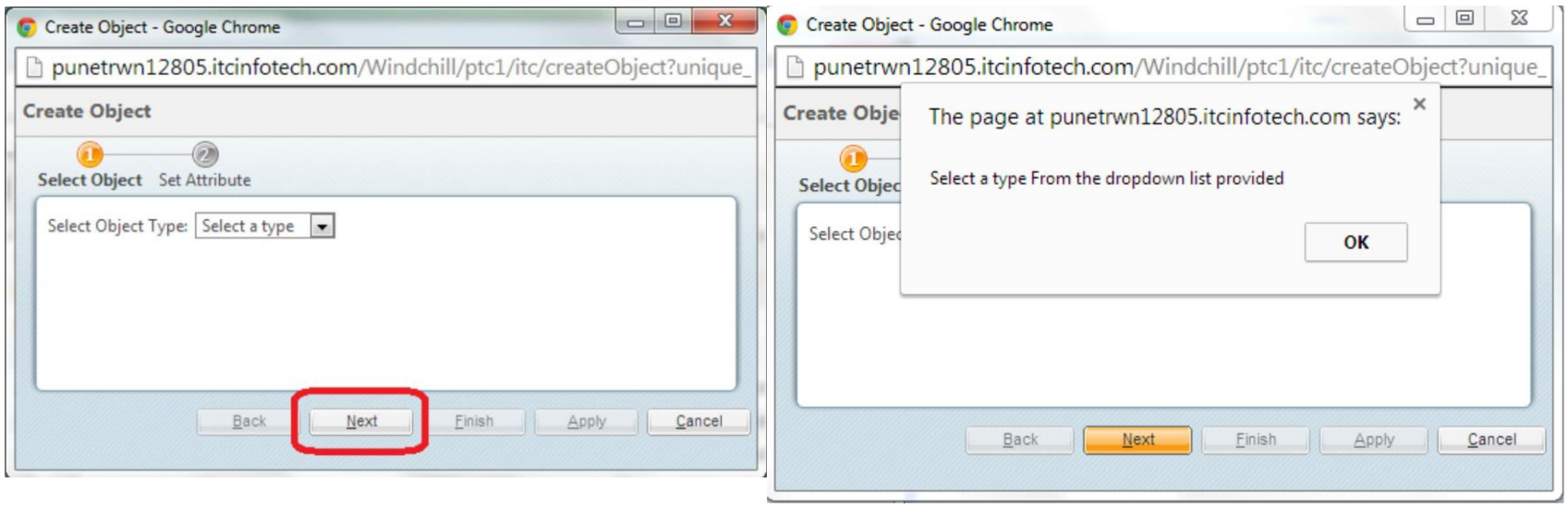
```
<action name="selectObject" type="itc" afterJS="displayResult">
		<command url="/netmarkets/jsp/itc/selectObject.jsp"
	windowType="wizard_step" /> </action>
	<action name="setAttribute" type="itc" required="true" afterJS="checkResult">
		<command url="/netmarkets/jsp/itc/setAttribute.jsp"
		windowType="wizard_step" />
	</action>
```

Using afterJS followed by the JS function name we can invoke that javascript on client side validation for a wizard step after the step is completed.

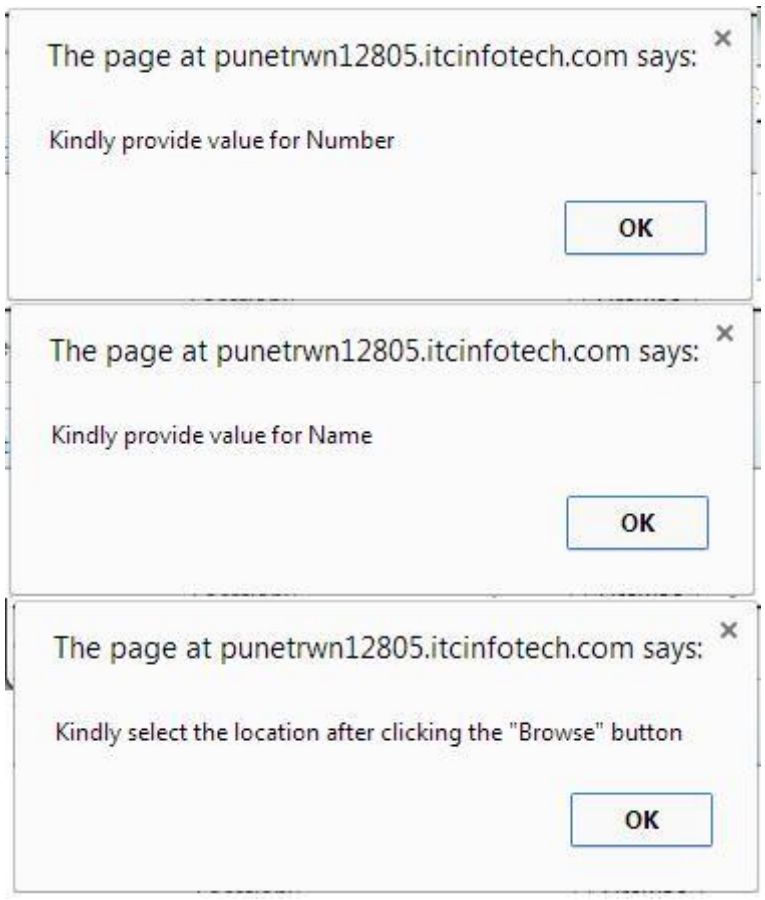
- After that we can comment out few lines from our processor as all the validations are from the client side now using javaScript.

```
		/*if (name == null || number == null) {
				* Checking whether user provide the name or Number if no then show
				* proper error message
				throw new WTException("Provide value for Name and Number");
			} else if (folderOid == null || containerOid == null) {
				Checking whether user select the location or not
				throw new WTException(
						"Select the location properly by clicking on Browse button kindly
						don't copy paste it");
			}*/
```

- Start/Restart method server to observe the changes.
  If user forgot to choose any valid type and click on next then the error message is

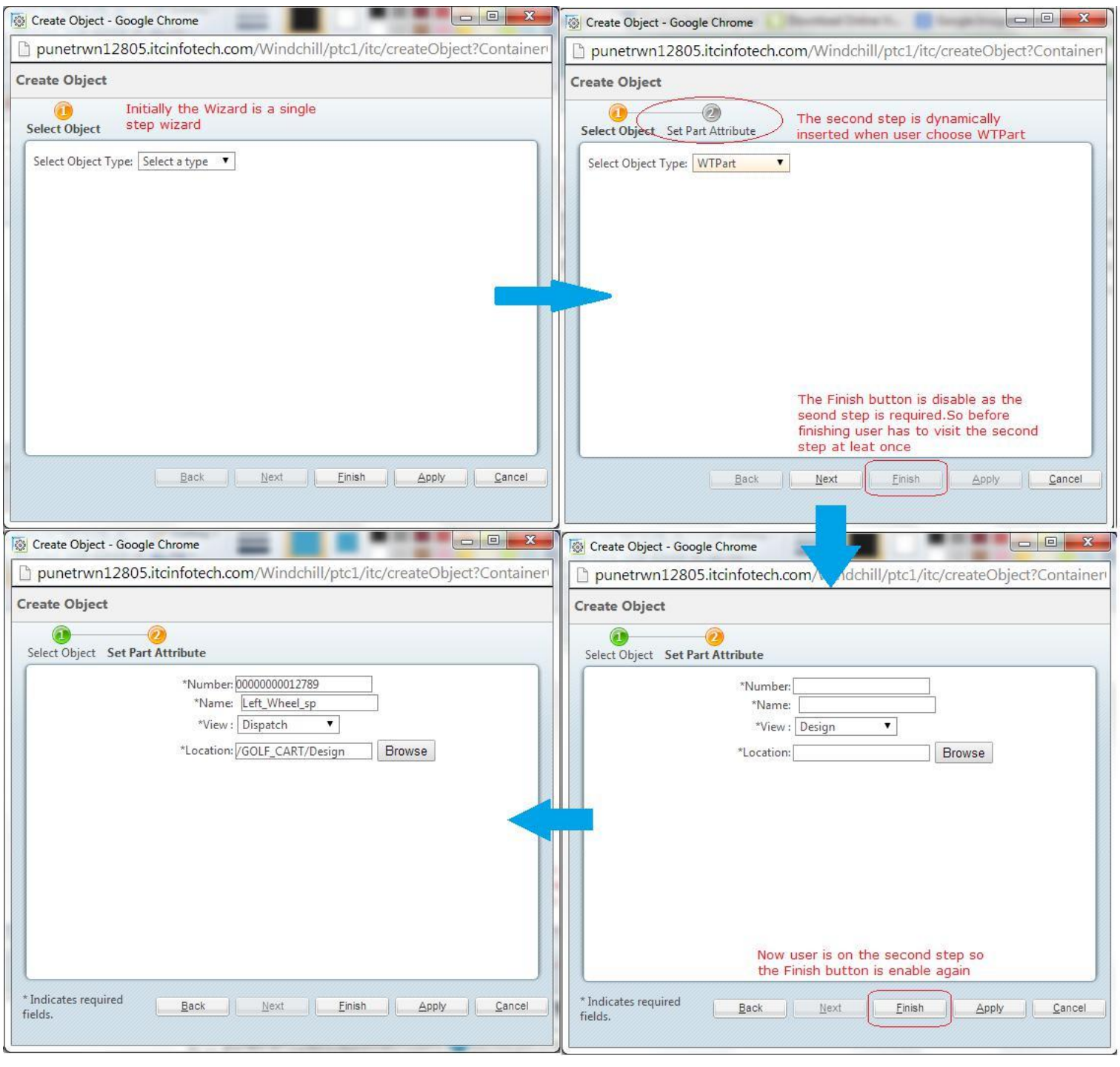


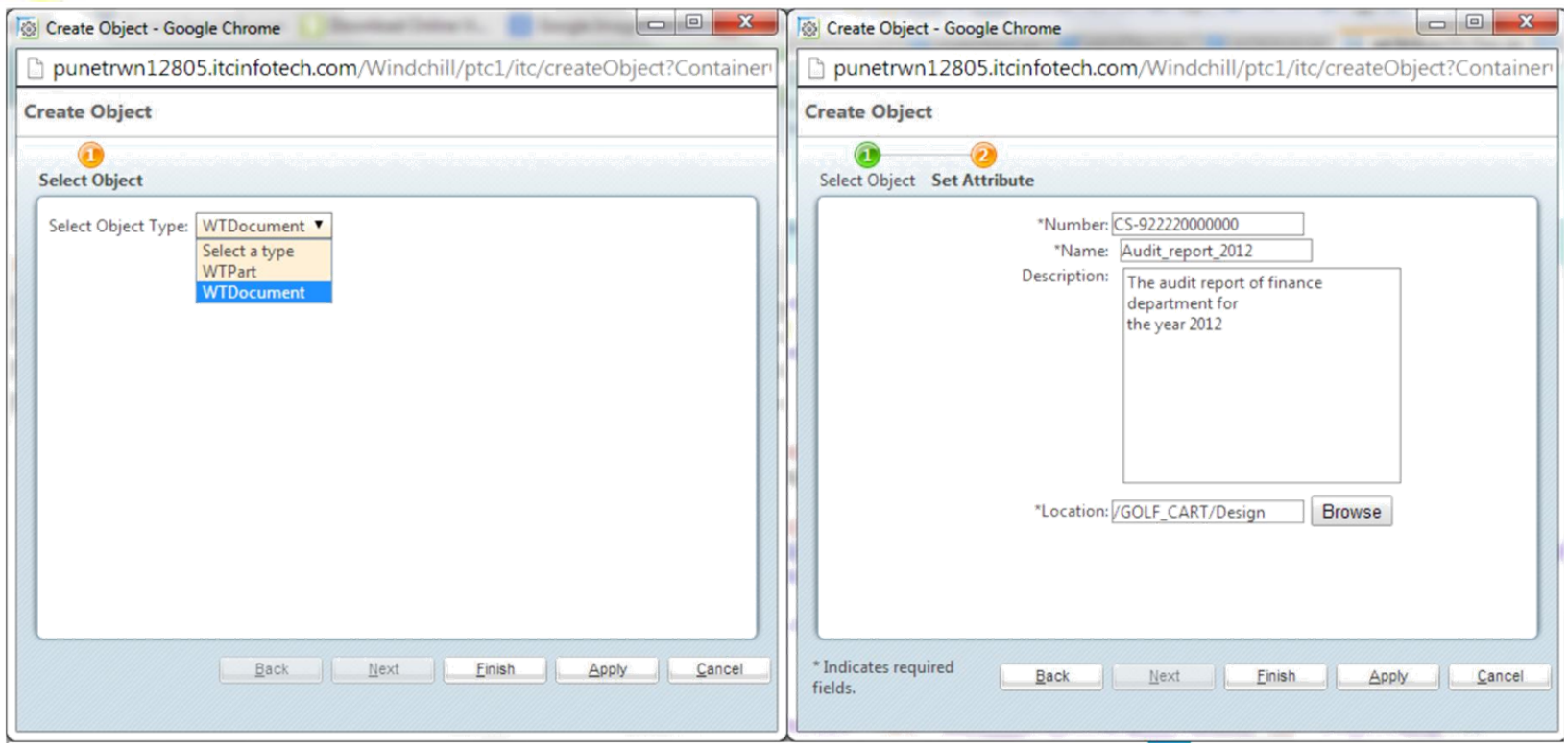
- If user forgot to give any value in the Second page then various error message will be displayed to the user.



3.2 Requirement II : Dynamically Add/Remove wizard step

This Requirement is pretty much similam to our previous one with fex extra enhancements.
The frist wizard step will be same.
Now if user choose to create WTPart the second wizard step will be inserted dynamically to get the attribute's value from user and there will be an extra attribute named "View" and if user choose to create "WTDocument" then another jsp will be inserted dynamically with an extra attribute named "description".





❖ **Step 1:**

For the solution we need another wizard step. (Action)
Add below lines in "CustomActionResource.java"

```
/** Set Attribute title **/
        @RBEntry("Set Part Attribute")
        public static final String SETATTRIBUTEWIZSTEP_TITLE = "itc.setAttributeWizStep.title";

        /** Set Attribute tool tip **/
        @RBEntry("Set Part Attribute")
        public static final String SETATTRIBUTEWIZSTEP_TOOLTIP = "itc.setAttributeWizStep.tooltip";

        /** Set Attribute description **/
        @RBEntry("Set Part Attribute")
        public static final String SETATTRIBUTEWIZSTEP_DESCRIPTION = "itc.setAttributeWizStep.description";
```

❖ **Step 2 :**

Modify "custom-actions.xml" as described below

```
<!DOCTYPE listofactions SYSTEM "actions.dtd">
<listofactions>
<objecttype name="itc"
        resourceBundle="ext.test.resource.CustomActionResource"> <!--
        Wizard to create document -->
        <action name="createObject" type="itc" >
                <command url="/netmarkets/jsp/itc/createObject.jsp"
class="ext.test.processor.SelectedObjectFormProcessor"
windowType="popup"/>
        </action>
        <action name="selectObject" type="itc" afterJS="displayResult">
                <command url="/netmarkets/jsp/itc/selectObject.jsp"
        windowType="wizard_step" /> </action>
        <action name="setAttribute" type="itc" hidden="true"
        afterJS="checkResult"> <!-- The step is hidden initially -->
                <command url="/netmarkets/jsp/itc/setAttribute.jsp"
        windowType="wizard_step" /> </action>
        <action name="setAttributeWizStep" type="itc" hidden="true" afterJS="checkVal">
                <command url="/netmarkets/jsp/itc/setAttributeWizStep.jsp" windowType="wizard_step"
                />
        </action>
</objecttype>
</listofactions>
```

❖ **Step 3 :**

"createObject.jsp"

```
<%@ taglib prefix="jca"
uri="http://www.ptc.com/windchill/taglib/components" %> <%@include
file="/netmarkets/jsp/components/beginWizard.jspf"%> <%@include
file="/netmarkets/jsp/components/includeWizBean.jspf"%> <jca:wizard
title="Create Object">
      <jca:wizardStep action="selectObject" type="itc"/>
                  <jca:wizardStep action="setAttribute" type="itc"/>
                  <jca:wizardStep action="setAttributeWizStep" type="itc"/>
 </jca:wizard>
<%@include file="/netmarkets/jsp/util/end.jspf"%>
```

"selectObject.jsp"

```
<html><head><script>
function checkResult()
{
        var fold=document.getElementById("loc_folder").value;
        var loc=document.getElementById("loc_container").value;
         if(fold == "" || loc == ""){
                alert('Kindly select the location after clicking the "Browse" button');
                return false;
                }
        return true;
}
function checkVal()
{
        var fold=document.getElementById("loc_folderp").value;
        var loc=document.getElementById("loc_containerp").value;
         if(fold == "" || loc == ""){
                alert('Kindly select the location after clicking the "Browse" button');
                return false;
                }
        return true;
}
function displayResult()
{
var x=document.getElementById("Type").value;
if(x == "Select a type")
{
        alert(x + ' From the dropdown list provided');
        return false;
}
return true;
}
/**
 * add a new step to the list of steps based on the user input(dynamic step) and make it required also */
function insertNext(astepId){
        wizardSteps[astepId].hidden = false;
        PTC.wizard.StepLinksPanelUtil.repaintStepPanel();
   PTC.wizard.StepLinksPanelUtil.attachOnClickHandlers();
   resetOkButton();
   resetNextBackButton(currentStepStrName);
        if (wizardSteps[astepId].required === false) {
                wizardSteps[astepId].required = true;
      wizardSteps[astepId].isComplete = false;
      resetOkButton();
   }
}
/**remove a step */
function removeNext(astepId){
        if(wizardSteps[astepId].hidden === false){
        wizardSteps[astepId].hidden = true;
   PTC.wizard.StepLinksPanelUtil.repaintStepPanel();
   PTC.wizard.StepLinksPanelUtil.attachOnClickHandlers();
   resetOkButton();
   resetNextBackButton(currentStepStrName);
   }
}
function chooseNext(val){
 if(val == "WTDocument"){
        insertNext('itc.setAttribute'); // The step id is action_type.action_name
        removeNext('itc.setAttributeWizStep');
        }
        else if(val != "Select a type"){
        insertNext('itc.setAttributeWizStep');
        removeNext('itc.setAttribute');
        }
}
</script></head><body>
Select Object Type: <select name="Type" id="Type"
        onChange="chooseNext(document.getElementById('Type').value)"> <option>Select a type</option>
  <option>WTPart</option>
  <option>WTDocument</option>
</select></body></html>
```

"setAttribute.jsp"

```
<%@ include file="/netmarkets/jsp/components/beginWizard.jspf" %>
<%@ taglib prefix="w" uri="http://www.ptc.com/windchill/taglib/wrappers"%> <table>
<tr>
<td scope="row" width="200" align="right">
*Number:
</td>
<td align="left">
<w:textBox name="number" id="number" maxlength="30" required="true"/>
</td>
</tr>
<tr>
<td scope="row" width="200" align="right">
*Name:
</td>
<td align="left"> 
<w:textBox name="name" id="name" maxlength="30" required="true"/>
</td>
<!--  Both input type textbox and w:textBox will create a text box for user input.
One is using normal HTML tag another is from Windchill Tag Library thats the only difference-->
</tr>
<tr>
<td scope="row" width="200" align="right" valign="top">
Description:
</td>
<td align="left"> 
<w:textArea name="description" id="description" cols="30" maxLength="10000" rows="10"/>
</td>
</tr>
<tr>
<td scope="row" width="200" align="right">
*Location:
</td>
<td align="left">
<input type="text" name="loc_displayLocation" id="loc_displayLocation" readonly> <button
onclick="ITC.launchFolderPicker(event)" type="button">
Browse
</button>
</td>
</tr>
</table>
<input type="hidden" name="null___loc_folder___textbox" id="loc_folder" />
<input type="hidden" name="null___loc_container___textbox" id="loc_container" />
<script type="text/javascript">
<!--
//namespace for paste example to avoid colliding with any globally defined functions -->
ITC = {};
ITC.launchFolderPicker = function(event) {
var url = getBaseHref() +
'servlet/WindchillAuthGW/wt.enterprise.URLProcessor/invokeAction?action=cadxBrowseLocations&containerVisibilityMa
sk=PDMLink&accessPermission=modify&displayHotlinks=false&displayCreateFolder=true'; launchBrowseFolders(url,
$('loc_displayLocation'), ITC.locationCallback);
Event.stop(event);
return false;
}
ITC.locationCallback = function(contextValues) { var containerOID = contextValues.containerOID; var folderOID =
contextValues.folderOID;
$('loc_folder').value = folderOID;   // 'OR:wt.pdmlink.PDMLINKProduct:123'
$('loc_container').value = containerOID; // 'OR:wt.folder.SubFolder:123'
}
</script>
<%@include file="/netmarkets/jsp/util/end.jspf"%>
```

"setAttributeWizStep.jsp"

```
<%@ include file="/netmarkets/jsp/components/beginWizard.jspf" %>
<%@ page import="wt.vc.views.ViewHelper" %>
<%@ page import="wt.vc.views.View" %>
<%@ page import="java.util.ArrayList" %>
<%@ taglib prefix="w" uri="http://www.ptc.com/windchill/taglib/wrappers"%>
<table>
<tr>
<td scope="row" width="200"  align="right">
*Number:
</td>
<td align="left">
<w:textBox name="numberp" id="numberp" maxlength="30" required="true"/>
</td>
</tr>
<tr>
<td scope="row" width="200"  align="right">
*Name:
</td>
<td align="left"> 
<w:textBox name="namep" id="namep" maxlength="30" required="true"/>
</td>
<!--    Both input type textbox and w:textBox will create a text box for user input.
One is using normal HTML tag another is from Windchill Tag Library thats the
only difference-->
</tr>
<tr>
<td scope="row" width="200"  align="right">
*View :
</td>
<td align="left">
<select name="null___view___textbox" id="view">
<%
View[] views = ViewHelper.service.getAllViews();
ArrayList<String> viewNames = new ArrayList<String>();
for(View view : views)
{
viewNames.add( view.getName());
}
for(String str : viewNames)
{
%>
<option><%=str%></option>
<%}%>
</select>
</td></tr>
<tr>
<td scope="row" width="200"  align="right">
*Location:
</td>
<td align="left">
<input type="text" name="loc_displayLocationp" id="loc_displayLocationp" readonly> <button onclick="KD.launchFolderPicker(event)"
type="button">
Browse
</button>
</td>
</tr>
</table>
<input type="hidden" name="null___loc_folderp___textbox" id="loc_folderp" />
<input type="hidden" name="null___loc_containerp___textbox" id="loc_containerp" />
<script type="text/javascript">
<!--
//namespace for paste example to avoid colliding with any globally defined functions --> KD = {};
KD.launchFolderPicker = function(event) {
var url = getBaseHref() +
'servlet/WindchillAuthGW/wt.enterprise.URLProcessor/invokeAction?action=cadxBrowseLocations&containerVisibilityMask=PDMLink&accessPer
mission=
modify&displayHotlinks=false&displayCreateFolder=true';
launchBrowseFolders(url, $('loc_displayLocationp'), KD.locationCallback);
Event.stop(event);
return false;
}
        KD.locationCallback = function(contextValues) {
        var containerOID = contextValues.containerOID;  // 'OR:wt.pdmlink.PDMLINKProduct:123'
        var folderOID = contextValues.folderOID;  // 'OR:wt.folder.SubFolder:123'
        $('loc_folderp').value = folderOID;
        $('loc_containerp').value = containerOID;
}
        </script>
        <%@include file="/netmarkets/jsp/util/end.jspf"%>
```

Changes made in the previous processor class

```
@SuppressWarnings("rawtypes")
@Override
public FormResult doOperation(NmCommandBean commandBean, List<ObjectBean> list) throws
WTException {
/*
*     Fetch the data given by the user. If you know the exact parameter
*     name then you may use this method
*/
HttpServletRequest request = commandBean.getRequest();
String type = request.getParameter("Type");
/*
*     This is another method to fetch user input */
HashMap map = commandBean.getText();
Object[] keys = map.keySet().toArray();
String name = null;
String number = null;
String containerOid = null;
String folderOid = null;
String desCripTion = null;
String viewName = null;
if (type.toLowerCase().contains("wtpart")) {
name = ((String) map.get("namep")).toUpperCase();
number = ((String) map.get("numberp")).toUpperCase();
folderOid = ((String) map.get("loc_folderp"));
containerOid = ((String) map.get("loc_containerp"));
viewName = ((String) map.get("view"));
} else {
name = ((String) map.get("name")).toUpperCase();
number = ((String) map.get("number")).toUpperCase();
folderOid = ((String) map.get("loc_folder"));
containerOid = ((String) map.get("loc_container"));
}
/* To extract textArea's value there are two methods */
/* **********************1************************** */
/*Enumeration enumer = request.getParameterNames();
while (enumer.hasMoreElements()) {
String nameObj = enumer.nextElement().toString();
if (nameObj.contains("description") && !(nameObj.endsWith("old"))) { desCripTion =
request.getParameter(nameObj);
}
}*/
/* **********************2************************** */
HashMap mapArea = commandBean.getTextArea(); desCripTion = ((String)
mapArea.get("description"));
Persistable per = null;
if (type.toLowerCase().contains("part")) {
per = createPart(name, number, containerOid, folderOid, viewName); } else if
(type.toLowerCase().contains("document")) {
per = createDocument(name, number, desCripTion, containerOid, folderOid);
}
ObjectBean ob = ObjectBean.newInstance();
ob.setObject(per);
list.add(ob);
/* Adding the Persistable object in the List */
return super.doOperation(commandBean, list);
}
```

```
private Persistable createDocument(String name, String number,
                String desCripTion, String containerOid, String folderOid)
                throws WTException {
        WTDocument doc = WTDocument.newWTDocument();
        try {
                /*
                 * Setting the name and number of the WTDocument as per the user
                 * input
                 */
                doc.setName(name);
                doc.setNumber(number);
                if (desCripTion != null) {
                        doc.setDescription(desCripTion); // method to set description
                }
                doc.setContainer((WTContainer)
                fromReferenceToObject(containerOid));
                /* Setting the container as well as folder as selected by the user */
                assignFolder(doc, folderOid);
                /* Storing the WTDocument in DB */
                doc = (WTDocument) PersistenceHelper.manager.save(doc);
        } catch (WTPropertyVetoException e)
                { e.printStackTrace();
        } catch (WTException e) {
                // TODO Auto-generated catch
                block e.printStackTrace();
        }
        /*
                 }
                 /*
                  * Returning the WTDocument object. As WTDocument implements Persistable
                  * so we can use Persistable's object to hold the data.
                  */
                 return doc;
         }
         /**
          *Create {@link WTPart} as per the input given by the user.
          * @param name
          *          name of the {@link WTPart}
          *
          * @param number
*                 number of the {@link WTPart}
          * @param containerOid
*                 OID of the {@link WTContainer} select by the user
          * @param folderOid
          *          OID of the {@link Folder} given by the user
          *
*   @param viewName
*   the name of the View
*   @return {@link Persistable} object
*   @throws WTException

*   throws {@link WTException}
          */
         private Persistable createPart(String name, String number,

                         String containerOid, String folderOid, String viewName)
                         throws WTException {
                 WTPart part = WTPart.newWTPart();
                 try {

                         part.setName(name);
                         part.setNumber(number);
                         part.setContainer((WTContainer) fromReferenceToObject(containerOid));
                         assignFolder(part, folderOid);
                         part.setView(ViewReference.newViewReference(ViewHelper.service
                                         .getView(viewName)));
                         /*

*   Storing the WTPart after creating and assigning different
*   attribute's value as per user input
                          */

                         part = (WTPart) PersistenceHelper.manager.save(part);

                 } catch (WTPropertyVetoException e)
                         { e.printStackTrace();
                 } catch (WTException e) {

//   TODO Auto-generated catch block e.printStackTrace();
                 }
                 return part;
         }
```

Start/Restart and observe the changes.

3.3 Requirement III : Persist Multi level attribute value

In previous section we learn how to create a Multi-Valued dropdown list as an input parameter. Now we will learn how to persist the value using processor.
Create a custom processor for WTDocument to persist the multi-valued IBA's value. Assume the IBA's internal name is VendorName and it's only in a specific subtype of WTDocument whose internal name is com.ITCINFOTECH.Agenda.
For create mode wrote the following code

```
package ext.customization.processor;
import java.util.Arrays;
import java.util.Enumeration;
import java.util.HashSet;
import java.util.Iterator;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import wt.doc.WTDocument;
import wt.fc.ObjectReference;
import wt.fc.Persistable;
import wt.fc.PersistenceHelper;
import wt.folder.Folder;
import wt.util.WTException;
import com.ptc.core.components.beans.ObjectBean;
import com.ptc.core.components.forms.FormResult;
import com.ptc.core.lwc.server.LWCNormalizedObject;
import com.ptc.core.meta.common.TypeIdentifierHelper;
import com.ptc.core.meta.common.UpdateOperationIdentifier;
import com.ptc.netmarkets.util.beans.NmCommandBean;
import com.ptc.windchill.enterprise.doc.forms.CreateDocFormProcessor; import
com.ptc.windchill.wp.delivery.DeliveryRecord;
public class CustomDocProcessor extends CreateDocFormProcessor
{
	@SuppressWarnings({ "rawtypes", "unchecked" })
	@Override
	public FormResult postProcess(NmCommandBean paramNmCommandBean,
			List<ObjectBean> paramList) throws WTException
	{
		/* Taking the object of WTDocument which is created form the CreateDocFromProcessor*/
		WTDocument localWTDocument = null;
		for (Iterator localIterator = paramList.iterator(); localIterator.hasNext(); ) localWTDocument =
			(WTDocument)(((ObjectBean)localIterator.next()).getObject());
		try{		/* Checking whether the WTDocument is of desired subtype */
			if(( TypeIdentifierHelper.getType(localWTDocument).toString().equals("WCTYPE|wt.doc.WTDocument|
com.ITCINFOTECH.Agenda "))//internal name of your subtype
			{
				HttpServletRequest req = paramNmCommandBean.getRequest();
				Enumeration em =  req.getParameterNames();
				String temp = null;
				while(em.hasMoreElements())
				{
					String name="VendorName";//internal name of your iba
					String old = "old";
					temp = em.nextElement().toString();
					if(temp.contains(name) && !temp.endsWith(old))
						break;
				}
				//Giving value to the soft attribute
				LWCNormalizedObject obj = new LWCNormalizedObject(localWTDocument,null,null,new
UpdateOperationIdentifier());
				//removing duplicate value
				obj.load("VendorName");
				obj.set("VendorName",(String[])new HashSet( Arrays.asList( req.getParameterValues(temp) ) ).toArray(
new String[]{} ));
				/*persisting the value of IBA after removing duplicate value*/
				obj.apply();
				PersistenceHelper.manager.modify(localWTDocument);
			}
		}

		catch(WTException e)
		{
			e.printStackTrace();

		}
		// TODO Auto-generated method stub
		return super.postProcess(paramNmCommandBean, paramList);
	}

}
```

Now for edit mode.

```
package ext.customization.processor;
import java.util.*;
import javax.servlet.http.HttpServletRequest;
import wt.doc.WTDocument;
import wt.fc.PersistenceHelper;
import wt.util.WTException;
import com.ptc.core.components.beans.ObjectBean;
import com.ptc.core.components.forms.EditWorkableFormProcessor;
import com.ptc.core.components.forms.FormResult;
import com.ptc.core.lwc.server.LWCNormalizedObject;
import com.ptc.core.meta.common.TypeIdentifierHelper;
import com.ptc.core.meta.common.UpdateOperationIdentifier;
import com.ptc.netmarkets.util.beans.NmCommandBean;

public class CustomEditDocProcessor extends EditWorkableFormProcessor {
	@SuppressWarnings("rawtypes")
	@Override
	public FormResult postProcess(NmCommandBean paramNmCommandBean,
			List<ObjectBean> paramList) throws WTException
	{
		WTDocument localWTDocument = null;
		for (Iterator localIterator = paramList.iterator(); localIterator.hasNext(); ) localWTDocument =
			(WTDocument)(((ObjectBean)localIterator.next()).getObject());
		/* Taking the object of WTDocument which is created form  the EditWorkableFormProcessor*/
		try
		{
			if( (
TypeIdentifierHelper.getType(localWTDocument).toString().equals("WCTYPE|wt.doc.WTDocument|com.ITCINFOTECH.VendorPart")))
			{
				HttpServletRequest req = paramNmCommandBean.getRequest();
				Enumeration em =  req.getParameterNames();
				String temp = null;
				while(em.hasMoreElements())
				{
					String name="VendorName";
					String old = "old";
					temp = em.nextElement().toString();
					if(temp.contains(name) && !temp.endsWith(old))
						break;
				}
				//Giving value to the soft attribute
				LWCNormalizedObject obj = new LWCNormalizedObject(localWTDocument,null,null,new
UpdateOperationIdentifier());
				@SuppressWarnings({ "unchecked" })
				String[] s  = (String[])new HashSet( Arrays.asList( req.getParameterValues(temp) ) ).toArray( new
String[]{} );

				List<String> list = new ArrayList<String>();
			for(String str : s)
			{

			  if(str != null && str.length() > 0)
			  {
			    list.add(str);

			  }
			}
			s = list.toArray(new String[list.size()]);
				/* After removing any blank string from String array */

				obj.load("VendorName");
		obj.set("VendorName",s);

		obj.apply();
				PersistenceHelper.manager.modify(localWTDocument);
				  }
			}

			catch(WTException e)
			{
				e.printStackTrace();
			}

				// TODO Auto-generated method stub
				return super.postProcess(paramNmCommandBean, paramList);

			}
	}
```

After that open the "custom-actions.xml" and add the following lines between < listofactions> tag and restart your method server.

```
<objecttype class="wt.doc.WTDocument" name="document"
resourceBundle="com.ptc.windchill.enterprise.doc.documentRe
source" >


    <action ajax="row" dtiUpload="true" name="create" uicomponent="CREATE_DOC"
        > <command class="ext.customization.processor.CustomDocProcessor"
        method="execute"


onClick="validateCreateLocation(event)"
        windowType="popup"/> <includeFilter
        name="projectM4D"/>

        <includeFilter name="showNewActionPreValidation"/>

    </action>



       <action ajax="row" dtiUpload="true" name="edit" >

        <includeFilter name="sandboxSharingValidationSimple"/>


        <includeFilter name="hideSBHiddenFilter"/>

        <includeFilter name="hideForClashJobDefinition"/>


        <command class="ext.customization.processor.CustomEditDocProcessor"
method="execute" windowType="popup"/>


    </action>



       <action ajax="row" dtiUpload="true"
        name="checkoutAndEdit" > <includeFilter
        name="sandboxSharingValidationSimple"/>
        <includeFilter name="hideSBHiddenFilter"/>

        <includeFilter name="hideForClashJobDefinition"/>


        <command class="ext.customization.processor.CustomEditDocProcessor" method="execute"
onClick="onClickValidation(event,'checkout')" url="netmarkets/jsp/document/edit.jsp"
windowType="popup"/>


    </action>


</objecttype>
```

If you want to does the same thing for any other object like WTPart then for creation you should extend "CreatePartAndCADDocFormProcessor" and for edit purpose you should extend

"com.ptc.core.components.forms.EditWorkableFormProcessor".

4. SERVICE – LISTENER IMPLEMENTATION

Windchill services provide APIs and logic to manage modeled business objects.

They consist of:

- An (optional) helper consisting of static fields and methods, including a static field (generally named service or manager) referring to a service
- A service interface consisting of remotely-invocable method declarations
- A standard service which implements the service and is registered to run as a

service in the Method Server.

4.1 Requirement I : Listen for Container creation

Create a service that listen for a Project Creation event and create a document with same name in that project.

❖ Step 1
Create an Interface "CreateProjectListener"

❖ Step 2
Create a service class "CreateProjectListenerService"

❖ Step 3
Register the Listener in site.xconf

```
<!-- Listener service  -->
<Property name="wt.services.service.200019" overridable="true"
targetFile="codebase/wt.properties"
value="ext.customization.listener.CreateProjectListener/ext.customization.listener.CreateProjectListenerService"/>
```

The number after "wt.services.service." can be any number but it must be unique for every listener service.

Note :

Like "WTContainerServiceEvent" class there are many event classes.
Ex. ChangeService2Event , ContainerMoveEvent , ContainerTeamServiceEvent , ContentServiceEvent, EPMDocumentManagerEvent, FolderServiceEvent, LifeCycleServiceEvent, ProjectManagementEvent, WorkInProgressServiceEvent, PersistenceManagerEvent etc.

4.2 Requirement II : Listen for percentage change in Activity or Deliverable

Create a lestener to listen the percent change of Plan and change the project phase accordingly.

For an example 1-20% the phase will be Kickoff (Default) for next 20% the phase will be Planning , next 20% it will be Assignment , next 20% it will be Development, next 10% (81-90) it will be Analysis(Custom ProjectPhase added using enumCustomize utility) and finally when it reaches 100% the phase will be Completed.

There are many ways to achieve the above requirement.One of them is to implement a listener to listen the percent change of PlanActivity and fetch the corresponding Plan and according to the percent complete change the Project Phase.

❖ Step 1:

Create listener DeliverablePercentChangeListener and service class named DeliverablePercentChangeListenerService.

❖ Step 2:

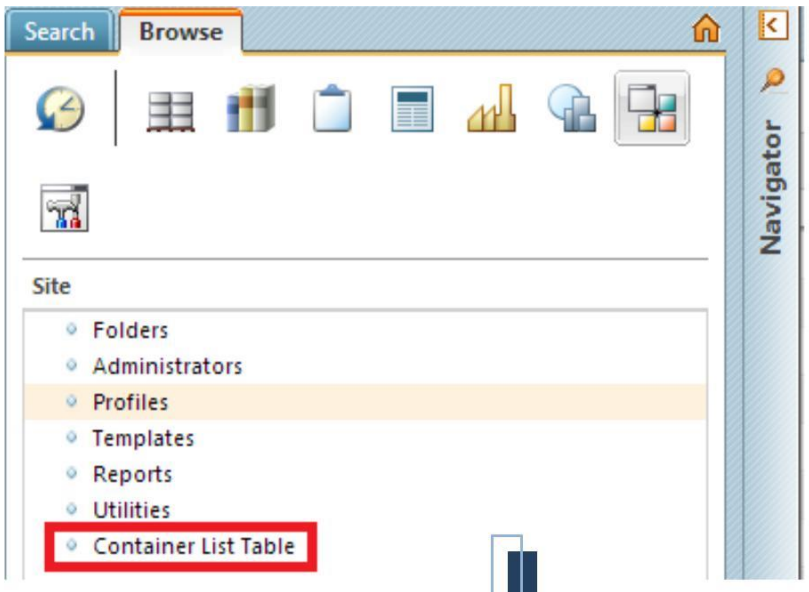
Register the listener and restart all the server.

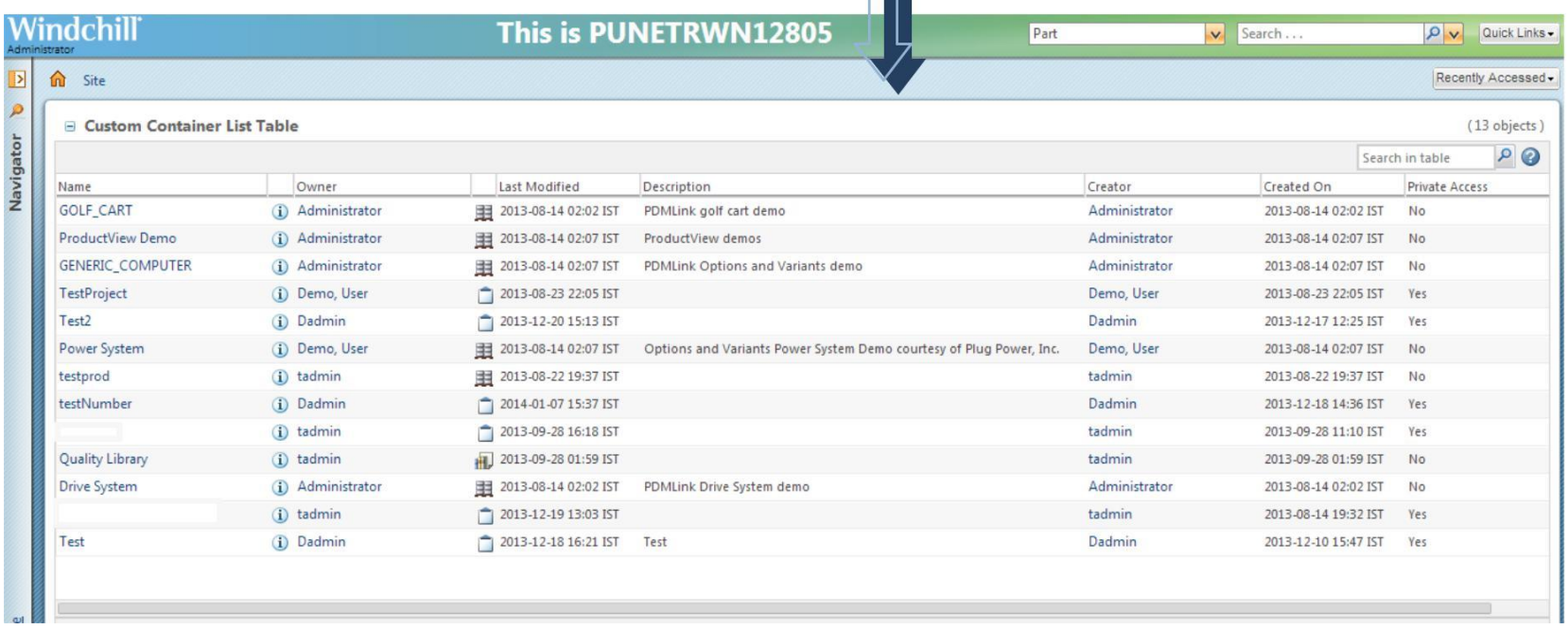
5. TABLE BUILDER

5.1 Requirement I: Create a simple table builder

We will make MVC table like following process

- Register custom dispatcher
- Register custom MVC component library
- Create MVC table class for display product, project and library
- Register MVC ID to action





Name	Owner	Last Modified	Description	Creator	Created On	Private Access
GOLF_CART	Administrator	2013-08-14 02:02 IST	PDMLink golf cart demo	Administrator	2013-08-14 02:02 IST	No
ProductView Demo	Administrator	2013-08-14 02:07 IST	ProductView demos	Administrator	2013-08-14 02:07 IST	No
GENERIC_COMPUTER	Administrator	2013-08-14 02:07 IST	PDMLink Options and Variants demo	Administrator	2013-08-14 02:07 IST	No
TestProject	Demo, User	2013-08-23 22:05 IST		Demo, User	2013-08-23 22:05 IST	Yes
Test2	Dadmin	2013-12-20 15:13 IST		Dadmin	2013-12-17 12:25 IST	Yes
Power System	Demo, User	2013-08-14 02:07 IST	Options and Variants Power System Demo courtesy of Plug Power, Inc.	Demo, User	2013-08-14 02:07 IST	No
testprod	tadmin	2013-08-22 19:37 IST		tadmin	2013-08-22 19:37 IST	No
testNumber	Dadmin	2014-01-07 15:37 IST		Dadmin	2013-12-18 14:36 IST	Yes
	tadmin	2013-09-28 16:18 IST		tadmin	2013-09-28 11:10 IST	Yes
Quality Library	tadmin	2013-09-28 01:59 IST		tadmin	2013-09-28 01:59 IST	No
Drive System	Administrator	2013-08-14 02:02 IST	PDMLink Drive System demo	Administrator	2013-08-14 02:02 IST	No
	tadmin	2013-12-19 13:03 IST		tadmin	2013-08-14 19:32 IST	Yes
Test	Dadmin	2013-12-18 16:21 IST	Test	Dadmin	2013-12-10 15:47 IST	Yes

❖ Step 1 Register MVC builder path

Open <*Windchill_Home*>/*codebase/WEB-INF/MVCDispatcher-servlet.xml* and add following

```
<import resource="classpath:config/mvc/mvc.xml" />
<import resource="classpath:config/mvc/jca-mvc.xml" />
<import resource="classpath:config/mvc/*-configs.xml" />
<import resource="classpath:config/mvc/custom.xml" />
<import resource="classpath:config/mvc/itc-custom.xml" />
```

Copy <*Windchill_Home*>/*codebase/config/mvc/custom.xml* and rename "*itc-custom.xml*"

Open <*Windchill_Home*>/*codebase/config/mvc/itc -custom.xml* and add following

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:mvc="http://www.ptc.com/schema/mvc"
    xsi:schemaLocation="
http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-2.5.xsd
http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-2.5.xsd
http://www.ptc.com/schema/mvc http://www.ptc.com/schema/mvc/mvc-10.0.xsd">
<!-- Base package is the package name of my custom builder classes -->
<mvc:builder-scan base-package="ext.test.builder"/>
      <bean class="ext.test.builder.ContainerListBuilder"/>
</beans>
```

❖ Step 2 Create MVC Component

Create class ContainerListBuilder.

❖ Step 3 Create Action

Add the following lines in "CustomActionResource.java"

```
/** Container List Table title **/
@RBEntry("Container List Table")
public static final String CONTAINER_LIST_TITLE = "itc.containerlist.title";

/** Container List Table tool tip **/
@RBEntry("Container List Table")
public static final String CONTAINER_LIST_TOOLTIP = "itc.containerlist.tooltip";

/** Container List Table description **/
@RBEntry("Container List Table")
public static final String CONTAINER_LIST_DESCRIPTION = "itc.containerlist.description";
```

Open custom-actionModels.xml and add below lines.

```
<model name="site navigation">
<description>Sub tabs under the site main tab</description>
<action name="listFiles" type="site"/>
<action name="listAdmin" type="site"/>
<action name="listProfiles" type="site"/>
<action name="listTemplates" type="site"/>
<action name="reports" type="site"/>
<action name="listAgreements" type="agreements"/>
<action name="listUtilities" type="site"/>
<action name="containerlist" type="itc" />
</model>
```

Open "custom-actions.xml" and add below lines

```
<action name="containerlist" type="itc" >
        <component windowType="page" name="ext.test.builder.ContainerListBuilder"/>
</action>
```

Restart all the services.

5.2 Requirement II : Example of separate builders for single table

This example is explaining the concept of separate builder class for one table and including tablebuilder in JSP.

We want to create an action in "folderbrowser_toolbar_actions" clicking which a jsp will pop-up to show us all the EPMDoucment and WTDocument in that particular product from which the action is invoked, in tabular format.

❖ Step 1 Create Custom Action and JSP

Open "custom-actions.xml" and add below lines

```
<objecttype name="itc"
      resourceBundle="ext.test.resource.CustomActionResource"> <action
      name="customtable" type="itc" >
            <command url="/netmarkets/jsp/itc/demoTable.jsp" windowType="popup"/>
      </action>
</objecttype>
```

Open "custom-actionModels.xml" and inside "folderbrowser_toolbar_actions" model name add below lines.

<action name="customtable" type="itc" shortcut="true"/>

Create a JSP "<Windchill_Home>\codebase\netmarkets\jsp\itc\demoTable.jsp" with the following content.

```
<%@include file = "/netmarkets/jsp/components/beginWizard.jspf" %>
 <%@ taglib uri="http://www.ptc.com/windchill/taglib/mvc"
prefix="mvc"%>
 <jsp:include page="${mvc:getComponentURL('custom.table')}" />
 <%@include file = "/netmarkets/jsp/util/end.jspf"%>
```

Open "CustomActionResource.java" and add below lines.

```
/** Custom Table title **/ //Add these lines in CustomActionResource for our custom action
        @RBEntry("Custom Table")
        public static final String CUSTOM_TABLE_TITLE = "itc.customtable.title";

        /** Custom Table tool tip **/
        @RBEntry("Custom Table")
        public static final String CUSTOM_TABLE_TOOLTIP = "itc.customtable.tooltip";

        /** Custom Table description **/
        @RBEntry("Custom Table")
        public static final String CUSTOM_TABLE_DESCRIPTION = "itc.customtable.description";

        /** Create Object Icon **/
        @RBEntry("dataMonitor.png")
        // The image is in netmarkets/images
        @RBPseudo(false)
        public static final String CUSTOM_TABLE_ICON = "itc.customtable.icon";
```

- **Step 2 the builder class**

Create "ShowPartAndCADDocumentTableBuilder.java" and "PartAndCADDocumentBuilder.java"

Restart all servers.

"ShowPartAndCADDocumentTableBuilder" is responsible for Config and the "PartAndCADDocumentBuilder" is responsible for Data only. Both the classes as well as the jsp should have same component id.

5.3 Requirement III: Multi Level BOM Report using Custom Bean Object

Want to create a Custom action in "more parts actions" model (Which is visible in every part's information page) and attach a JSP with that action.
In that JSP there will be a table which gives you a Multi Level BOM report like the OOTB report.
To achieve this we need to create a custom bean object, a tablebuilder to show the report in tabular format and a jsp to show the table.
In your report there is Level, Number, Name, Version, State, Line Number, Find Number and Reference Designator.
So, in your customBean object there will be eight attribute now based on your requirement those attributes value can be int,String etc.To simplify here I use String for all eight attribute.

- **Step 1:**

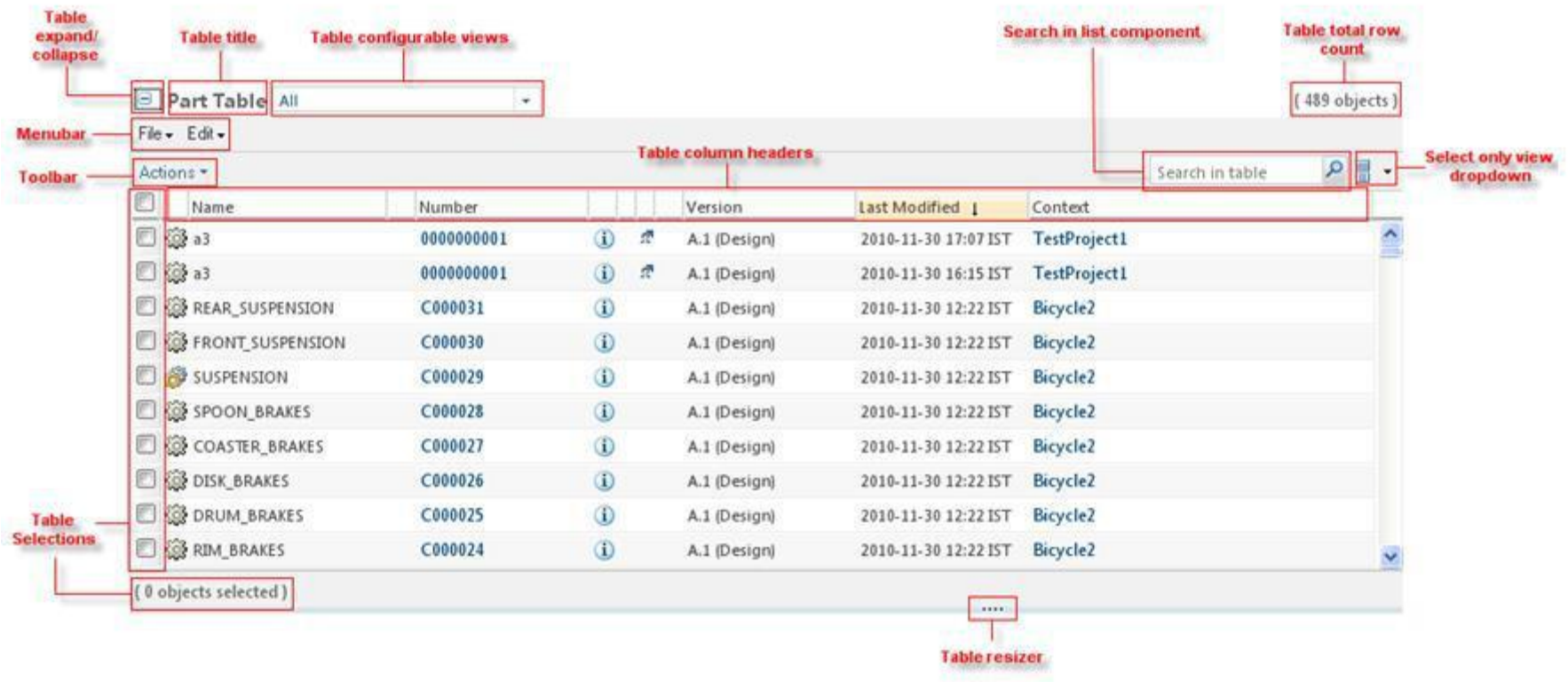
Create a customBean object named MyReportBean.

- **Step 2:**

Create the TableBuilder. (ext.tablebuilder.CustomTableBuilder)

Below is a picture about different component of a table.

Depending upon the configuration, Table can be displayed in different formats. Following is a general layout of a Table:



To know more about the table builder refer to windchill help section or windchill customization guide.

http://<Host name>/Windchill-WHC/index.jspx?id=WCCG_UICust_PresentInfoUI_ConstructRender&action=show

❖ Screenshot :

Custom Report Table (44 objects)

Actions — Search in table

Level	Name	Number	Version	State	Line Number	Find Number	Reference Designator
0	GOLF_CART	GC000001	A.6(Design)	In Work			
1	LOWER_SUPPORT	GC000019	A.3(Design)	In Work	20		D20022_1
2	SUPPORT_CAP	GC000021	A.1(Design)	In Work	20		D20024_1
2	LOWER_SUPP_BAR	GC000020	A.1(Design)	In Work	10		D20023_1
2	LOWER_BAG_HOLDER	GC000022	A.1(Design)	In Work	30		D20025_1
2	PIN	GC000023	A.1(Design)	In Work	40		D20026_1
1	LT_ARM	GC000039	A.1(Design)	In Work	60		D20042_1
1	RT_ARM	GC000038	A.1(Design)	In Work	50		D20041_1
1	WHEELS_ASSEM	GC000031	A.3(Design)	In Work	40		D20034_1
2	WHEEL_AXLE	GC000032	A.1(Design)	In Work	10		D20035_1-D20035_2
2	BEARING_AXLE	GC000033	A.1(Design)	In Work	20		D20036_1-D20036_2
2	WHEEL_HUB	GC000035	A.1(Design)	In Work	40		D20038_1-D20038_2
2	TIRE	GC000036	A.1(Design)	In Work	50		D20039_1-D20039_2
2	HUB_CAP	GC000037	A.1(Design)	In Work	60		D20040_1-D20040_2
2	AXLE_SLEEVE	GC000034	A.1(Design)	In Work	30		D20037_1-D20037_2
1	UPPER_SUPPORT	GC000024	A.3(Design)	In Work	30		D20027_1
2	BOTTOM_SLIDER_CAP	GC000026	A.1(Design)	In Work	20		D20029_1
2	UPPER_SLIDER	GC000027	A.1(Design)	In Work	30		D20030_1
2	HANDLE	GC000029	A.1(Design)	In Work	50		D20032_1
2	UPPER_BAG_HOLDER	GC000028	A.1(Design)	In Work	40		D20031_1
2	BOTTOM_SLIDER	GC000025	A.1(Design)	In Work	10		D20028_1
2	CARD_HOLDER	GC000030	A.1(Design)	In Work	60		D20033_1
1	LEG	GC000002	A.3(Design)	In Work	10	10T	D20001_1
2	BOLT_1_8	GC000017	A.1(Design)	In Work	50		D20020_1-D20020_4
2	ACTUATOR_LOCK	GC000016	A.1(Design)	In Work	40		D20019_1

At first your table will look like this.
Now it's time for some cosmetic changes. To do so we will use DataUtility.

❖ For State Column :

Create LifeCycleDataUtility.java (For detailed instruction read the source code)

❖ For Name :

For name create "CustomNameDataUttility.java"

❖ ScreenShot:

After applying the Data Utility, Your Report will look like this.

Custom Report Table (44 objects)

Actions — Search in table

Level	Name	Number	Version	State	Line Number	Find Number	Reference Designator
0	GOLF_CART	GC000001	A.6(Design)				
1	.... LOWER_SUPPORT	GC000019	A.3(Design)		20		D20022_1
2	..... SUPPORT_CAP	GC000021	A.1(Design)		20		D20024_1
2	..... LOWER_SUPP_BAR	GC000020	A.1(Design)		10		D20023_1
2	..... LOWER_BAG_HOLDER	GC000022	A.1(Design)		30		D20025_1
2	..... PIN	GC000023	A.1(Design)		40		D20026_1
1	.... LT_ARM	GC000039	A.1(Design)		60		D20042_1
1	.... RT_ARM	GC000038	A.1(Design)		50		D20041_1
1	.... WHEELS_ASSEM	GC000031	A.3(Design)		40		D20034_1
2	..... WHEEL_AXLE	GC000032	A.1(Design)		10		D20035_1-D20035_2
2	..... BEARING_AXLE	GC000033	A.1(Design)		20		D20036_1-D20036_2
2	..... WHEEL_HUB	GC000035	A.1(Design)		40		D20038_1-D20038_2
2	..... TIRE	GC000036	A.1(Design)		50		D20039_1-D20039_2
2	..... HUB_CAP	GC000037	A.1(Design)		60		D20040_1-D20040_2
2	..... AXLE_SLEEVE	GC000034	A.1(Design)		30		D20037_1-D20037_2
1	.... UPPER_SUPPORT	GC000024	A.3(Design)		30		D20027_1
2	..... BOTTOM_SLIDER_CAP	GC000026	A.1(Design)		20		D20029_1
2	..... UPPER_SLIDER	GC000027	A.1(Design)		30		D20030_1
2	..... HANDLE	GC000029	A.1(Design)		50		D20032_1
2	..... UPPER_BAG_HOLDER	GC000028	A.1(Design)		40		D20031_1
2	..... BOTTOM_SLIDER	GC000025	A.1(Design)		10		D20028_1
2	..... CARD_HOLDER	GC000030	A.1(Design)		60		D20033_1
1	.... LEG	GC000002	A.3(Design)		10	10T	D20001_1
2	..... BOLT_1_8	GC000017	A.1(Design)		50		D20020_1-D20020_4
2	..... ACTUATOR_LOCK	GC000016	A.1(Design)		40		D20019_1

If you want you can export this report into various format like .xls,.pdf etc using the Action provided in the table.

JSP :

```
<%@include file = "/netmarkets/jsp/components/beginWizard.jspf" %>
<%@taglib prefix="jcaMvc" uri="http://www.ptc.com/windchill/taglib/jcaMvc"%>

<jcaMvc:tableContainer compId="ext.tablebuilder.CustomTableBuilder" height="1600" />

<%@include file = "/netmarkets/jsp/util/end.jspf"%>
```

This JSP is linked with the custom action.
After clicking on the custom action the table builder will invoke from this JSP

5.4 Requirement IV : Reusing component config builder

Create two action Document List and Part List upon clicking which corresponding table will pop-up in a jsp to display all the corresponding object (WTPart or WTDocument) in that particular product.

The main objective of this example is to create two table builder with common component config builder but separate data builder.

- **Step 1 :**

  Add below lines into **folderbrowser_toolbar_actions** model

  <action name = "documentlist" type="itc"/>
  <action name = "partlist" type="itc"/>

- **Step 2 :**

  Modify CustomActionResource.java for documentlist and partlist action.(i.e add tooltip,description , title etc)

- **Step 3 :**

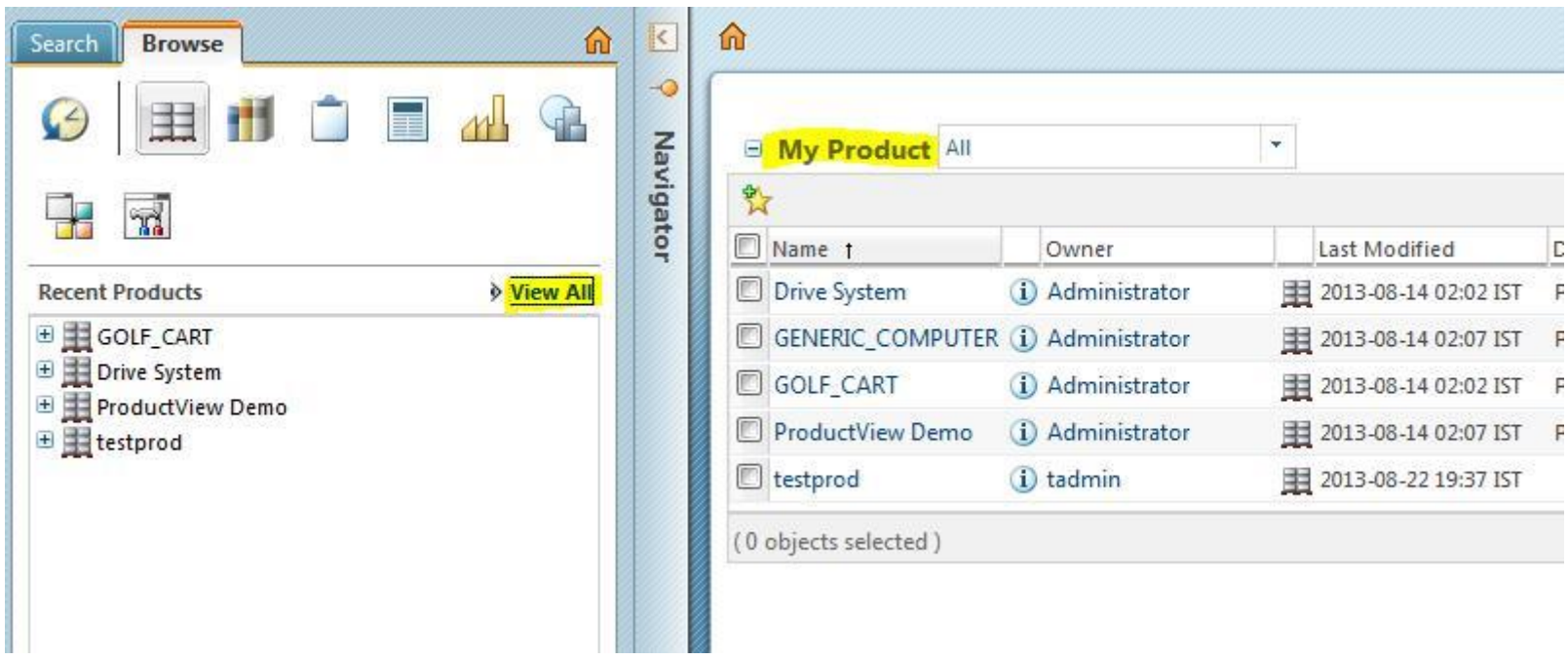
  Create class “CommonComponentConfigBuilder” , “ShowAllPartTableBuilder” and “ShowAllDocTableBuilder”.

- **Step 4 :**

  Restart method server.

5.5 Requirement V : Extending OOTB table builder

Need to change the Label of any OOTB table.
If you need to extend OOTB table builders then you have to use @OverrideComponentBuilder annotation. For an example I need to change the label of “Product” to “My Product”



Create class ext.tablebuilder.CustomProductListTableBuilder, Register it and Restart the method server.

- **Explanation:**

If you are overriding any OOTB builder you have to use @OverrideComponentBuilder. Please note this is not an optimum solution. Best way to achieve the same is to change the resource bundle.

5.6 Requirement VI : Advance feature of Table Builder

Now we will explore more advance feature of table builder.
For example purpose we will create one action in folder browser toolbar action.
Upon clicking on that action a jsp page will pop-up which will show all objects in that particular folder.
The table is view configurable.

❖ **Step 1 :**

Add below lines into **folderbrowser_toolbar_actions** model
<action name = "objectlist" type="itc"/>

❖ **Step 2 :**

Modify CustomActionResource.java for objectlist action.(i.e add tooltip,description , title etc)

❖ **Step 3 :**

Attach objectlist.jsp with your action.

```
<command url="/netmarkets/jsp/itc/objectlist.jsp" windowType="popup"/>
```

```
<%@include file = "/netmarkets/jsp/components/beginWizard.jspf" %>
<%@ taglib uri="http://www.ptc.com/windchill/taglib/mvc" prefix="mvc"%>
<jsp:include page="${mvc:getComponentURL('ext.test.builder.CustomFolderBrowserTableBuilder')}" />
<%@include file = "/netmarkets/jsp/util/end.jspf"%>
```

❖ **Step 4 :**

Create class "CustomFolderBrowserTableBuilder" and "MvcConfigurableTable"(private static).

❖ **Step 5:**

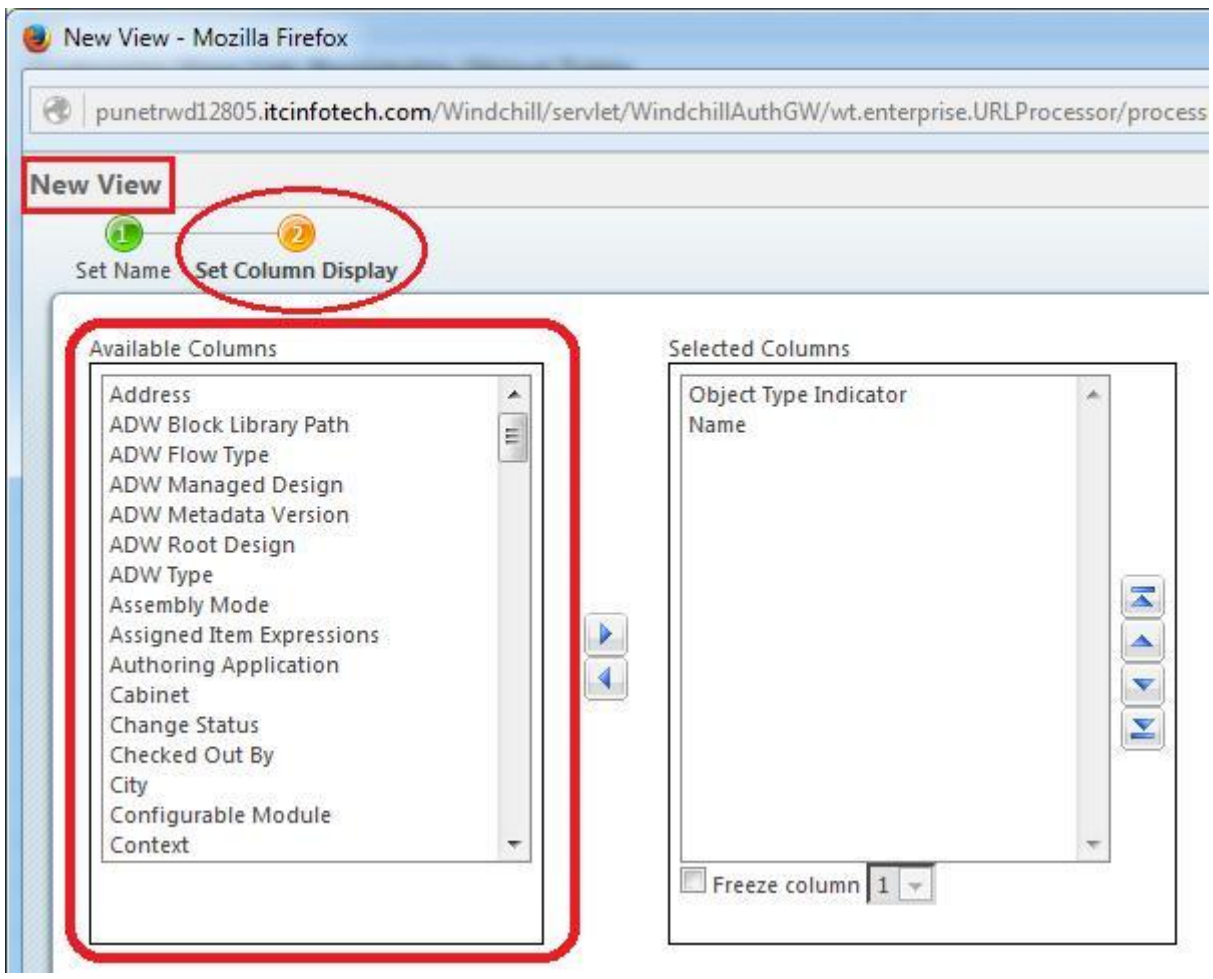
Start / Restart server.

❖ **Explanation:**

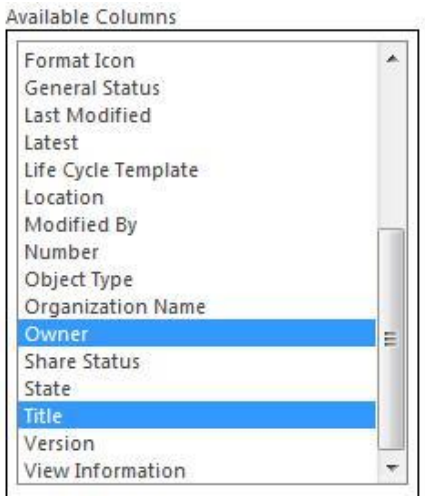
◇ In order to create view configurable table create a class which extends JCAConfigurableTable which is a child of AbstractConfigurableTable. After extending JCAConfigurableTable we have to implement following unimplemented method.

1. public Class[] getClassTypes()

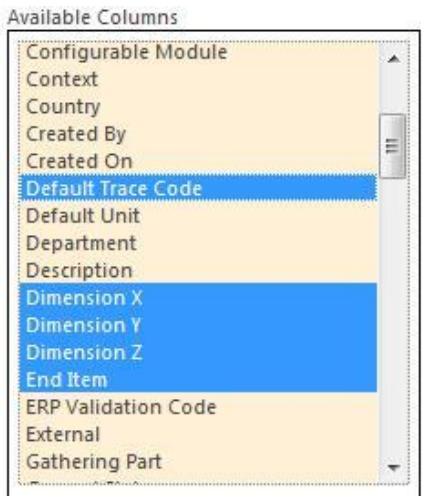
   Based on this value system will determine which columns need to be listed in "Available columns" section of "Set Column Display" wizard step in "New View" wizard.



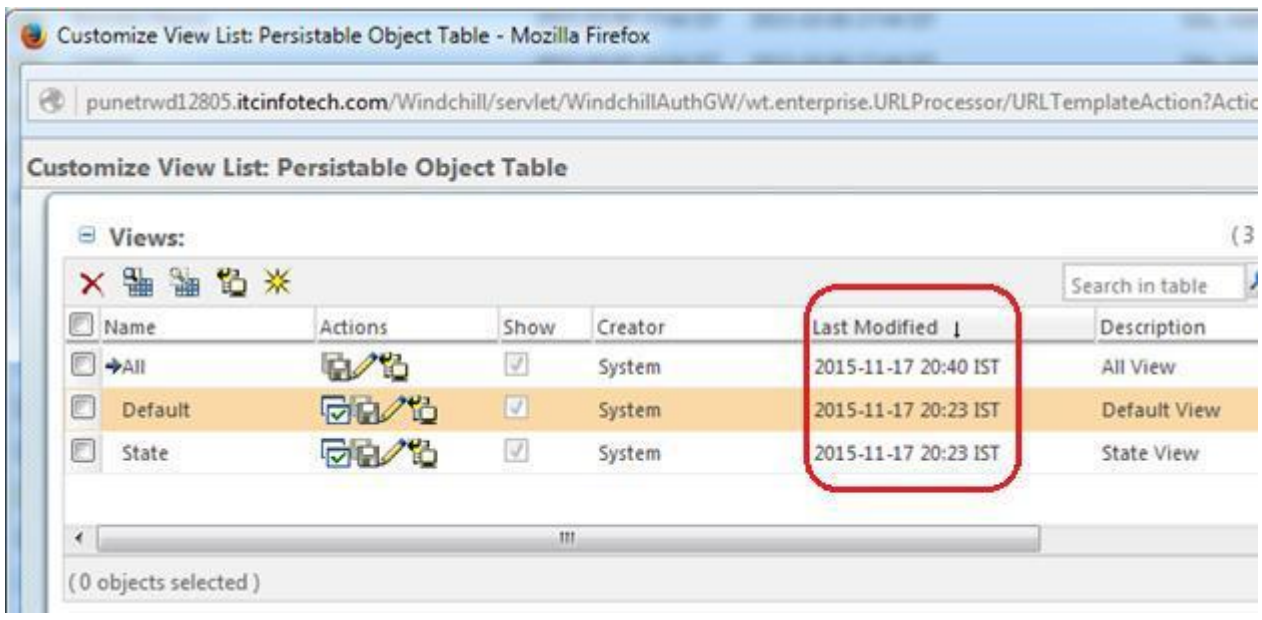
For WTDocument.class



For WTPart.class

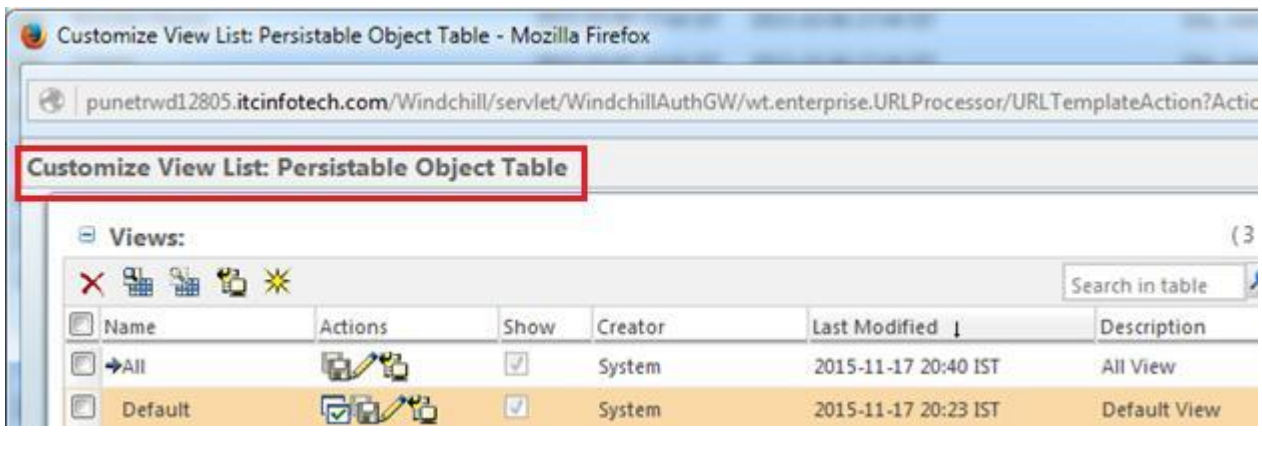


2. public String getDefaultSortColumn() This method shoud return a column id.
   Default sort column for "Views" table.



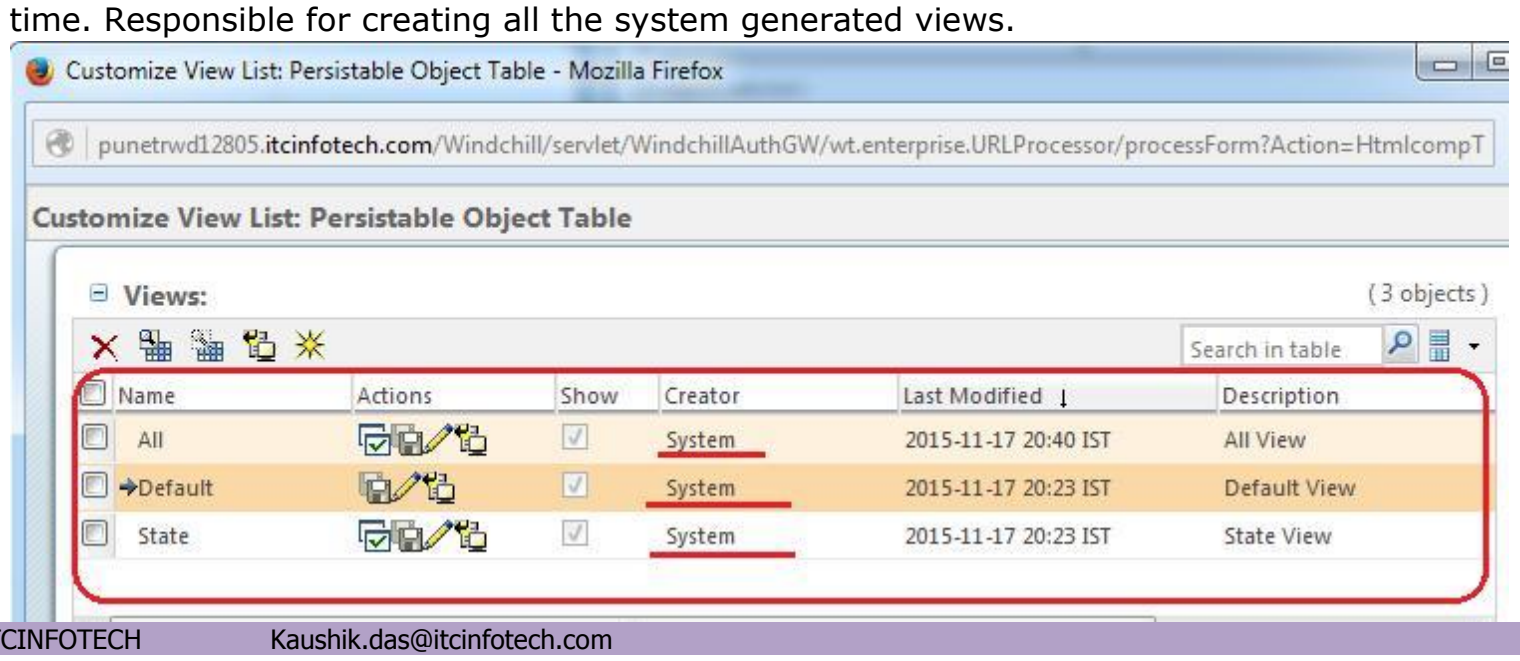
3. public String getLabel(Locale paramLocale)

   Label for the "Customize View List" wizard.



   Best Practice : Return a localized string.

4. public String getOOTBActiveViewName()
   Returns the name of the default view in localized string. In this example the name of the default view is default. You can override this value from UI.
5. public List getOOTBTableViews(String paramString, Locale paramLocale) Returns List of TableViewDescriptor. This method will get called only once when user is opening the table for the first time. Responsible for creating all the system generated views.

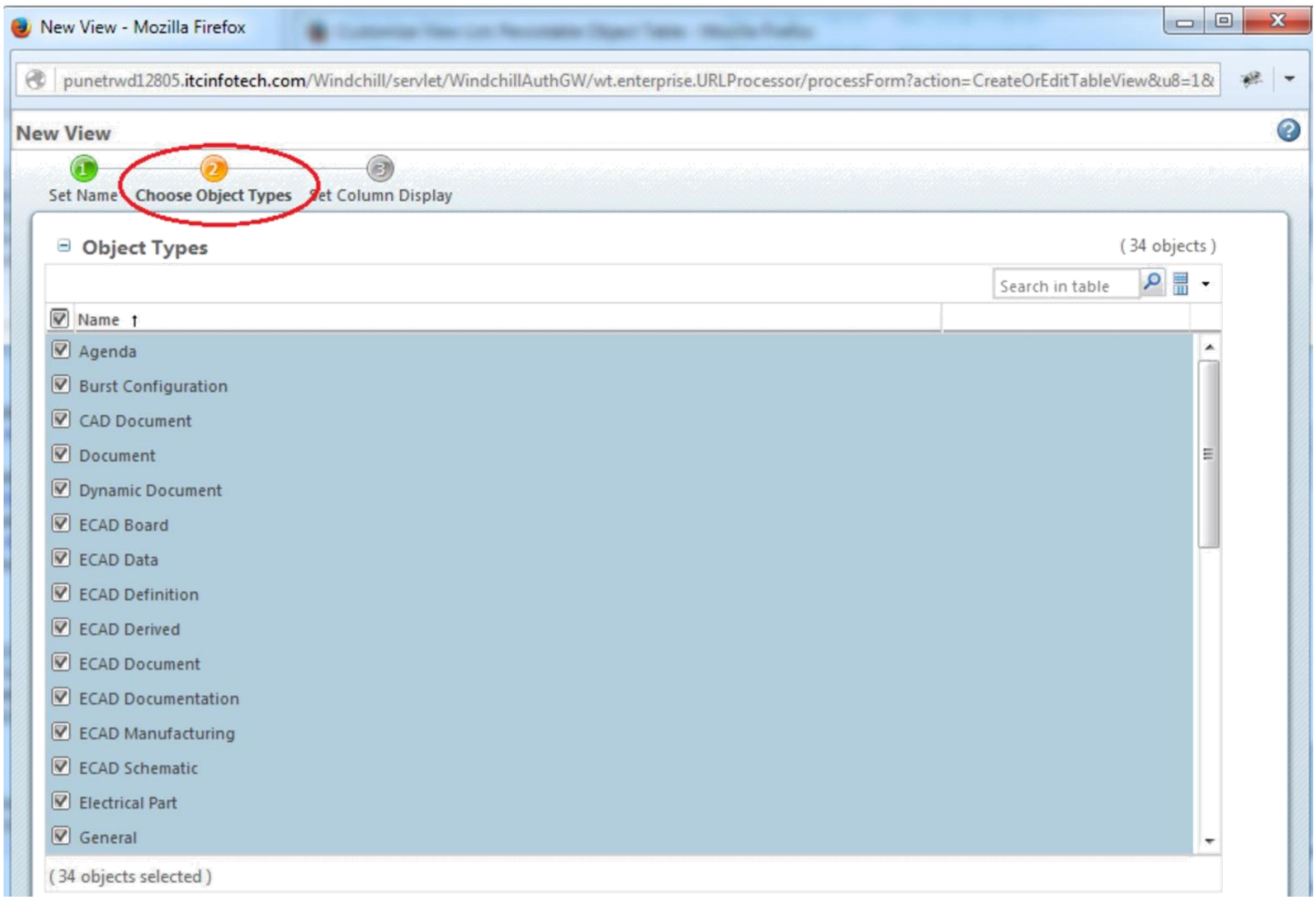


6. public List getSpecialTableColumnsAttrDefinition(Locale paramLocale)

Some other important methods are

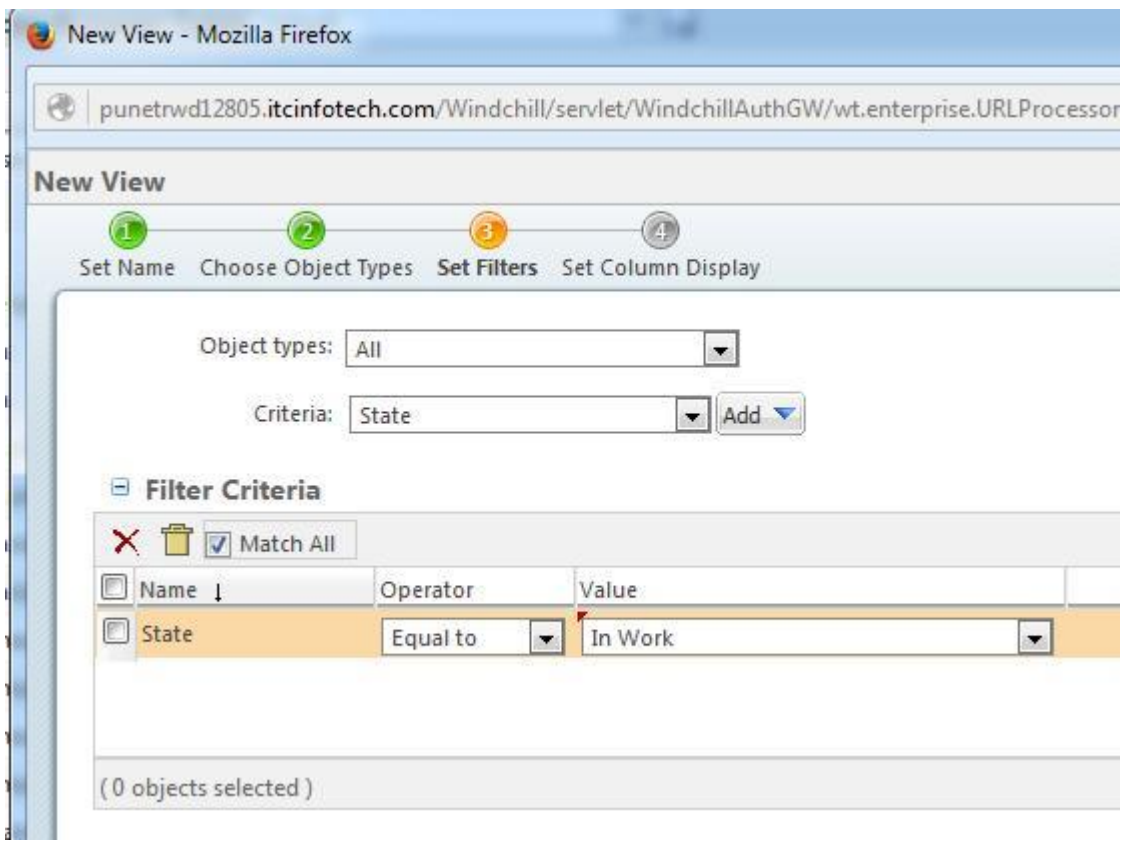
1. public boolean showChooseItemTypesStep()

At the time of creating new view, based on the return type of this method system will decide whether to show the wizard step "Choose Object Type".



2. public boolean showFilteringStep()

This method controls the visibility of the "Set Filters" wizard step.

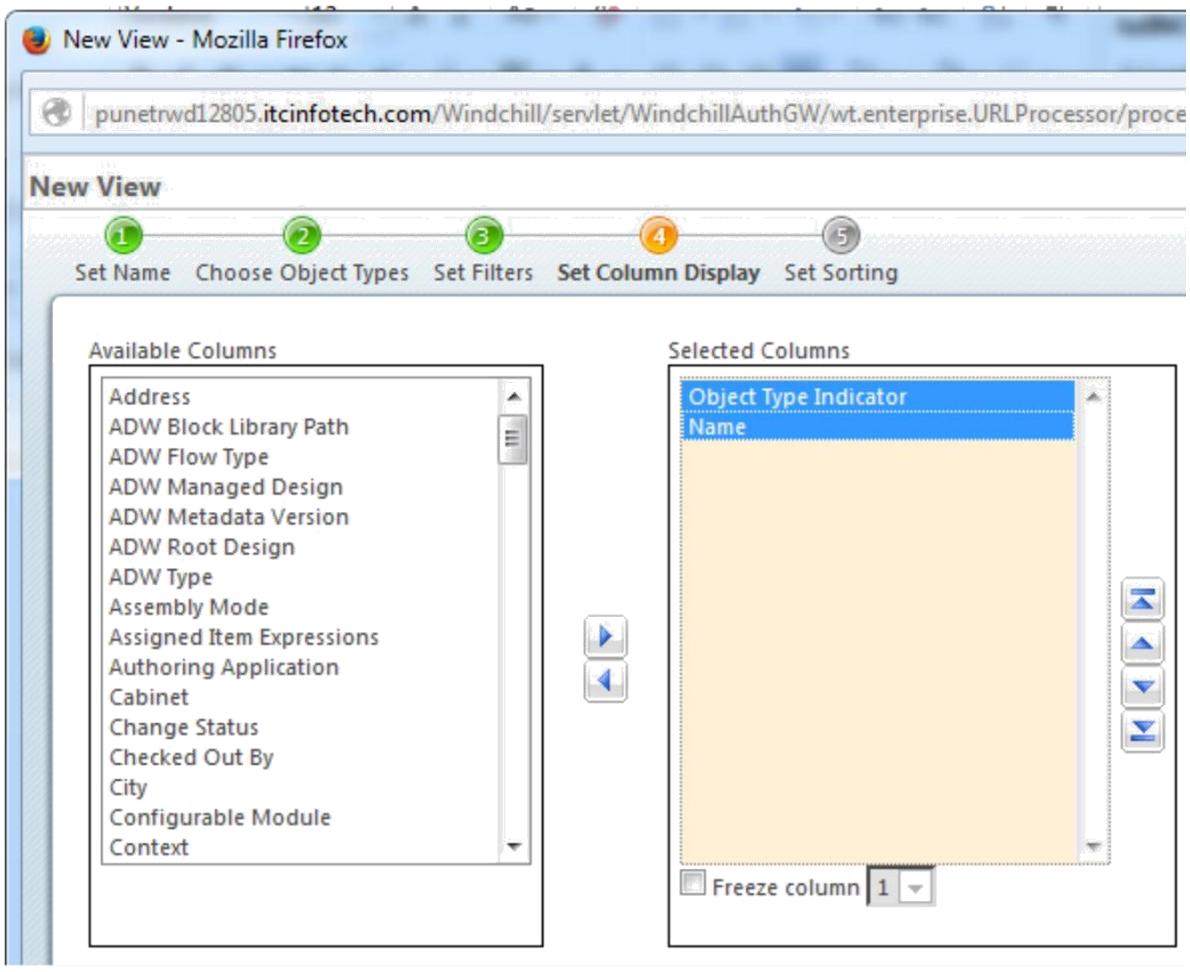


3. public boolean showSortingStep()
Similarly this method is for "Set Sorting" wizard step.

4. public boolean isColumnLocked(String paramString)

   Will get called for every column name available in "Available Column" list to determine which column(s) is/are mandatory in view.

   In our example "Obejct Type Indicator" and "Name" is locked.



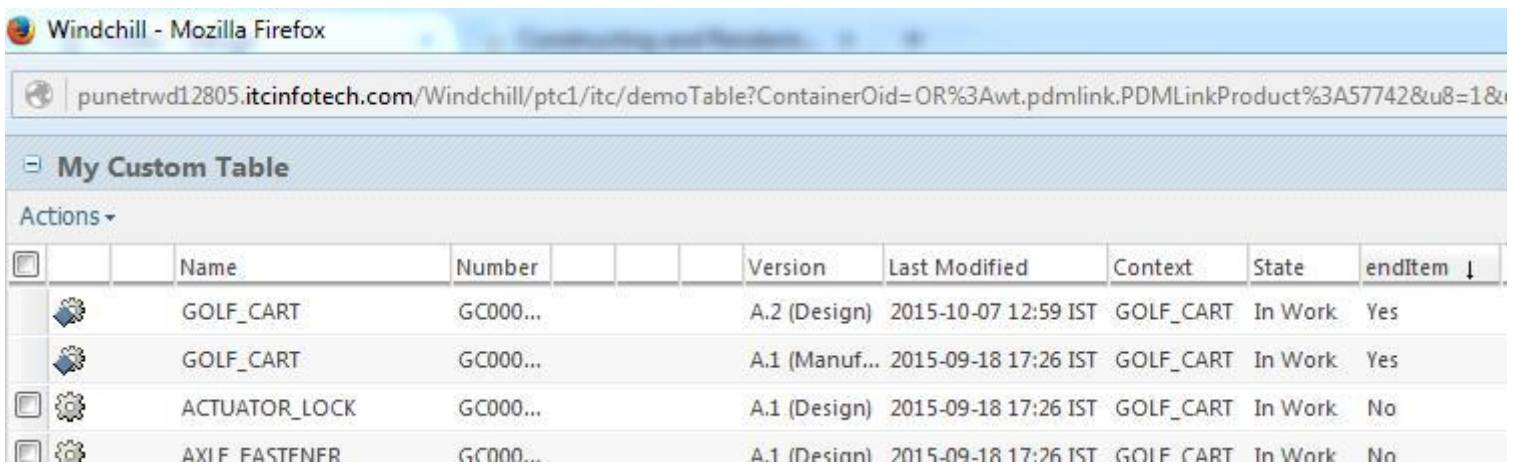
◊ Other Table Feature :-

1. Configuring Data Store Only Column and non-selectable column

   Setting a column as data store only would make the column value available in data store, but the column will not be displayed in the table.
   Non-selectable rows can be set depending upon a column that has a Boolean value.

```
ColumnConfig col = componentConfig.newColumnConfig (NON_SELECTABLE_COLUMN, false);
col.setNeed ("endItem");
/* Specify the attribute which will decide the row is selectable or not.*/

 col.setDataStoreOnly(true);
 tableConfig.addComponent (col);
 tableConfig.setNonSelectableColumn (col);
```

   In above example, all the end item parts are non-selectable.



If by default your attribute's value is not a Boolean value then you have to write a custom data utilty which will return Boolean variable.

Ex. **All the part which are in "Released" state are non-selectable**

```
ColumnConfig col = componentConfig.newColumnConfig
(NON_SELECTABLE_COLUMN, false); col.setNeed ("state");

 /*Specify the attribute which will decide the row is

  selectable or not.*/

col.setDataStoreOnly(true);
```

col.setDataUtilityId("customstatestrikethroughcolumn");

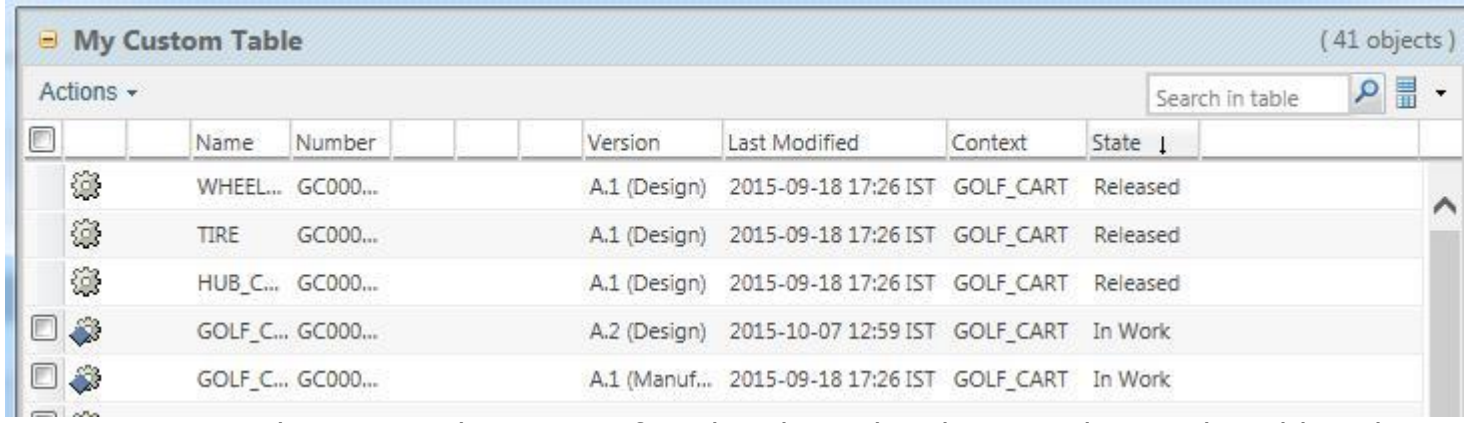
tableConfig.addComponent(col);

tableConfig.setNonSelectableColumn(col);

Register your data utility.

```
<Service context="default"
        name="com.ptc.core.components.descriptor.DataUtility"
        targetFile="codebase/com/ptc/windchill/enterprise/enterprise.dat
        aUtilities.properties">

        <Option cardinality="duplicate" requestor="java.lang.Object"
                selector="customstatestrikethroughcolumn"
                serviceClass="ext.datautility.CustomStateStrikeThroughColumn"/>
</Service>
```

Your data utility should return a Boolean value.



You can use similar approach in case of Strike Through column and Pre selectable column.

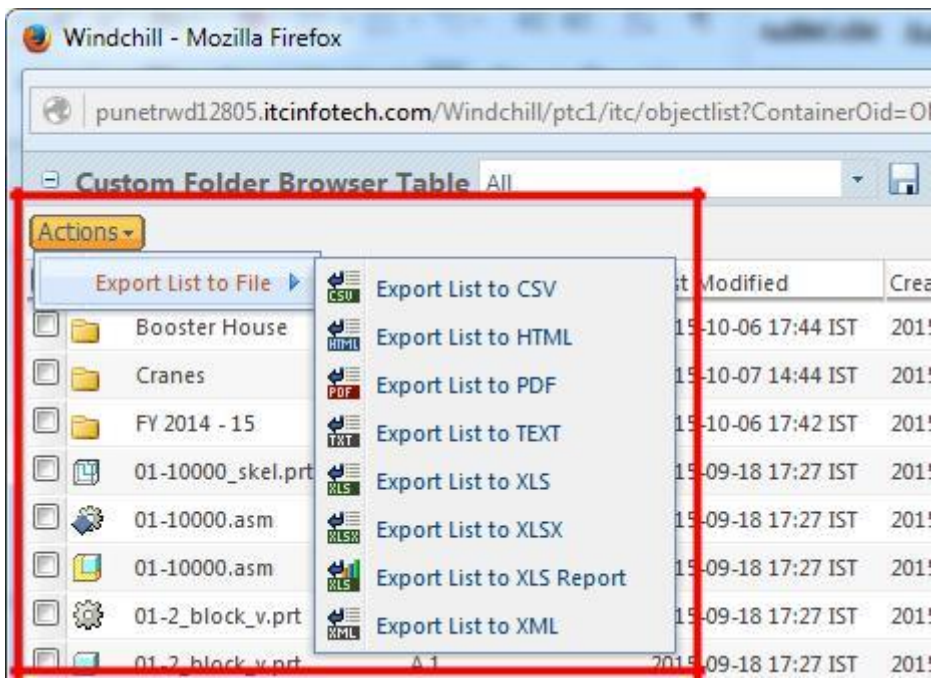
2. Adding a Toolbar

   In order to add a toolbar, the key attribute must be set to "actionModel" and the value attribute must be set to the name of the action model that contains the toolbar actions.

   Ex.

   JcaTableConfig tableConfig = (JcaTableConfig) componentConfig.newTableConfig();

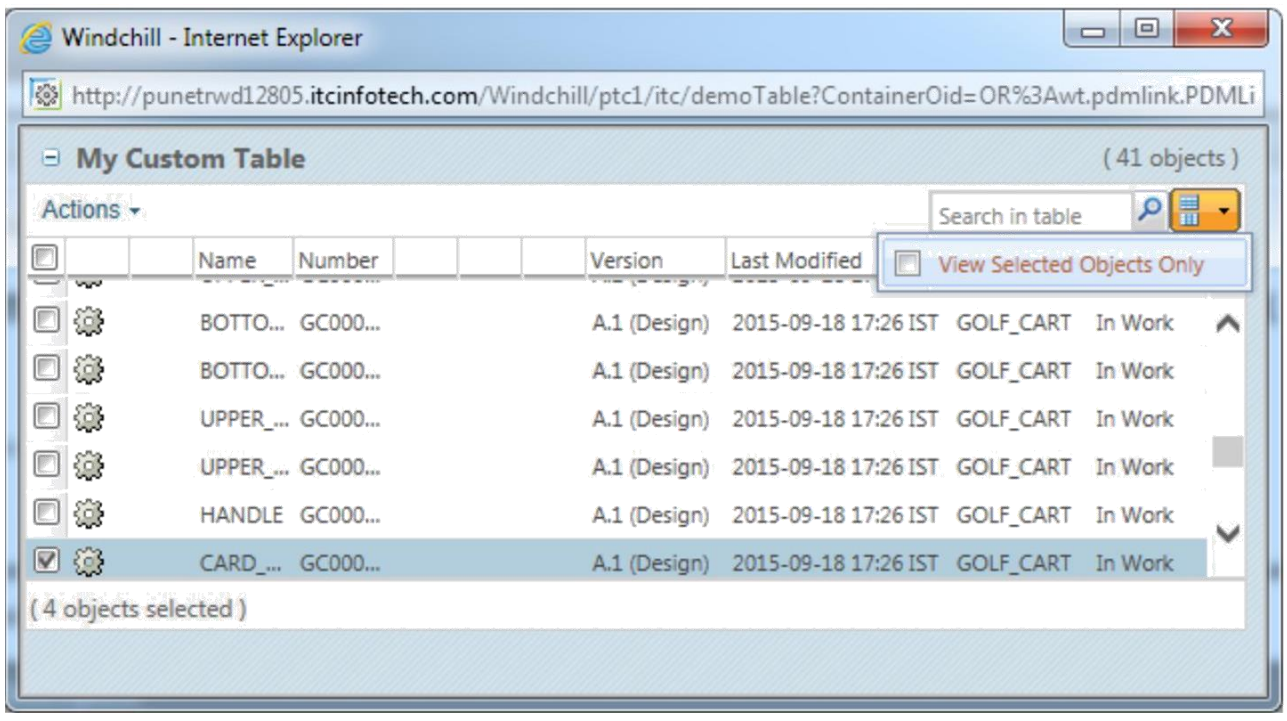
   tableConfig.setActionModel("part_report_toolbar_actions");

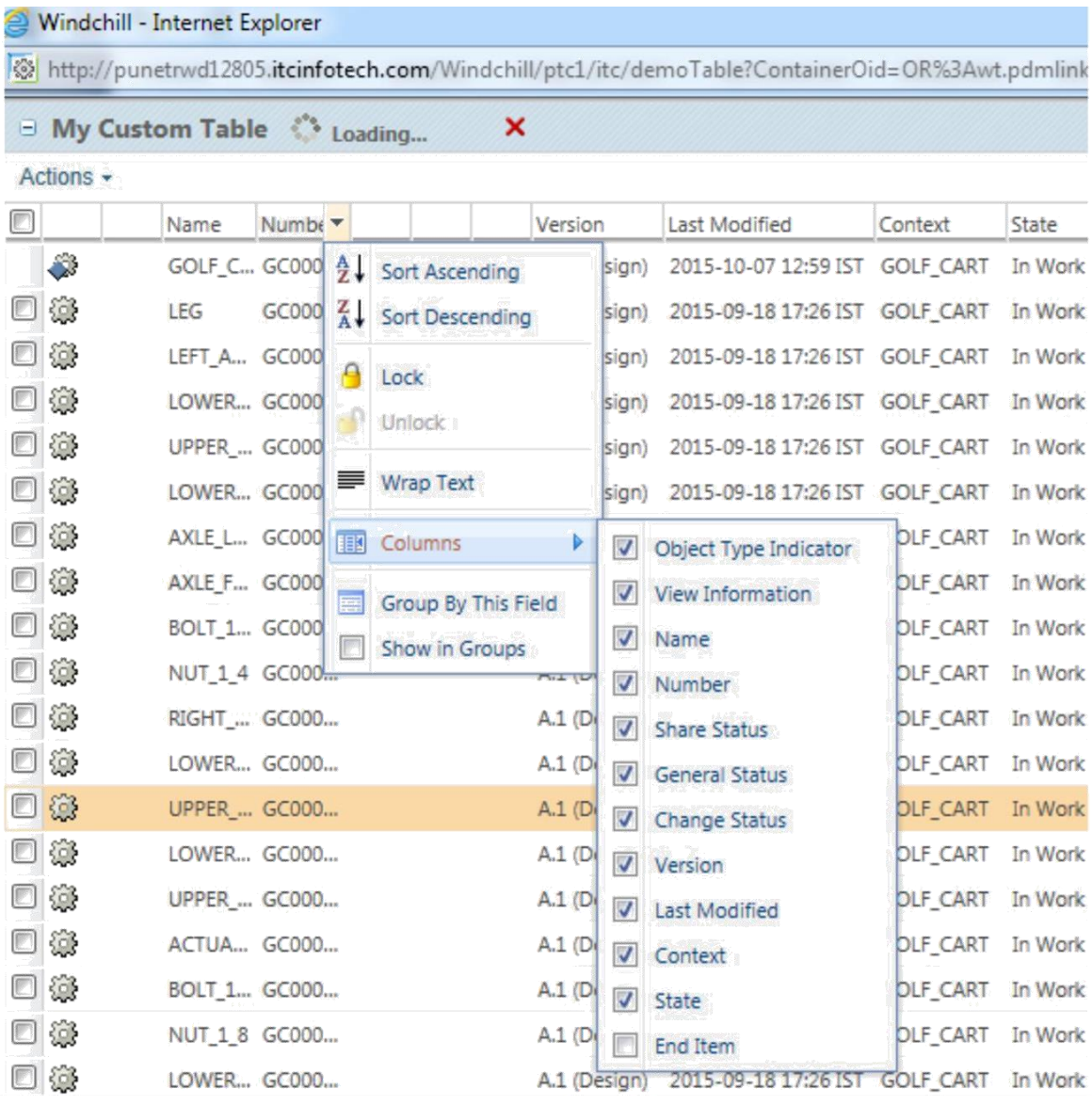


"part_report_toolbar_actions" is one already defined OOTB action model.
If you want you can create your own action model in custom-actionModel.xml and include that as toolbar.

3. Configuring Column as a Hidden Column
   A column can be set as hidden so that it is not displayed on the table. However, hidden column is available in the column list and thereby can be made un-hidden.
   For Example below code will sent end item column as hidden.

```
final ColumnConfig endItem = componentConfig
                .newColumnConfig("endItem", true);
endItem.setLabel("End Item");
endItem.setInfoPageLink(true);
endItem.setHidden(true);
tableConfig.addComponent(endItem);
```



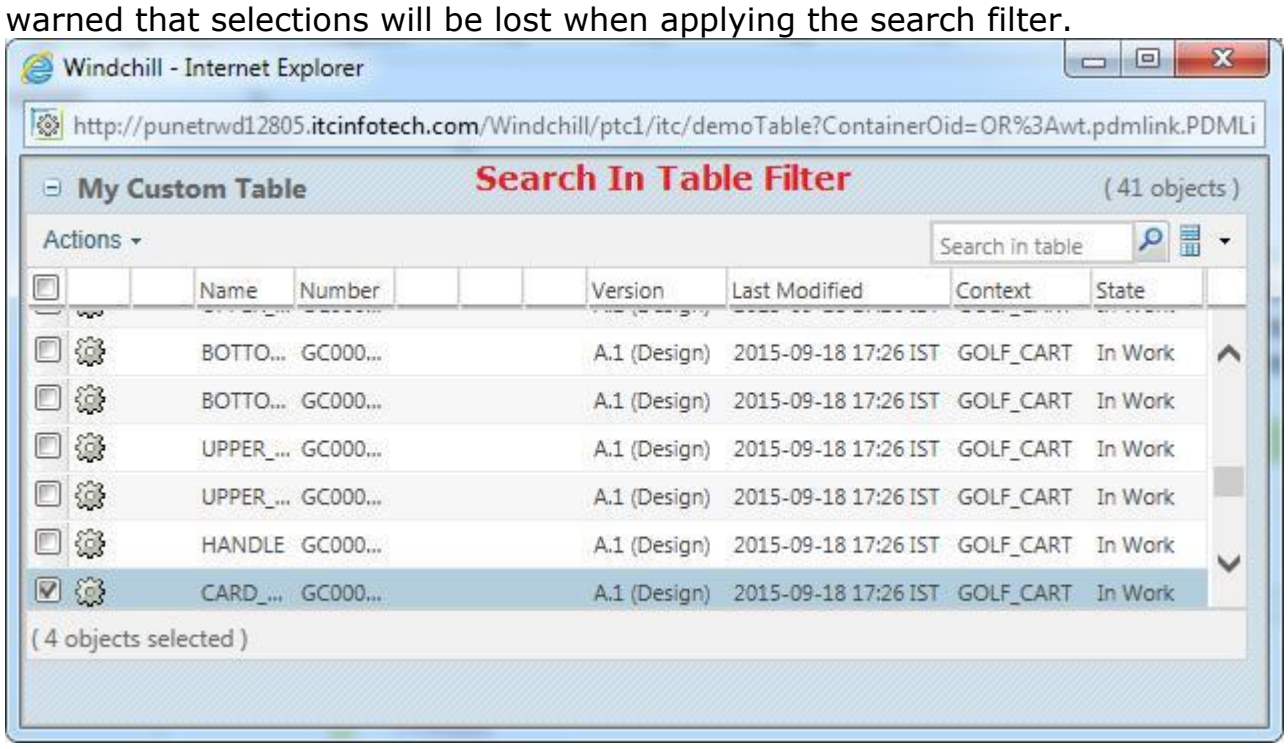


4. Configuring Search in List
   By default the Search In List feature is enabled for the table. A user can enter text and hit Enter or Search In List icon to execute the search. After applying the search in list filter, the table will be displayed with the records matching with the text entered. Only the values in the visible columns will be searched. The Search in List filter will be sticky across component/page refreshes.

   The Search in List table filter and View Selected Objects Only table filter interact as follows:

   - Selected and disabled rows stay checked/disabled when the Search in Table filter is applied. This is only allowed if the table in question is complete (the data source is no longer running) and it only has one

page of data. For tables that are still populating or that have more than one page of data, the user is warned that selections will be lost when applying the search filter.



- Only one filter is allowed to be applied to the table at a time. The user can apply either the Search in Table filter, OR the View Selected Objects Only filter. But not both at once. When the user chooses one filter when the other is on, they are prompted to that the original filter will be cleared as well.

Disabling Search In List feature :-

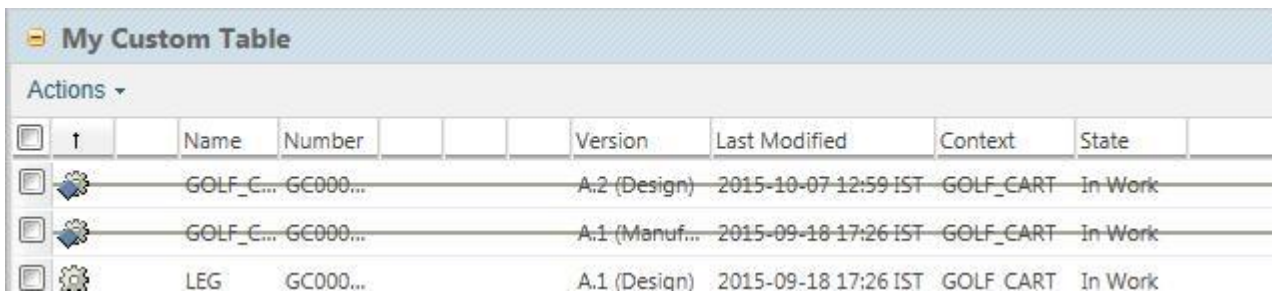
Search in list can be disabled from a table by setting "false" in setFindInTableEnabled.

```
TableConfig tableconfig = factory.newTableConfig('myTable');
tableConfig.setFindInTableEnabled(false); or
tableConfig.setFindInTableMode(FindInTableMode.DISABLED);
```

The search in list modes available are:

- FindInTableMode.CLIENT_AND_SERVER(default mode): Perform the search in table on the client or server data.
- FindInTableMode.CLIENT_ONLY: Perform search in table on the client data only.
- FindInTableMode.DISABLED: Search in list panel is not displayed.

5. Configuring strikethrough on a hidden column
   Add dataStoreOnly column with id "strikeThroughRow". Provide need attribute for boolean column.

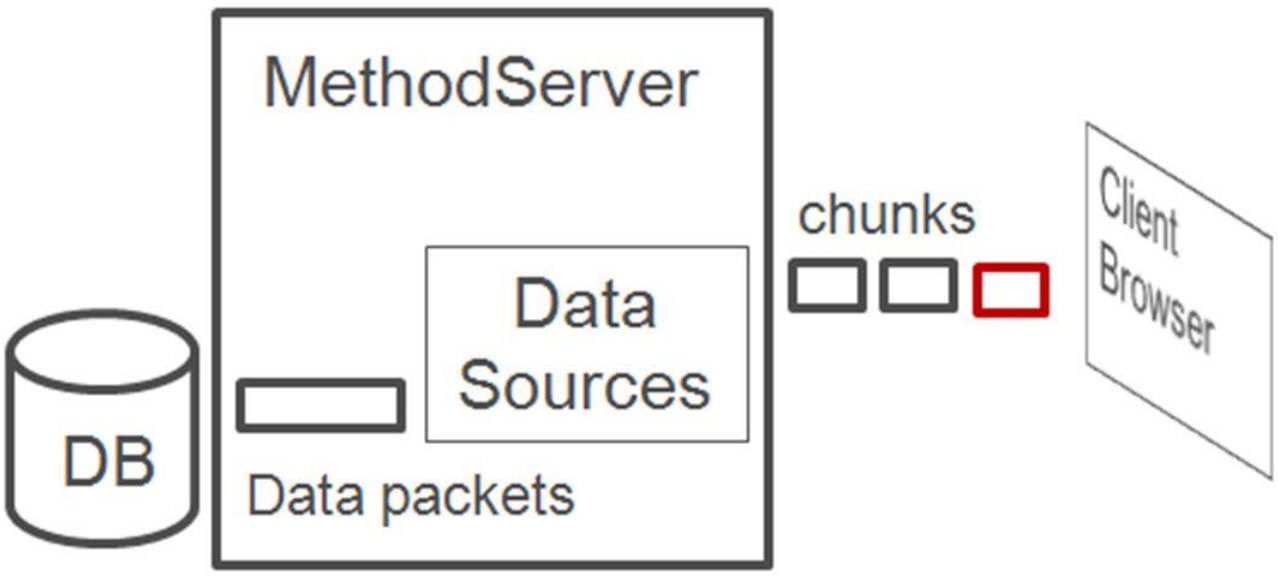



```
/* To make strke through row */
ColumnConfig col = componentConfig.newColumnConfig
(STRIKETHROUGH_COLUMN, false); col.setNeed ("endItem");
 /*Specify the attribute which will decide the row is
   selectable or not.*/
col.setDataStoreOnly(true);
tableConfig.addComponent (col);
tableConfig.setStrikeThroughColumn(col) ;
```

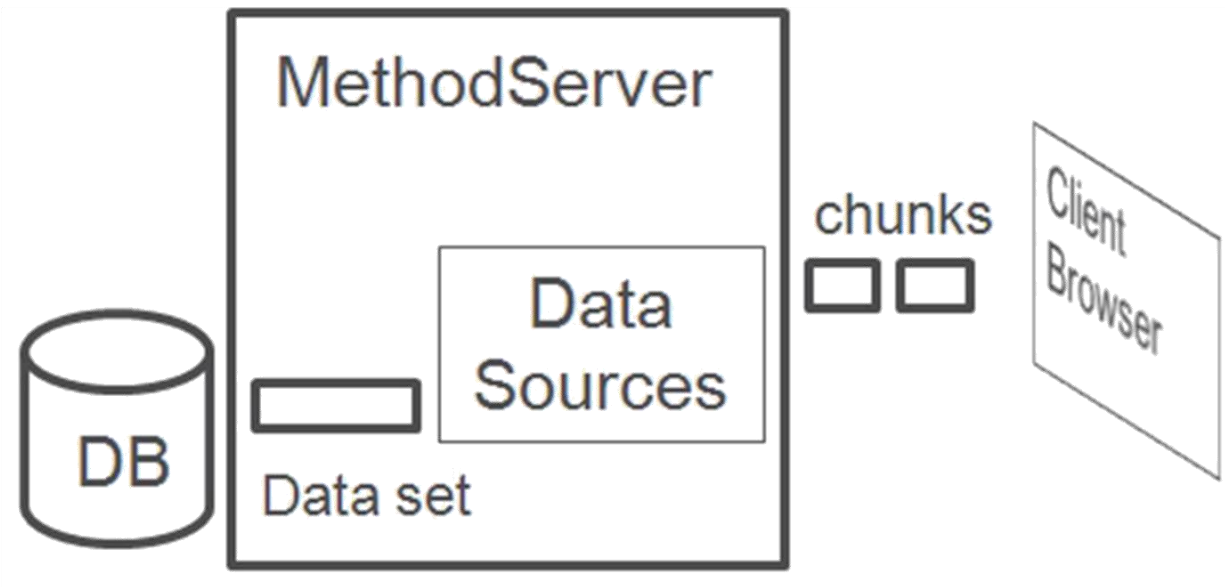
❖ **Note :**

DataSources Synchronous and Asynchronous

Synchronous data sources: Uses synchronous data acquisition (all data comes from the database in one chunk) and streams chunks to the client. No blank chunk is sent, so the component is not rendered until the first data chunk is received. Currently we don't support this for Tree components.



Asynchronous data sources: Uses synchronous data acquisition (all data comes from the database in one chunk) and streams chunks to the client. The first chunk sent to the client contains no table data but enables the table to be rendered before a full data chunk is available. This gives the user control of the table and some of its actions sooner. The data chunks then stream in behind the blank chunk as they are processed.



5.7 Requirement VII : Info*Engine as data source

Create an action in "folderbrowser_toolbar_actions" which will show all the part in a particular container in tabular format.

Here, we will use info engine task to fetch the data.

- Step 1:

Create action and jsp.

custom-actions.xml

```
<action name="customtableie" type="itc" >
        <command url="/netmarkets/jsp/itc/demoTableie.jsp" windowType="popup"/>
</action>
```

demoTableie.jsp

```
<%@include file = "/netmarkets/jsp/components/beginWizard.jspf" %>
<%@ taglib uri="http://www.ptc.com/windchill/taglib/mvc" prefix="mvc"%>

<jsp:include page="${mvc:getComponentURL('part.doc.ie')}" />

<%@include file = "/netmarkets/jsp/util/end.jspf"%>
```

- **Step 2:**

Create class "ext.test.builder.PartDocumentBuilderWithIE" for table.
For data utility, create "ext.datautility.CustomViewNonSelectableColumn"

❖ **Step 3:**

Create IE task.

```
<?xml version="1.0" standalone="yes"?>
<%@page language="java" access="internal|http"%>

<%@taglib uri="http://www.ptc.com/infoengine/taglib/core" prefix="ie"%>

<ie:webject name="Get-Properties" type="MGT">
   <ie:param name="ATTRIBUTE" data="wt.federation.ie.VMName"/>
   <ie:param name="GROUP_OUT" data="properties"/>
 </ie:webject>

<ie:webject name="Query-Objects" type="OBJ">

      <ie:param name="INSTANCE"
      data="<%=com.infoengine.au.NamingService.getVMName()%>"/>
      <ie:param name="CONTAINER_REF" data="${@FORM[]container[0]}"/>
      <ie:param name="WHERE" data="()"/>

      <ie:param name="ATTRIBUTE"
      data="name~number~state.state~thePersistInfo.modifyStamp" delim="~"
      valueSeparator="~"/> <ie:param name="ATTRIBUTE"
      data="view.id~name|iterationInfo.creator~name" delim="|" valueSeparator="|"/> <ie:param
      name="TYPE" data="${@FORM[]search_type[0]}" default="wt.part.WTPart"/>

      <ie:param name="GROUP_OUT" data="output"/>

</ie:webject>

<ie:webject name="Change-Group" type="GRP">
 <ie:param name="GROUP_IN" data="output"/>
 <ie:param name="GROUP_OUT" data="output"/>
 <ie:param name="RENAME" data="'view.id~name'='ViewName'"/>

 <ie:param name="RENAME"
data="'iterationInfo.creator~name'='creatorName'"/>
</ie:webject>

<%
System.out.println("Query Executed");
%>
```

❖ **Step 4:-**

If you are using data utility then register it.

```
<Service context="default" name="com.ptc.core.components.descriptor.DataUtility"
      targetFile="codebase/com/ptc/windchill/enterprise/enterprise.dataUtilities.properties">

      <Option cardinality="duplicate" requestor="java.lang.Object"
            selector="customstatestrikethroughcolumn"
            serviceClass="ext.datautility.CustomStateStrikeThroughColumn"/>

      <Option cardinality="duplicate" requestor="java.lang.Object"
            selector="customviewnonselectablecolumn"
            serviceClass="ext.datautility.CustomViewNonSelectableColumn"/>
</Service>
```

❖ Screenshot :

Custom Table (I*E)

Actions

	Number	Name	Create...	Version	Last Modified	View Name ↓	State
	GC000040	WHEEL...	Admini...	A.1 (Manuf...	2015-09-18 17:26:49 IST	Manufacturing	In Work
	GC000001	GOLF_...	Admini...	A.1 (Manuf...	2015-09-18 17:26:49 IST	Manufacturing	In Work
	GC000001	GOLF_...	Admini...	A.2 (Design)	2015-10-07 12:59:44 IST	Design	In Work
	00000000...	part2	Admini...	A.1 (Design)	2015-11-23 19:08:54 IST	Design	In Work
	GC000002	LEG	Admini...	A.1 (Design)	2015-09-18 17:26:42 IST	Design	In Work
	GC000003	LEFT_A...	Admini...	A.1 (Design)	2015-09-18 17:26:43 IST	Design	In Work
	GC000004	LOWE...	Admini...	A.1 (Design)	2015-09-18 17:26:43 IST	Design	In Work
	GC000005	UPPER...	Admini...	A.1 (Design)	2015-09-18 17:26:43 IST	Design	In Work
	GC000006	LOWE...	Admini...	A.1 (Design)	2015-09-18 17:26:43 IST	Design	In Work
	GC000007	AXLE_L...	Admini...	A.1 (Design)	2015-09-18 17:26:43 IST	Design	In Work
	GC000008	AXLE_F...	Admini...	A.1 (Design)	2015-09-18 17:26:43 IST	Design	In Work
	GC000009	BOLT_1...	Admini...	A.1 (Design)	2015-09-18 17:26:43 IST	Design	In Work
	GC000010	NUT_1_4	Admini...	A.1 (Design)	2015-09-18 17:26:44 IST	Design	In Work
	GC000011	RIGHT_...	Admini...	A.1 (Design)	2015-09-18 17:26:44 IST	Design	In Work

6. VALIDATOR

❖ Overview

It is probably helpful to begin with a definition of the term *validation*. For the purposes of this discussion, the term *validation* refers to activities performed to determine what a user can see or do. For example:

- ☐ Should we display the "Create Part" action?
- ☐ Should we allow the checkout of this object?
- ☐ Is everything the user entered in this create wizard OK?

For the purposes of our discussion, *validation* can be broken down into three broad categories:

❖ Pre-Validation

- o Attempts to answer the questions: "Should something appear to the user in the UI? And, if so, should it be editable/selectable?"
- o For example, Should we display and enable the "Create Part" action for user A in container B?
- o Pre-Validation can be performed for actions or other UI components (status glyphs, attributes, tables, etc.)

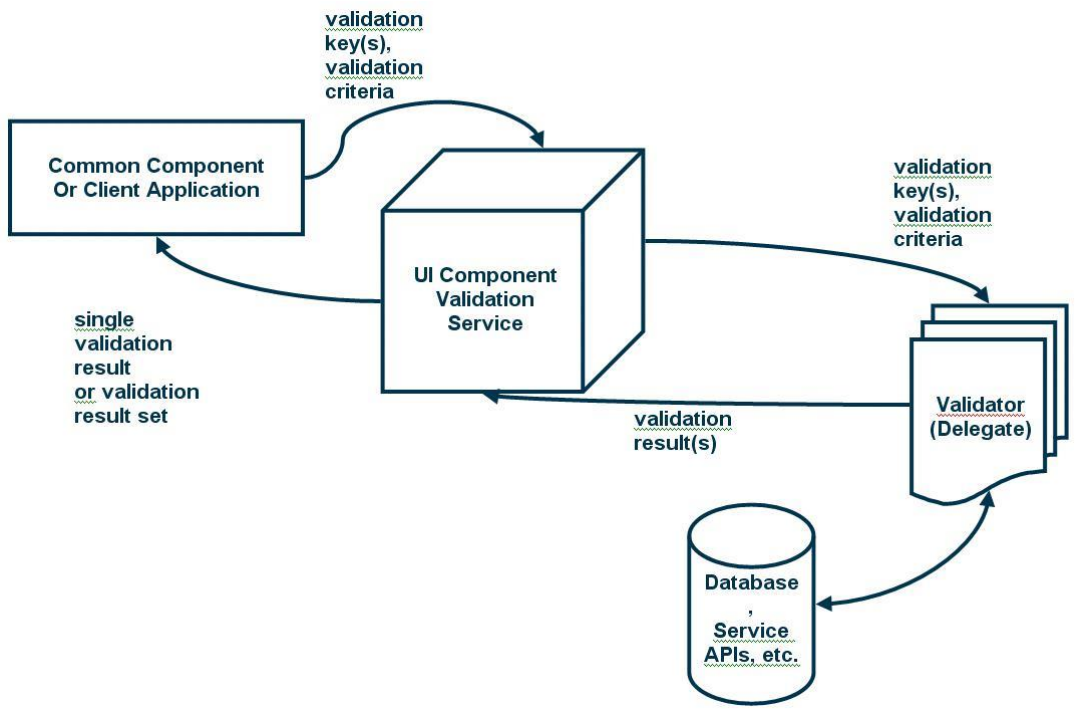
❖ Post-Select Validation

Attempts to answer the question: "Should the operation that was just selected in the UI be allowed to proceed?"
For example, Can we allow the checkout of parts A, B, and C?

❖ Post-Submit Validation

- o Attempts to answer the question: "is the data the user just entered valid?"
- o For example, When the user clicks 'Next' in the "Create Part" wizard, are we going to let them go to the next step, or do they need to modify some data (e.g., name, number) in the current step?

The UI Component (Action) Validation Service exposes one or more APIs for each of the types of validation listed above.
From a high level, a common component or some other client invokes a validation API on the Validation Service, passing one or more validation keys (which can be thought of as an action name, like create, for instance) and a UIValidationCriteria bean, which contains data known by the client that is required to perform validation. The Validation Service uses the validation key(s) to perform a lookup and identify the Validator class(es) that should be called to carry out the validation activity. The service then passes the key and UIValidationCriteria on to the identified Validator(s) and awaits a (set of) result(s). When the service has the result(s) from the Validator(s), it (creates a bundle of results and) returns it to the client.



❖ Packaging/Modularization

All of the classes related to the UI Component Validation Service are packaged in com.ptc.core.ui.validation. Their source is located in the module \Windchill\src\com\ptc\core\ui\validation. It is strongly recommended that any time a developer is doing Validator development, they update all of the files in this directory and compile the latest versions of the Java classes.
Developers writing Validators should put their Validator classes in a package or module that is meaningful for the action(s)/UI component(s) that the Validator validates.

❖ Authoring a Validator Class

To create a Validator, you'll need to create a subclass of com.ptc.core.ui.validation.DefaultUIComponentValidator.java. There are currently five public methods defined in DefaultUIComponentValidator.java. Your subclass can override one to all of them:

performFullPreValidation() , performLimitedPreValidation(), validateFormSubmission()

validateSelectedAction() , validateSelectedMultiSelectAction()

For those methods which you do not override, the default behavior (always enable/permit) will be inherited from the DefaultUIComponentValidator class.

You will also need to create a properties entry to associate your Validator class with a particular validation key (action). The validation service uses these entries to find the right Validator for a given validation key (action). The entry will go in service.properties, or your application team's service properties file (ask your group lead where you should put your entry), and should have this format:

wt.services/rsc/default/com.ptc.core.ui.UIComponentValidator/**<validationKey>**/
null/0=**com.ptc.my.validators.MyValidator**

Where <*validationKey*> is the validation key for your action/component and the right-side value is the fully-qualified class name of your Validator.

There are three types of checks you should never have to perform in your Validator implementations. These checks are performed by the validation service before the Validators are called. They include:

- ☐ Role-based checking (visibility of actions based on input into the RBUI system, which is not to be confused with access control checking, which needs to be done in the Validators.)
- ☐ Install-based checking (should an action or UI component be available given the set of solutions installed on a given system?)
- ☐ Client-based checking (should an action or UI component be available in a given client, like DTI or PSE?)

6.1 Requirement I: Example of Pre Validation (Action visibility based on container type)

Want to create a validator so that our action named "createObject" (to invoke Create Object wizard) will only be visible in PDMLinkProduct context.

- **Step 1 :**

Write CreateObjectValidator.java to validate the action.

- **Step 2 :**

Register validator in site.xconf

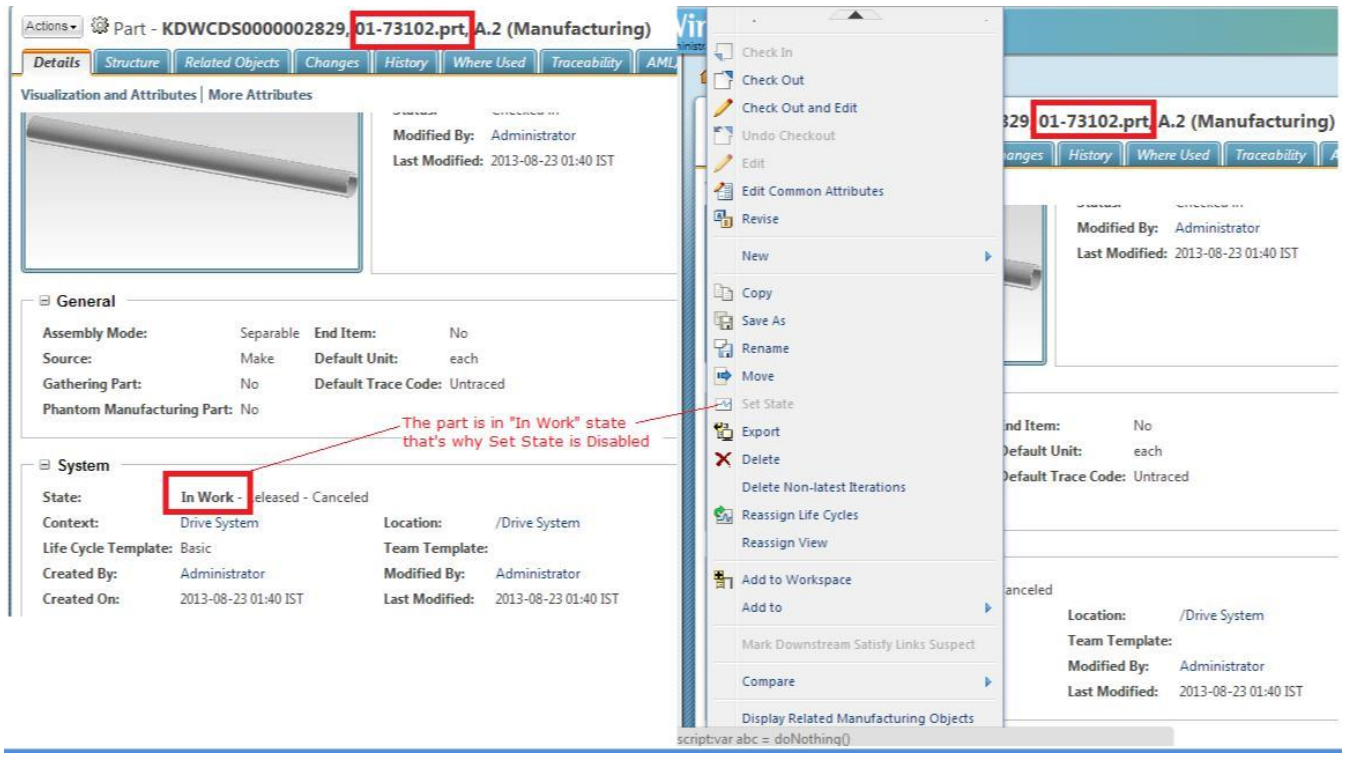
```
<Service context="default" name="com.ptc.core.ui.validation.UIComponentValidator"
        targetFile="codebase/service.properties">
    <Option cardinality="duplicate" order="0" overridable="true" requestor="null"
          selector="createObject"
          serviceClass="ext.test.validator.CreateObjectValidator"/>
</Service>
```

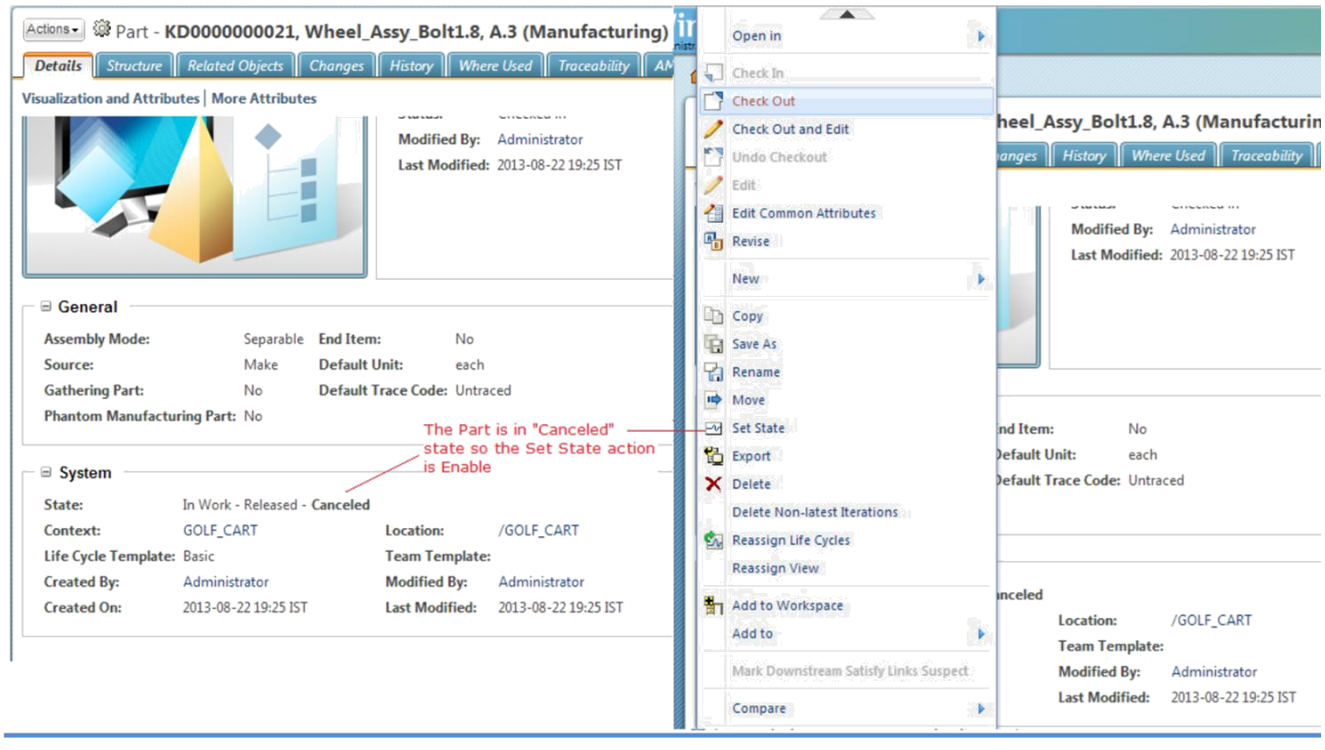
- **Step 3 :**

Start/ Restart server after compiling site.xconf using xconfmanager –p

6.2 Requirement II : OOTB Action visibility based on State

Based on the life cycle state's value the action "Set State" is Disabled or Hidden.
In OOTB windchill, administrators has the privilege to set the state of WTPart using Set State action.
In this example if the WTPart is in "In Work" state then the action is Disabled for all user including administrator.





To implement the solution we need a validator to validate the set state action.

❖ **Step 1 :**

Using Action report generated by Action tool (look at the 2nd point of Code Snippet section for a detail description) find out the action name of Set State.

❖ **Step 2 :**

Create ext.test.validator.SetStateValidator

❖ **Step 3 :**

Register it in site.xconf, compile the xconf using xconfmanager -p and start/restart the server.

```
<Service context="default" name="com.ptc.core.ui.validation.UIComponentValidator"
        targetFile="codebase/service.properties">

    <Option cardinality="duplicate" order="0" overridable="true" requestor="null"
          selector="SETSTATE"
          serviceClass="ext.test.validator. SetStateValidator"/>
</Service>
```

Similarly validator can be used to disable/enable/hide an action based on Part's view , sub-type name or combination of both etc.

6.3 Requirement III : Post Submit Validation of one attribute's value based on other attribute's value

Validate the value of ItemGroup (IBA of WTPart) at edit multiple wizard based on the value of Source.

Ex. If source is "**make**" then the permissible values are **"**210375-Lubricant & Industrial Oil,210525-Paints & Varnishe,210650-Welding Electrode **"** and if the value is **"buy"** it's "010000-Fabricated Item,020000-Machined Part,030000-Machined Fabrication" etc.

❖ **Step 1 :**

Put all the permissable values in a legal value list / enumerated list and attch that with the IBA. Permissable values are **010000-Fabricated Item|020000-Machined Part|030000-Machined Fabrication|040000-Assembly/Sub-assembly|050000-Prepared Material|160000-Bearing|170000-Coupling|210375-Lubricant & Industrial Oil|210525-Paints & Varnishe|210650-Welding Electrode** Create ext.test.validator.SourceValueValidator to validate whether the values are in permissible range.

❖ **Step 2 :**

Register the validator. Propagate using xconfmanager –p .

```
<!-- Validator -->
<Service context="default" name="com.ptc.core.ui.validation.UIComponentValidator"
      targetFile="codebase/service.properties">
      <Option cardinality="duplicate" order="0" overridable="true" requestor="null"
                selector="editMultiObjects"
                serviceClass="ext.tgs.validator.SourceValueValidator"/>
</Service>
```

Start/Restart the server.

7. MISCELLANEOUS

7.1 Requirement I : Create custom sequence in Oracle Database without Info Modeller

Want to create custom oracle sequence for WC object(or their Sub-Type) like WTPart,WTDocument etc and call the sequence from OIR.

1. Log in sqlplus with Windchill schema user.

2. Create Oracle sequence by running the following command:

```
CREATE SEQUENCE customers_seq
 START WITH             1000
 INCREMENT BY            1
 NOCACHE;
```

3. Download OIR file of the desired object, then change the sequence name with newly created sequence name.

```
<!-- set the number to a generated number -->
<AttrValue id="number"
algorithm="com.ptc.windchill.enterprise.revisionControlled.server.impl.NumberGenerator">
<Arg>{GEN:wt.enterprise.SequenceGenerator:customers_seq:8:0}</Arg>
</AttrValue>
```

4. Upload this OIR.

7.2 Requirement II : Reduce display time of inline message

Want to reduce the display time and fade out time of Inline messaging.

1. In 10.0: Open file <WT_HOME>\codebase\netmarkets\javascript\util\jsfrags\messaging.jsfrag in a text editor
   In 10.1: Open file <WT_HOME>\codebase\netmarkets\javascript\util\jsfrags\messaging\messaging.jsfrag in a text editor

2. Locate following content:

```
/** after user has not touched the message window for 4 seconds, begin to fade it out */
PTC.messaging.deferTimeoutMill = 4000;
```

3. Change the value of PTC.messaging.deferTimeoutMill from 4000 (4 seconds) to 1000 (1 second), this will shorten the display time for the inline message

4. Locate following content:

```
PTC.messaging.messageFadeout = function(){
    PTC.messaging.msgWindow.getEl().fadeOut({
        endOpacity: 0.01,
        duration: 6,
                callback: PTC.messaging.closeInlineMessageFadeout
    });
};
```

5. Change the duration value from 6 to 1, this will shorten the fade out time for the inline message

6. Save the file

7. Execute following command from a Windchill

   shell: ant -f bin\jsfrag_combine.xml

8. Restart Windchill

9. Open a new browser session to observe the change.

7.3 Requirement III : Create drop down list in WF activity variable

In OOTB windchill if we create a variable in workflow activity as data type String, in task page it's rendered as TextArea. But what if we want a drop down list to select the value of the variable? There is no OOTB feature to fulfill this but using a liitle customization we can achieve that.

(Procedure Courtsey :- piyush.chhabra@itcinfotech.com)

1. Create a custom rbInfo file with your own set of values. For example I am using my own custom rbInfo file named EnhancementTypeRB.rbInfo.

pickerAttributes.xml | [redacted] | EnhancementTypeRB.rbInfo

```
1  ResourceInfo.class=wt.tools.resource.EnumResourceInfo
2  ResourceInfo.customizable=true
3  ResourceInfo.deprecated=false
4
5  CAPACITY.value=500 ton
6  COMPANYNAME.value=ITC
7  OTHERS.value=Others
8  PURPOSE.value=Test
```

2. Put the rbInfo file in <WINDCHILL_HOME>/src/ext/test/common folder.(My java file's package name is ext.test.common that's why I have to put the rbInfo file in exact same folder)

3. Make sure that the extension is rb**I**nfo with a capital "I". In windows server the extension is case insensitive but in Linux or in UNIX server it is case sensitive.

4. In windchill shell run a command ResourceBuild <packagename>.<rbInfo file name>. In my example the command will be ResourceBuild ext.test.common.EnhancementTypeRB (In Unix ./ResourceBuild.sh ext.test.common.EnhancementTypeRB)

5. An EnhancementTypeRB.RB.ser file will be made of that rbInfo file that will serve as our resource bundle.

6. Later you can add more value using enumCustomize from windchill shell. If you add any value using enumCustomize then you have to run ant –f codebase/MakeJar.xml from windchill shell.

7. Add below property in site.xconf

   wt.workflow.definer.variable.ShowWfEvolvableWarning=true

   and run xconfmanager –p to propagate it in wt.properties. So that the non persistable and non evolvable classes can be used to define workflow variables.The user will receive a warning message but user can create or update non-persistable and non evolvable objects into variables.

8. Put the java file in the corresponding package.(In my example it is ext.test.common)

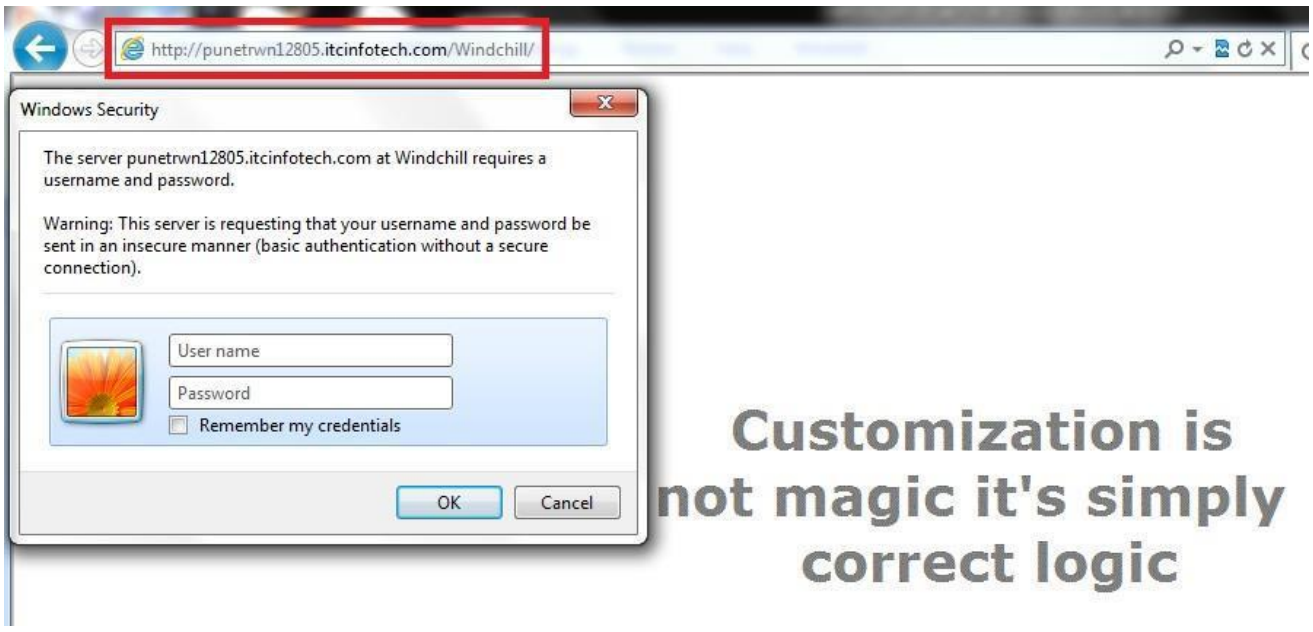
9. Restart the server.

10. Choose the data type of a variable in the activity of a workflow as other class and specify the class. In my example it is ext.test.common.EnhancementType

7.4 Requirement IV : Change the PTC logo at the time of logon

Want to change the PTC logo with any other logo or image which is visible at the time of Log In.

1. Replace <Windchill_Home>\Windchill\codebase\wtcore\images\bblks\ l_ptc_splash.gif image with your own image file.
2. Rename your image to **l_ptc_splash.gif(482 * 205)**
3. Restart Windchill.



7.5 Requirement V (System banner alert message):

You can create a system wide banner message that will display on all the pages within the Windchill application.

Turning on the alert banner feature

1. Put the following line code into <Windchill>\codebase\netmarkets\jsp\util\begin_custom.jspf <%@ include file="/netmarkets/jsp/util/banner_custom.jspf"%>

2. Restart Tomcat.

3.Write the alert message with **html format** into the following file:

<WT_HOME>\codebase\netmarkets\jsp\util\banner.txt

For example:

```
<div style="float:left;clear: both;position: absolute;margin: 2px 170px;padding: 3px;border: 1px solid
#000;background-color: #fff;top: 3px;color: #006600;font-weight: bold;font-size: 12px;z-index: 1000">
 <font face="verdana">
<!-- Banner begins here -->
   Windchill server will be down 1/1 for maintenance
Keyword Search works ...use with understanding! ->
<!-- Banner ends here -->
</font>
</div>
```

Removing the alert message from Windchill UI page

Set windchill_home\codebase\netmarkets\jsp\util\banner.txt as empty file.

In Banner.txt the message must be in html format. If your message involved any kind of java code then you have to write it in “banner_custom.jspf”.The following example will show the hostname of the server as a banner message. For example if you want to show the banner host name in the banner

```
<%@ page import="com.ptc.netmarkets.util.NmUtils, com.ptc.netmarkets.util.beans.NmHelperBean"
%><%@ page contentType="text/html; charset=UTF-8"
%><%response.setContentType("text/html; charset=UTF-8");
%><%
  NmHelperBean helper = new NmHelperBean(pageContext);
  try {
    %><jsp:include page="<%=NmUtils.getLocaleSpecificFileName(helper.getNmLocaleBean().getLocale(),
\"/netmarkets/jsp/util/banner\", \"txt\") %>" flush="true"/><%//
  }
  catch (Throwable t){
    helper.getNmErrorBean().setThrowable(t);
    %><jsp:include page="/netmarkets/jsp/util/error.jsp" flush="true"/><%//
    helper.getNmErrorBean().setThrowable(null);
}
%><% java.net.InetAddress localMachine = null;
            try {
                  localMachine = java.net.InetAddress.getLocalHost();
            } catch (java.net.UnknownHostException e) {
                   e.printStackTrace();
            }
            String hostName = localMachine.getHostName();
%><%@ page contentType="text/html; charset=UTF-8"
%><%response.setContentType("text/html; charset=UTF-8");%>
<html><head>
<style type="text/css">
wc {color:white;
font-weight:bold;font-size:25px;
padding:0% 35% 0% 9%;
}
</style></head><body><marquee behavior="alternate">
<!-- Banner begins here -->  <wc>This is <%=hostName%></wc> <!-- Banner ends here -->
</marquee>
</body>
</html>
```

❖ Screenshot:

7.6 Requirement VI : Cerate and call custom locale file

It’s not advisable to hard code the internal names (Internal name of IBA, Subtype, Lifecycle state etc) , custom message (Message which will be displayed to user to guide them) etc. in customize code.
Because if any subtypes name got changed or user asked to change displayed message then developer has to check all the classes and modify where neseccary. And moreover the display messages should be locale configurable.
To avoid this kind of scenario there are few ways.

1> **Use @RBEntry annotation :-**

Create a class CustomMessageResource.java

```
package ext.test.resource;

import wt.util.resource.RBEntry;
import wt.util.resource.RBUUID;
import wt.util.resource.WTListResourceBundle;

@RBUUID("ext.test.resource.CustomMessageResource")

public final class CustomMessageResource extends WTListResourceBundle {


    @RBEntry("Check the given number")
    public static final String PRIVATE_CONSTANT_00 = "CHK_NUMBER";

    @RBEntry("You dont have access to this object.Contact your administrator")
    public static final String PRIVATE_CONSTANT_01 = "ACCESS_DENIED";
}
```

Now in your code where you want those custom message use

WTMessage.*getLocalizedMessage*("ext.test.resource.CustomMessageResource","ACCESS_DENIED",null,null);
First parameter is the fully qualified name of the Resource class. **Value can't be null**
Second parameter is the key mentioned in the class. (i.e for the message "Check the given number" the key is CHK_NUMBER) **Value can't be null.**
Third parameter is the array of object. **Value can be null**
Fourth parameter is the Locale. **Value can be null. If null is given then the locale will be derived internally using WTContext.*getContext*().getLocale();**

2> **Using RBInfo file :-**
Create a RBInfo file (Sample below).

```
ResourceInfo.class=wt.tools.resource.StringResourceInfo
ResourceInfo.customizable=true
ResourceInfo.deprecated=false
#ResourceInfo.UUID=9d18c407-9b08-495c-a0ff-bb9e9cab51ca

#       Entry Format (values equal to default value are not included)
#       <key>.value=
#       <key>.comment=
#       <key>.argComment<n>=
#       <key>.constant=
#       <key>.customizable=
#       <key>.deprecated=

#       Entry Contents

REPORT_TITLE.constant=REPORT_TITLE
REPORT_TITLE.value=Change Throughput/CycleTime Report

ALERT_REPORT_TYPE.constant=ALERT_REPORT_TYPE
ALERT_REPORT_TYPE.value=Select Report Type
```

Run ResourceBuild <File Name> in windchill shell.(For linux/Unix it will be ./ResourceBuild.sh)

If your file name is CustomAlertRB.rbInfo and it's under src/ext/test then the command will be
ResourceBuild ext.test.CustomAlertRB

Now in your code where you want those custom message use

WTMessage.*getLocalizedMessage*(<rbInfo file name i.e ext.test.CustomAlertRB>,<key>,**null**,null);

7.7 Requirement VII : To extend Global Attribute column length by steps :

1. Create **valueModel.properties** file in **<Windchill>\wtCustom\wt\iba\value**, and add the following to the file (2000 is the maximum length need to extend to):

```
StringValue.value.UpperLimit=2000
StringValue.value2.UpperLimit=2000
```

2. Run the following in Windchill Shell:
   - **ant -f bin\tools.xml custom_column -Dgen.input=wt.iba.value.***
     Note: **Information*Modeler** must be installed.
3. Perform **inforeport wt.iba.value.StringValue** in Windchill Shell to make sure UpperLimit has been changed in generated file **<Windchill>\temp\value.StringValue.out**.
4. Please check if "**wt.db.maxBytesPerChar=3**" is set in **wt.properties**.
   1. If yes, generated sql located in <**Windchill>\db\sql3\wnc\Foundation\wt\iba\value\create_StringValue_Table.sql**
   2. If no, generated sql located in <**Windchill>\db\sql\wnc\Foundation\wt\iba\value\create_StringValue_Table.sql**
   3. If Windchill is using a Microsoft SQL Server, generate SQL is located in**<Windchill>\db\sqlServer\wnc\Foundation\wt\iba\value**

      Instructions for Oracle database
      - Log in as Windchill DB user, using sqlplus
      - Execute below SQL script to create a backup table:
        - **create table StringValue_bak as select * from StringValue;**
      - Execute below SQL script to drop a existing table:
        - **drop table StringValue;**
      - Execute "create_StringValue_Table.sql"
      - Execute below SQL script to insert all rows from backup table:

- **insert into StringValue select * from StringValue_bak;**
- **commit;**

Recreate the indexes

- Open **<Windchill>\db\sqlServer\wnc\Foundation\wt\iba\value\create_StringValue_Index.sql** and replace **$USER_NAME** with the Windchill DB user name
- Execute changed **<Windchill>\db\sqlServer\wnc\Foundation\wt\iba\value\create_StringValue_Index.sql**

Restart Windchill.

☐ More details refer:- https://www.ptc.com/appserver/cs/view/solution.jsp?source=subscription&n=CS94753

7.8 Requirement VIII : Make workflow comment mandatory

To make the comment field of WorkFlow task as mandatory.

Instructions for Windchill PDMLink

- Create the following entry in **<WT_HOME>/codebase/config/actions/custom-actions.xml**

```
<objecttype name="work" class="wt.workflow.work.WorkItem">
  <action name="complete" enabledwhensuspended="true" ajax="page">
  <command class="com.ptc.netmarkets.work.NmWorkItemCommands" method="complete"
onClick="validateComments()"/>
  </action>
</objecttype>
```

- Create a file **custom.jsfrag** under **<WT_HOME>/codebase/netmarkets/javascript/util/jsfrags** and include a Javascript function to perform the validation

```
function validateComments(){
if( (document.getElementById("workitem_comment").value == "") ||
(document.getElementById("workitem_comment").length == 0)){
  alert("Comments are required!!");
  return false;
}
  return true;
}
```

- Run: **ant -f bin/jsfrag_combine.xml**
- Restart Windchill

7.9 Requirement IX : Auto generate Project Number

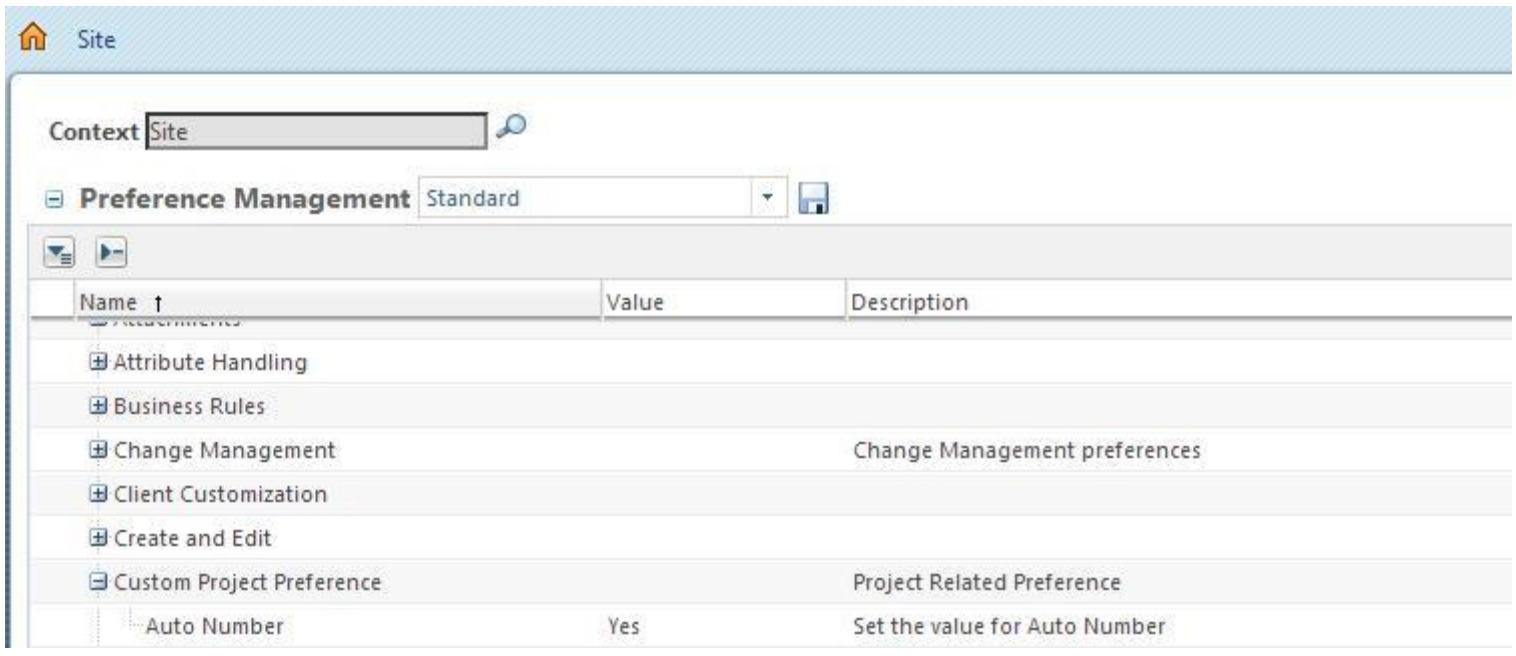
Project Number will be auto genareated based on custom preference's value.

❖ **Step 1 :**

Create sequence "project_seq" in database as described earlier.

❖ **Step 2 :**

Create Custom Preference "Custom Project" inside which there will be "Auto Generated" which can only take Boolean value.



☐ Create a resource bundle for labels used for your new preference. Labels are needed for display name and description of preference category which the new preference will be visible under in **Preference Management** UI. Labels are also needed for the display name, description and long description of your preference. Create the file projectResource.rbInfo in package ext.tgs.project. In this example, this file would be added to the <Windchill>/src/ext/tgs/project directory

projectResource.rbInfo

```
1  ResourceInfo.class=wt.tools.resource.StringResourceInfo
2  ResourceInfo.customizable=true
3  ResourceInfo.deprecated=false
4
5  # Preference Category labels
6  ProjectCategory.displayName.value=Custom Project Preference
7  ProjectCategory.description.value=Project Related Preference
8
9  # Preference Definition labels
10 MyProjectPreference.displayName.value=Auto Number
11 MyProjectPreference.description.value=Set the value for Auto Number
12 MyProjectPreference.longDescription.value=If the preference is set to true then ProjectNumber will be Auto Generated
```

☐ Build the resource bundle by executing the following command from a windchill shell:

ResourceBuild ext.tgs.project

☐ Restart the MethodServer.

☐ Create Preference load file: prefLoad.xml. It will contain a definition for the new preference category and new preference definition.

```
<?xml version="1.0"?><!DOCTYPE NmLoader SYSTEM "standardX10.dtd">
<NmLoader>
        <csvPreferenceCategory handler="wt.preference.LoadPreference.createPreferenceCategory">
                <csvname>CUSTOM_PROJECT_PREFERENCE_CATEGORY_NUMBER</csvname>
                <csvparentName></csvparentName>
                <csvdisplayName>
                        ext.tgs.project.projectResource:ProjectCategory.displayName
                </csvdisplayName>
                <csvdescription>
                        ext.tgs.project.projectResource:ProjectCategory.description
                </csvdescription>
        </csvPreferenceCategory>
        <csvPreferenceDefinition handler="wt.preference.LoadPreference.createPreferenceDefinition">
                <csvname>ext/tgs/ProjectPreference</csvname>
                <csvvisibility>SITE</csvvisibility>
                <csvcategoryName>CUSTOM_PROJECT_PREFERENCE_CATEGORY_NUMBER</csvcategoryName>
                <csvdisplayName>ext.tgs.project.projectResource:MyProjectPreference.displayName</csvdisplayName>
                <csvdescription>ext.tgs.project.projectResource:MyProjectPreference.description</csvdescription>
                <csvlongDescription>ext.tgs.project.projectResource:MyProjectPreference.longDescription</csvlongDescription>
                <csvdefaultValue>true</csvdefaultValue>
                <csvhandler>com.ptc.windchill.enterprise.preference.handler.BooleanPreferenceValueHandler</csvhandler>
        </csvPreferenceDefinition>
        <csvLinkPreferenceClientDefinition handler="wt.preference.LoadPreference.setClientDefinitionLink">
                <csvname>ext/tgs/ProjectPreference</csvname>
                <csvclientName>WINDCHILL</csvclientName>
        </csvLinkPreferenceClientDefinition></NmLoader>
```

7.10 Requirement X : Insert parameters into a Creo Parametric model upon download

Customizing the Parameters in the Download Service

Windchill provides a server-side delegate that can be used to insert parameters into a Creo Parametric model upon download. This mechanism can be used to pass information from the server down to Creo Parametric, where it can be used like any other Creo Parametric parameter (for example, to place information on drawing forms). Parameters beginning with "PTC" or "PROI" are regarded as

reserved system parameters and cannot be propagated by the customization. If they are added in the customization, they are ignored by the download service.

Below are the steps involved to use Download Service.

- **Step 1 :**

- Create a Java class that implements the interface ModeledAttributesDelegate.

  The OOTB example "DefaultModeledAttributesDelegate" is attached in src.

- Edit site.xconf file (found in <Windchill>) to add following property to indicate availability of customization service on the server:

```
<Service context="default"
        name="com.ptc.windchill.uwgm.proesrv.c11n.ModeledAttributesDelegate"
        targetFile=codebase/service.properties">

        <Option cardinality="singleton" requestor="java.lang.Object"
        serviceClass="com.ptc.windchill.uwgm.proesrv.c11n.DefaultModeledAttributesDelegate"/>
```

</Service>

- Then use the xconfmanager tool to apply the changes to service.properties file (run xconfmanager -p)

  Use the path of your class in place of value of serviceClass (that is, replace com.ptc.windchill.uwgm.proesrv.c11n.DefaultModeledAttributesDelegate with the path to your class).

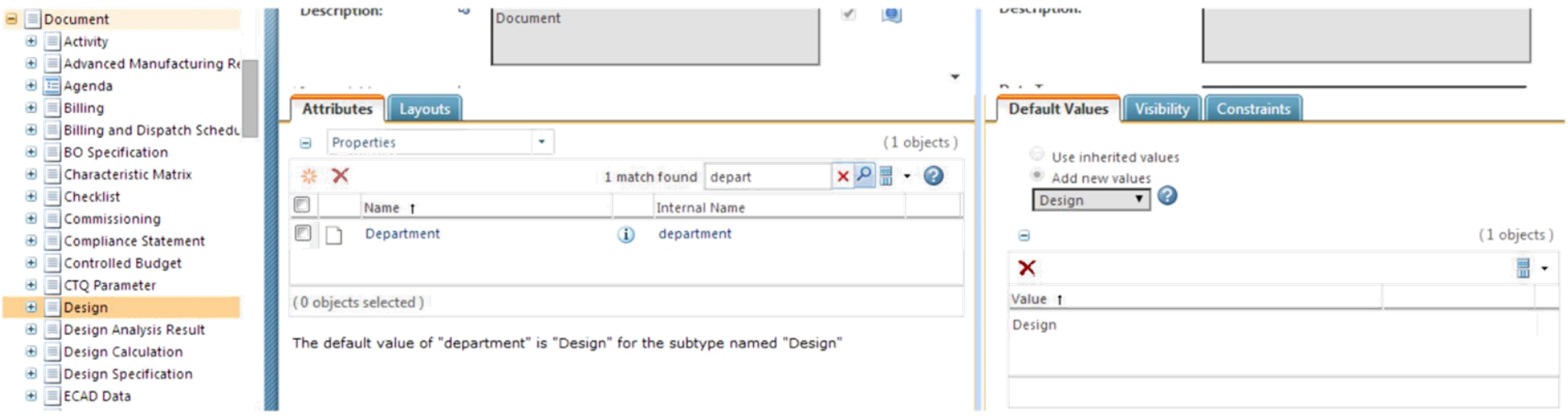
- Restart the method server.

7.11 Requirement XI : Filter Document Soft Type

Filter document soft type based on the IBA value in Create New Document wizard.
Ex. There is one IBA in Document name "department" and it has values as "Project, Supplay Chain, Design , Standard" etc.
Now, we will change the OOTB create wizard into a three step wizard. In first step user will select the Department and based on that selection in second step we will filter the soft type of the Document.

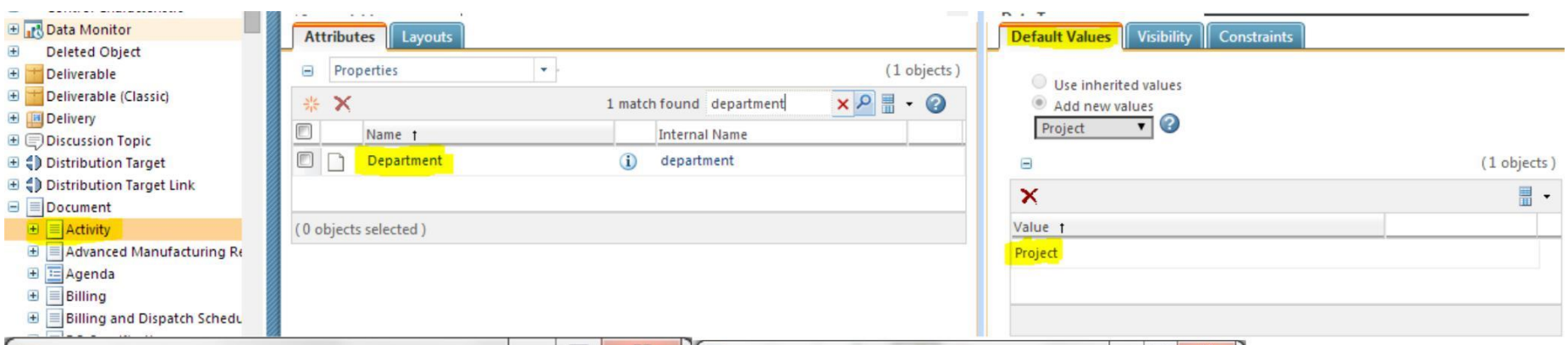
- Solution Approach :

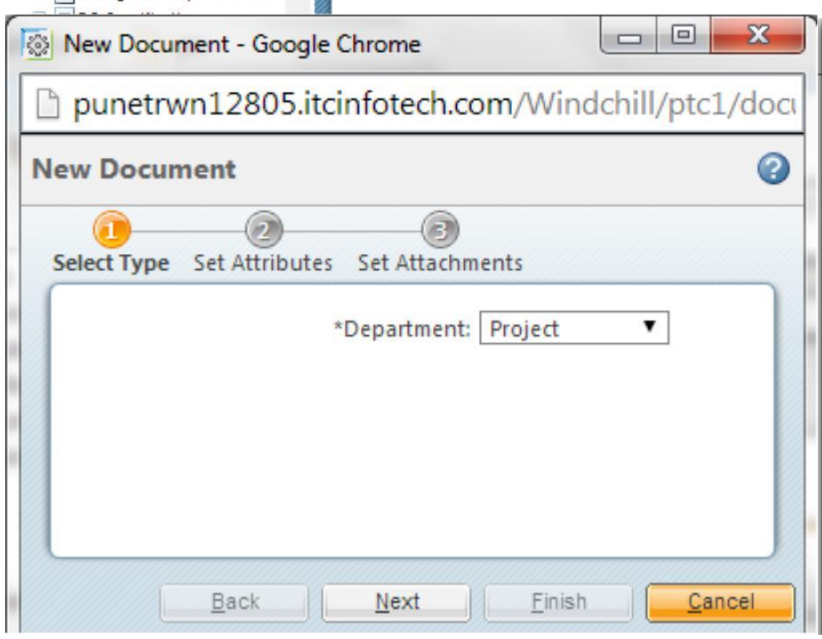
Create one IBA(global) named "department" on "WTDocument" and also Create a global enumeration named "department" with the values "Project,Supplay Chain,Design,Standard" and attach it with the IBA.
Go to the subtype of document and provide default value of the IBA "department" as required.

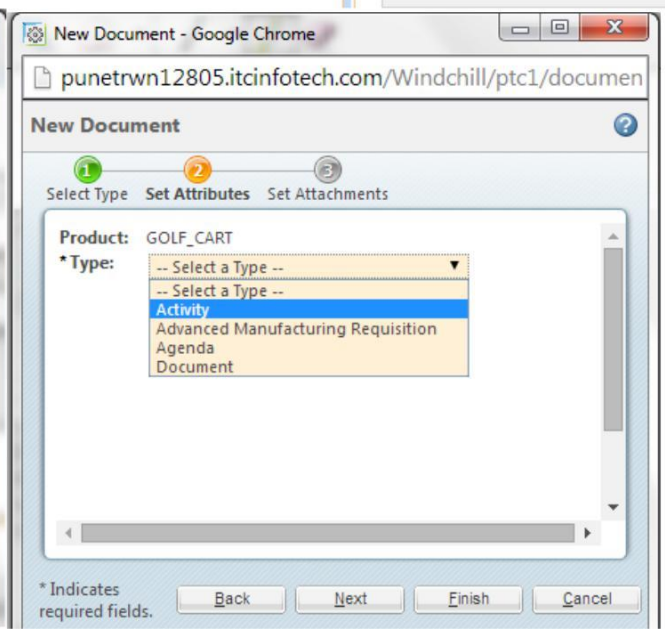


The above image is showing the default value of the IBA in a sub-type.

If user select department as "Design" then in next step the subtype "Design" will be in the list.
Similarly if user select "Project" then "Activity" will be in the list.







Below are the list of files used in this customization.
Please take necessary back up of any OOTB file(s).

The code still contains Sysout statements or comment line for better understanding.
All the files mentioned here are present in IBA.zip .

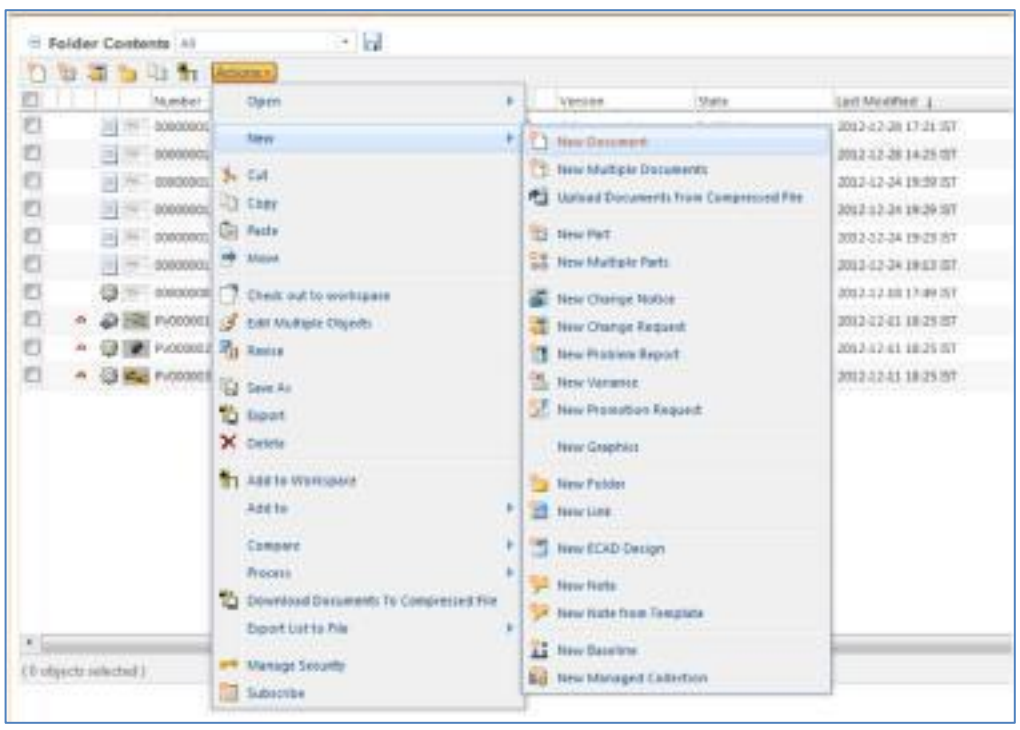
Contains various useful methods to find the default value of an IBA, all the subtype of a parent type, Display Name of the subtype from its internal name, Display name of global enumeration from its internal name etc.

File Name	File Location (w.r.t codebase)	OOTB
CustomActionResource.java	ext/test/resource	No
Constant.java	ext/test	No
IBAReader.java	ext/test	No
Document.properties	ext/itc	No
selectType.jsp	netmarkets/jsp/itc	No
custom-actions.xml	config/actions	Yes
createDocumentSetTypeAndAttributes WizStep.jsp	netmarkets/jsp/document	Yes
create.jspf	netmarkets/jsp/document	Yes

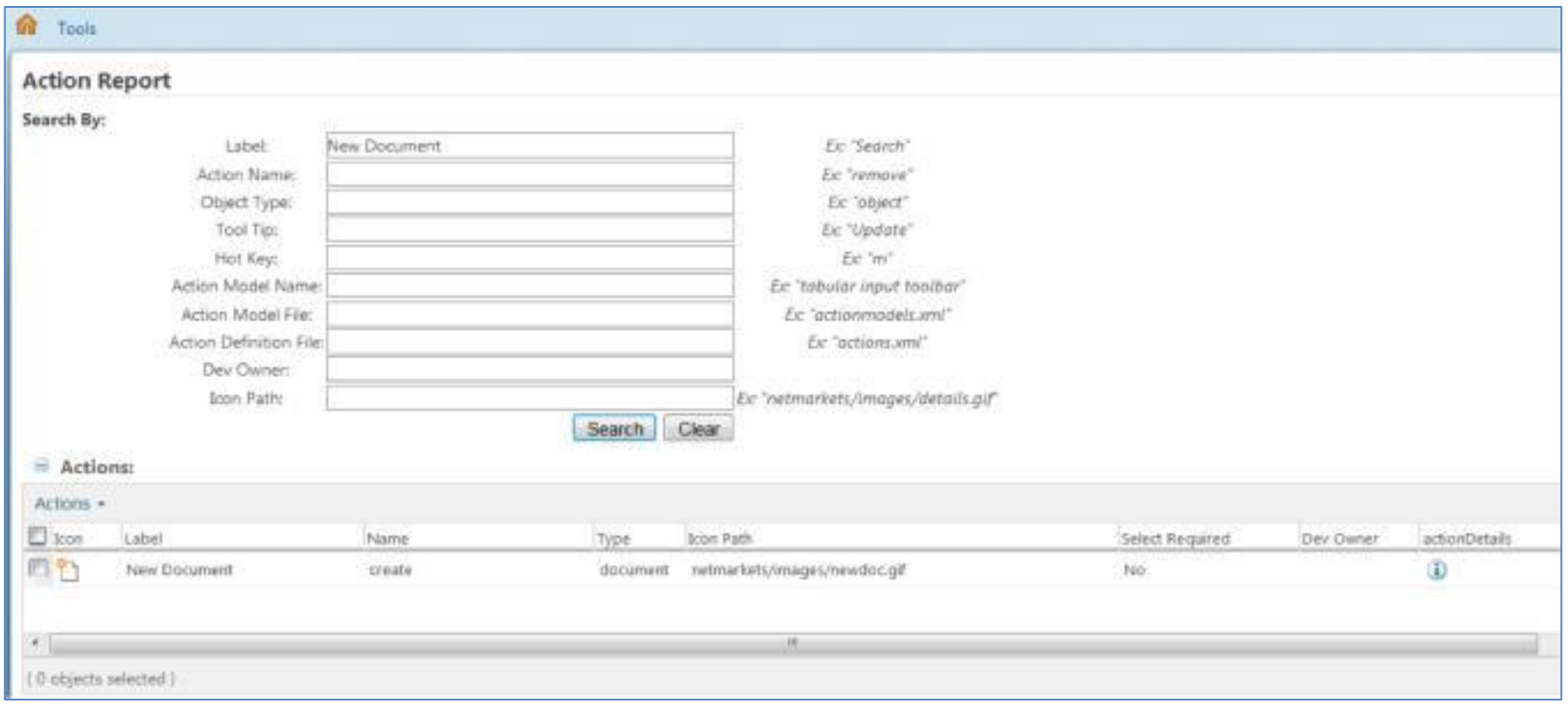
8. CODE SNIPPET

8.1 Using Action Report tool in windchill :

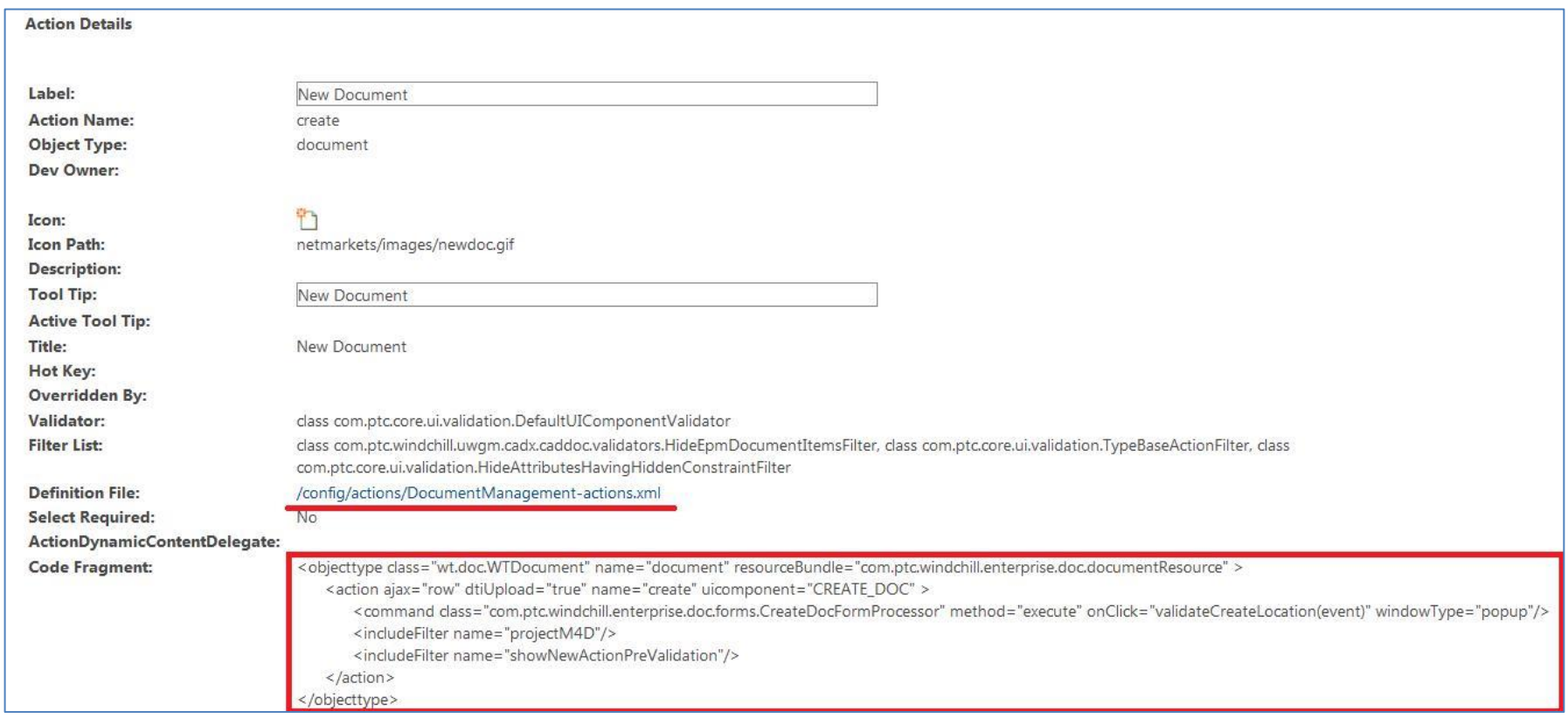
Below I explained how to know which JSP is invoking after you click on "New Document" action.
To find the JSP first you have to find the action's label or tool tip after clicking which the document wizard is pop-upping



Write "New Document" at label box or at tooltip box and press search. Your will redirect to search result page.



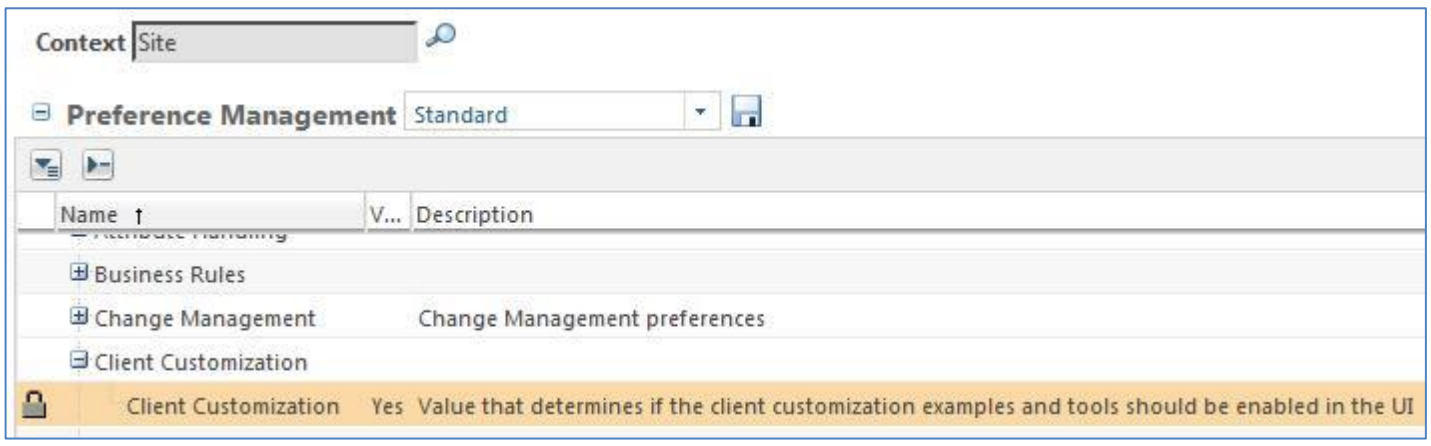
Click on the action details button of new document. Then you will redirect to another page contained details information about that action like action name, label, Icon Path, Definition File etc.



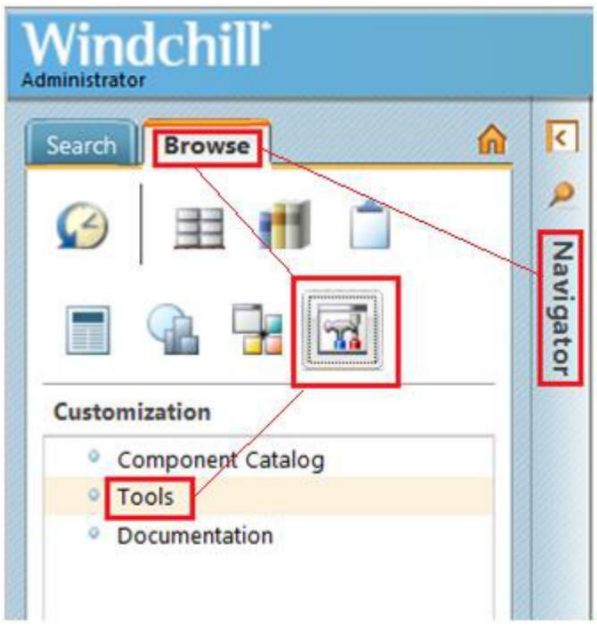
From that page you will get to know about the Definition file (**DocumentManagement-actions.xml**), the action name is ("**Create**") and after clicking that action which jsp page is showing

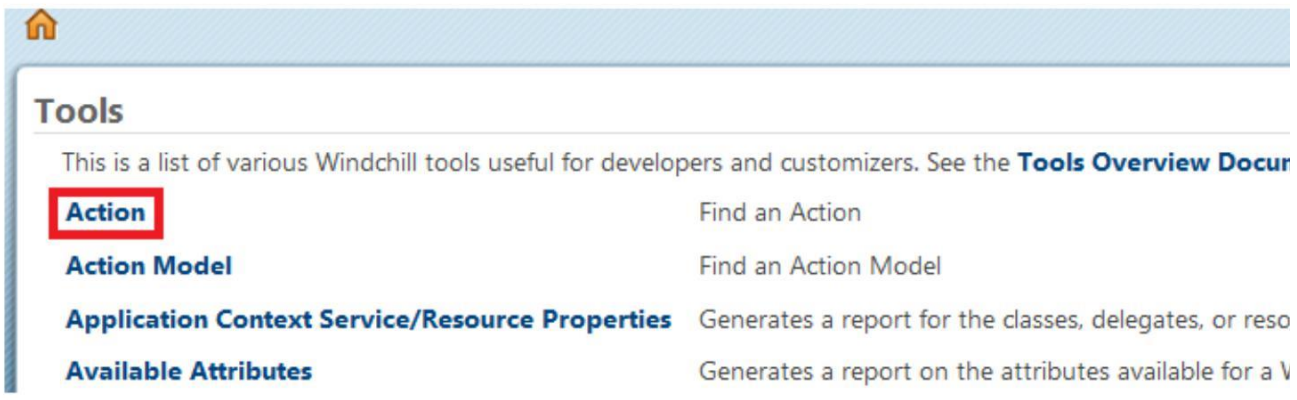
You can use this technique for almost any Action which is defined in XML's.

If the customization tab is unavailable in Navigation → Browse then go to site→utilities→Preference management and set the preference for client customization as yes

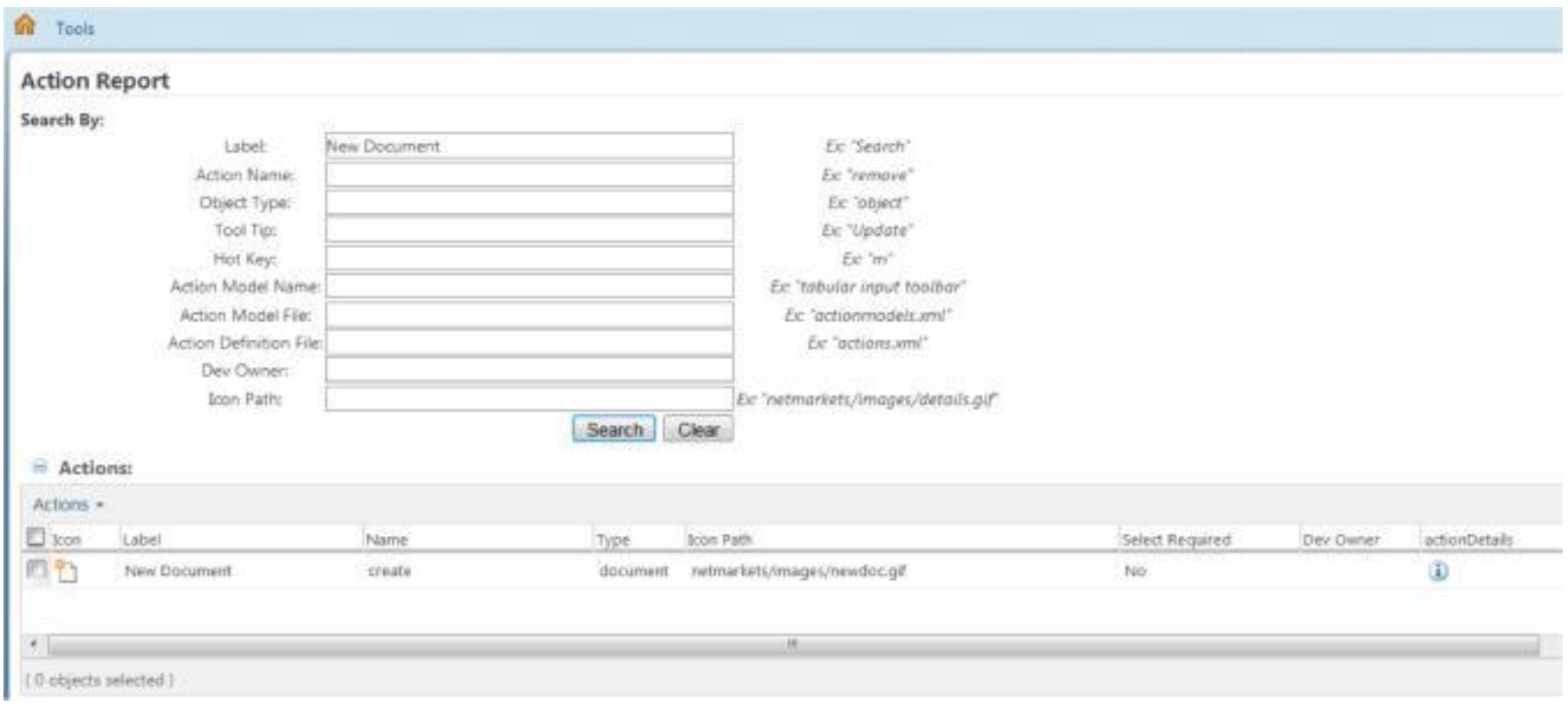


After that go to Navigator ☐ Browse ☐ Customization ☐ Tools ☐ Action





Write "New Document" at label box or at tooltip box and press search. Your will redirect to search result page.



Click on the action details button of new document. Then you will redirect to another page contained details information about that action like action name, label, Icon Path, Definition File etc.

Action Details

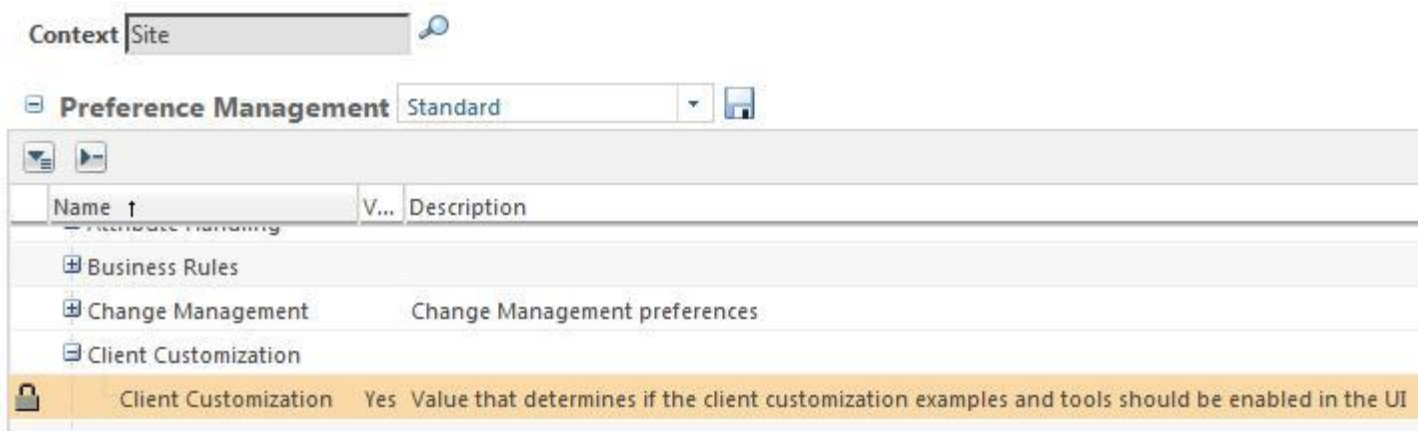
Label:	New Document
Action Name:	create
Object Type:	document
Dev Owner:	
Icon:	
Icon Path:	netmarkets/images/newdoc.gif
Description:	
Tool Tip:	New Document
Active Tool Tip:	
Title:	New Document
Hot Key:	
Overridden By:	
Validator:	class com.ptc.core.ui.validation.DefaultUIComponentValidator
Filter List:	class com.ptc.windchill.uwgm.cadx.caddoc.validators.HideEpmDocumentItemsFilter, class com.ptc.core.ui.validation.TypeBaseActionFilter, class com.ptc.core.ui.validation.HideAttributesHavingHiddenConstraintFilter
Definition File:	/config/actions/DocumentManagement-actions.xml
Select Required:	No
ActionDynamicContentDelegate:	
Code Fragment:	

```
<objecttype class="wt.doc.WTDocument" name="document" resourceBundle="com.ptc.windchill.enterprise.doc.documentResource" >
    <action ajax="row" dtiUpload="true" name="create" uicomponent="CREATE_DOC" >
        <command class="com.ptc.windchill.enterprise.doc.forms.CreateDocFormProcessor" method="execute" onClick="validateCreateLocation(event)" windowType="popup"/>
        <includeFilter name="projectM4D"/>
        <includeFilter name="showNewActionPreValidation"/>
    </action>
</objecttype>
```

From that page you will get to know about the Definition file (**DocumentManagement-actions.xml**), the action name is ("**Create**") and after clicking that action which jsp page is showing

You can use this technique for almost any Action which is defined in XML's.

If the customization tab is unavailable in Navigation → Browse then go to site→utilities→Preference management and set the preference for client customization as yes



To create a resource bundle you should extend WTListResourceBundle and your file name should ends with Resource or RB like MyResource.java or CustomRB.java else it will not work properly.

8.2 Standard Code for Create WTPart and WTDocument:

```
WTPart part = WTPart.newWTPart("Myname","MyNumber");
        try {
                part.setContainer(nmCommandBean.getContainer()); Persistable
                persistable = PersistenceHelper.manager.save(part);
        } catch (WTPropertyVetoException e) {
                //        TODO Auto-generated catch block
                e.printStackTrace();
        }
```

You can give others attribute value also using get method of WTPart class but you have to mention at least name and number.
Similarly you can create WTDocument.

8.3 From OR to create the Object :

```
ReferenceFactory factory = new ReferenceFactory();
ObjectReference objRef = (ObjectReference)factory.getReference("OR: wt.pdmlink.PDMLinkProduct:162524");
PDMLinkProduct product=(( PDMLinkProduct) objRef.getObject());
```

8.4 From VR to the Object :

```
ReferenceFactory factory = new ReferenceFactory();
VersionReference objRef = (VersionReference) factory.getReference("VR:wt.doc.WTDocument:164316");
WTDocument doc = (WTDocument) objRef.getObject();
```

8.5 From Object to ObjectReference :

```
ObjectReference objReference = ObjectReference.newObjectReference("my_persistable");
```

8.6 To create a subtype through code

```
wt.type.TypeDefinitionReference typeRef =
wt.type.TypedUtilityServiceHelper.service.getTypeDefinitionReference("wt.doc.WTDocument|com.ITCINFOTECH.Age nda");
WTDocument document = WTDocument.newWTDocument();
document.setTypeDefinitionReference(typeRef);
```

8.7 Giving value to an IBA through code (This will not work in 9.1):-

```
LWCNormalizedObject obj = new LWCNormalizedObject(my_persistable,null,Locale.US,new
UpdateOperationIdentifier());
obj.load("attributeA","attribtueB");obj.set("attributeA",Boolean.TRUE);
obj.set("attribtueB","Value of attribute B");
obj.apply();
PersistenceHelper.manager.modify(my_persistable);
```

If your IBA is MultiValued then you can pass Array also in the "set" method.

For further query please refer to Windchill java API documentation explained very well using example. The location of documentation : - <WT_HOME>\codebase\wt\clients\library\api\Index.html

In case of 10.2 or higher use PersistableAdapter class.

8.8 Change the master of part without changing the iteration

- **Want to change only name and number**

  ```
  WTPartHelper.service.changeWTPartMasterIdentity(part.getMaster(), name, number, localWTOrganization);
  ```

- **Want to change other thing also like Default Unit,Default Trace code**

  ```
  WTPartMaster master = part.getMaster(); <Use the
  setter methods to change values > <save the master
  using PersistenceHelper>
   PersistenceHelper.manager.save(master);
  ```

You can use WTPartHelper API for other uses also like if you want to get the associated described document there is a method named
WTPartHelper.service.getDescribedByDocuments(paramWTPart) It will return you QueryResult

Assign View using : WTPartHelper.service.assignView(paramWTPart, paramString)

8.9 To add primary or secondary attachments (a file) to any document :

```
ContentHolder contentHolder = ContentHelper.service.getContents(localWTDocument);
ApplicationData applicationdata = ApplicationData.newApplicationData(contentHolder);
applicationdata.setRole(ContentRoleType.PRIMARY);
applicationdata.setFileName("DocumentDetails");
ContentServerHelper.service.updateContent(contentHolder, applicationdata, "D:\\NewPDF.pdf");
PersistenceHelper.manager.save(applicationdata);
```

8.10 To check in and check out any Object :

```
workingCopy = (WTPart)WorkInProgressHelper.service.checkout(part, WorkInProgressHelper.service.getCheckoutFolder(),
null).getWorkingCopy();
WTPartDescribeLink link = WTPartDescribeLink.newWTPartDescribeLink(workingCopy, doc);
  Persistable per = PersistenceHelper.manager.save(link);
 WorkInProgressHelper.service.checkin(workingCopy,null);
```

Above code is Checking Out the "part" and after that before Checking it in it will create a WTPartDescribeLink between the "part" and the "doc".

If you want to retrieve a parts revision and iteration you can use

```
String str1 = parentPart.getVersionInfo().getIdentifier().getValue();//for revision
String str2 = parentPart.getIterationIdentifier().getValue();//for iteration
```

8.11 To Create partdoc relation between a WTDocument and WTPart :

PartDocHelper.service.createPartDocRelation(paramWTPart, paramWTDocument);

8.12 To check whether the object is of a specific subtype :

TypeIdentifierHelper.*getType* (my_persistable).toString ().equals ("WCTYPE|<Hard types name>|<Soft types name>")
"my_persistable" means those Objects which implements the "Persistable" interface ex.
TypeIdentifierHelper.*getType* (localWTDocument).toString ().equals ("WCTYPE|wt.doc.WTDocument|com.ITCINFOTECH.Agenda")
Above code will check whether the document is of type agenda or not

8.13 Standard code for querying database

```
QuerySpec qs = new QuerySpec(WTPart.class); WTPart pt  = null;
qs.appendWhere(new SearchCondition(WTPart.class,WTPart.NAME,SearchCondition.EQUAL,"KD"),new int[] { 0 });
QueryResult qr = PersistenceHelper.manager.find((StatementSpec)qs);
while(qr.hasMoreElements())
	{
		pt = (WTPart)qr.nextElement();
	}
```

8.14 To retrieve 1st level BOM Structure :

```
QueryResult usesLinks = StructHelper.service.navigateUses(parentPart, false);
WTPartUsageLink wtu = null;
	while(usesLinks.hasMoreElements()){
				wtu = (WTPartUsageLink) usesLinks.nextElement();
				System.out.println(((WTPartMaster)wtu.getRoleBObject()).getName());//print the child parts name
	}
```

```
 QueryResult queryResult= WTPartHelper.service.getUsesWTParts(parentPart, new
	LatestConfigSpec()); while(queryResult.hasMoreElements())
	{
			WTPart part=null;
			Persistable[] persistable=(Persistable[])queryResult.nextElement();
			part=(WTPart)persistable[1];
			System.out.println(part.getName() + "    Child part");
			WTPartUsageLink partLink=(WTPartUsageLink)persistable[0];
	}
```

```
public static void getChildParts(WTPart[] parentParts) throws WTException //use this method if you have to find for WTPart Array
	{
		WTArrayList paramWTList = new WTArrayList();
		for(WTPart part : parentParts)
			paramWTList.add(part);
		Persistable[][][] queryResult= WTPartHelper.service.getUsesWTParts(paramWTList, new LatestConfigSpec());
		for(int i=0 ; i < queryResult.length; i++)
			{
				System.out.println("Parent part :-" + parentParts[i].getName());
				for(int j=0;j<queryResult[i].length;j++)
				{
					WTPart part = (WTPart)queryResult[i][j][1];
					//if you need WTPartUsageLink object then change 1 to 0
					System.out.println("Child Part :- " + part.getName());
				}
			}
  }
```

```
public static void getChildParts(WTPart parentPart) throws WTException
	{
		QueryResult queryResult= PersistenceHelper.manager.navigate(parentPart, WTPartUsageLink.USES_ROLE,
WTPartUsageLink.class);
		System.out.println(queryResult.size());
	while(queryResult.hasMoreElements())
	{
				Object obj = queryResult.nextElement();
				System.out.println(obj);
		WTPart part=null;
		WTPartMaster wpm=(WTPartMaster)obj;
		System.out.println("Child part name" + wpm.getName());
	}
}
```

8.15 Retrieve Multi –level bom Structure :

```
public static void getChildParts(WTPart parentPart) throws WTException
     {
          @SuppressWarnings("deprecation")
          QueryResult queryResult= WTPartHelper.service.getUsesWTParts(parentPart, new
LatestConfigSpec());
        while(queryResult.hasMoreElements())
        {
                        WTPart part=null;
                        Persistable[] persistable=(Persistable[])queryResult.nextElement();
                        part=(WTPart)persistable[1];
                        WTPartUsageLink partLink=(WTPartUsageLink)persistable[0];
               System.out.println("ParentPart :-" + parentPart.getName() + " ChildPart :- "+ part.getName());
                        getChildParts(part);
        }
     }
```

Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM

```
public static void getBOM(WTPart parentParts) throws WTException
      {
            WTArrayList paramWTList = new WTArrayList();
            paramWTList.add(parentParts);
            Persistable[][][] queryResult= WTPartHelper.service.getUsesWTParts(paramWTList, new LatestConfigSpec());
            for(int i=0 ; i < queryResult.length; i++)
                  {
                        for(int j=0;j<queryResult[i].length;j++)
                        {
                              WTPart part = (WTPart)queryResult[i][j][1];
                              //if you need WTPartUsageLink object then change 1 to 0
                              try{
                                    System.out.println("Parent part :-" + parentParts.getName());
                                    System.out.println("Child Part :- " + part.getName());
                                    System.out.println();
                                    getBOM(part);
                              }
                              catch(NullPointerException e)
                              {
                                    continue;
                              }
                        }
                  }
         }
```

More about QuerySpec :-

8.16 To Query about EPMDocument in a particular product

```
EPMDocument ed = null;
Persistable[] pt =null;
QuerySpec queryspec = new QuerySpec(EPMDocument.class);
 queryspec.setAdvancedQueryEnabled(true);
 int contIndex = queryspec.appendClassList(PDMLinkProduct.class,false);
                    SearchCondition sc1 = new SearchCondition(EPMDocument.class, "containerReference.key.id"
,PDMLinkProduct.class
                    ,       wt.util.WTAttributeNameIfc.ID_NAME);
                    queryspec.appendWhere(sc1, new int[] { 0, contIndex });
                    queryspec.appendAnd();
                    SearchCondition scSWPart = new SearchCondition(PDMLinkProduct.class,
                    "containerInfo.name",
                    SearchCondition.EQUAL,"Drive System",true);
                    queryspec.appendWhere(scSWPart, new int[] { contIndex });
                    QueryResult queryResult = PersistenceHelper.manager.find((StatementSpec)queryspec);
                    while(queryResult.hasMoreElements())
                        {
                     pt = (Persistable[])queryResult.nextElement();
                     for(Persistable p:pt)
                     {
                           ed = (EPMDocument)p;
                     }
 }
```

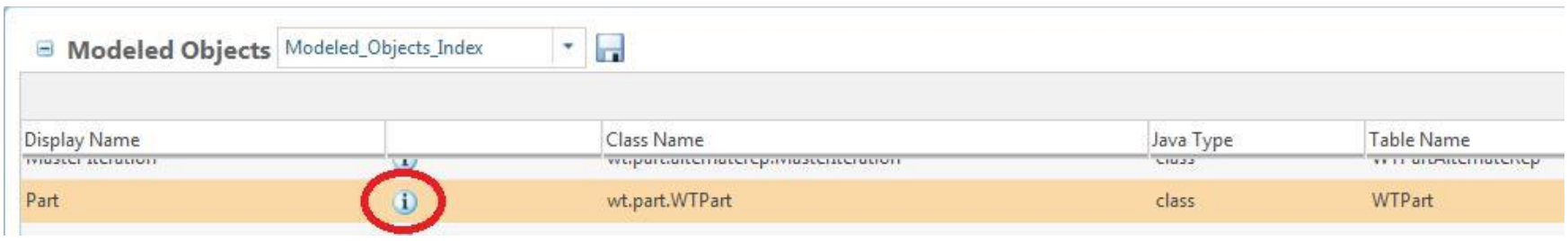
8.17 To Query for subtype of Part :- (Here I use Electrical Part as an Example)

```
QuerySpec queryspec = new QuerySpec(WTPart.class);
                queryspec.setAdvancedQueryEnabled(true);
                int typeIndex = queryspec.appendClassList(WTTypeDefinition.class,false);
                SearchCondition sc1 = new SearchCondition(WTPart.class,
"typeDefinitionReference.key.id",WTTypeDefinition.class, "thePersistInfo.theObjectIdentifier.id");
                queryspec.appendWhere(sc1, new int[] { 0, typeIndex }); queryspec.appendAnd();
                  queryspec.appendOpenParen();
                  SearchCondition scSWPart = new SearchCondition(WTTypeDefinition.class ,
"logicalIdentifier",SearchCondition.EQUAL,"com.ptc.ElectricalPart",true);
                  queryspec.appendWhere(scSWPart, new int[] { typeIndex });
                  queryspec.appendCloseParen();
                 QueryResult queryResult =
PersistenceHelper.manager.find((StatementSpec)queryspec);
                System.out.println("Size :- " + queryResult.size());
                Persistable[] pt =null;
                while(queryResult.hasMoreElements())
                     {
                  pt = (Persistable[])queryResult.nextElement();
                  for(Persistable p:pt)
                  {
                        WTPart part = (WTPart)p;
                        System.out.println(part.getName());
                  }
}
```

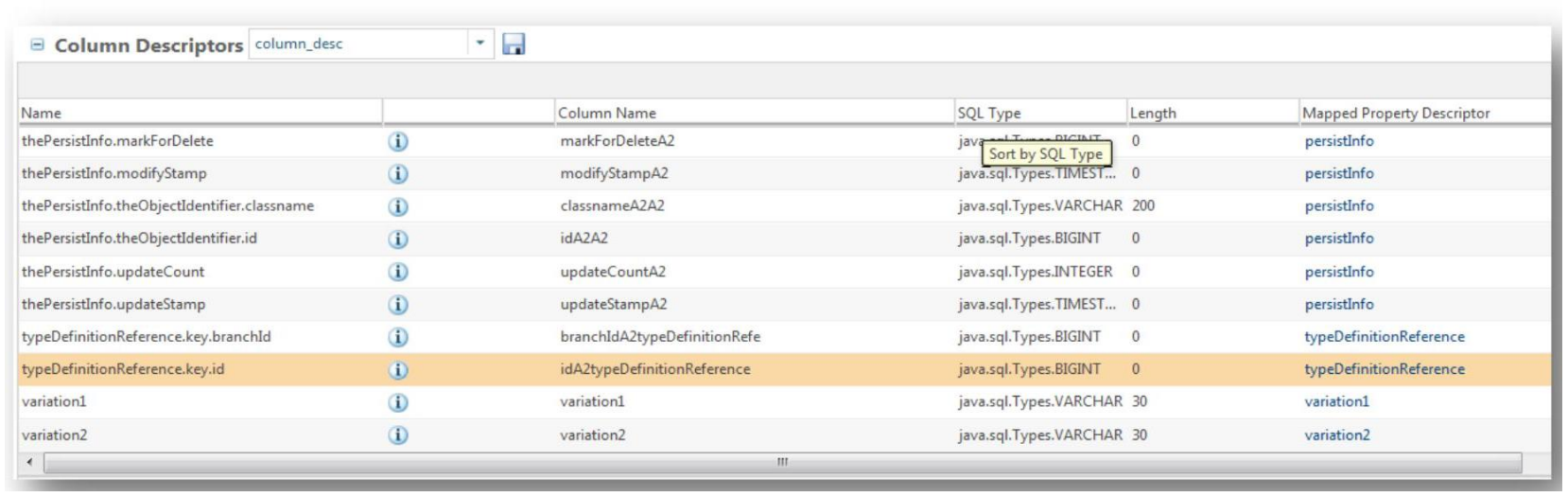
Note :

To find the subtype we have to join two tables one is WTPart and another is WTTypeDefinition.After that we have to identify the foreign key in WTPart and the Primary Key in WTTypeDefinition.Usually all table's primary key is the "IDA2A2"column and in this case the column name in WTPart is "IDA2TYPEDEFINITIONREFERENCE".

But to use QuerySpec we need the column descriptor of corresponding column name. To determine the column descriptor Go to. Customization → Tools → Modeled Object and Search for wt.part.WTPart. Your search result will be looks like this.



Click on the Red circled button to view the details page there you will find the column descriptor table.



Using the same way you can find the column descriptor of "IDA2A2" in WTTypeDefinition table.

To import workflows from 9.1 to 10.X you should change the following property.

<Windchill>\codebase\wt.properties file.

wt.ixb.import.allowCrossReleaseImport=true

8.18 Query to find Latest Document which are Released along with the Released Date

Oracle query for the requirement.

```
select
versionIdA2versionInfo,iterationIdA2iterationInfo,wtdocumentnumber,wtdocument.statestate,wtdocument.ida2a2,lifec
yclehistory.modifystampa2
,lifecyclehistory.state
from wtdocument,wtdocumentmaster,objecthistory,lifecyclehistory where wtdocument.ida3masterreference =
wtdocumentmaster.ida2a2
and wtdocument.statestate='RELEASED' and objecthistory.ida3a5= wtdocument.ida2a2 and objecthistory.ida3b5 =
lifecyclehistory.ida2a2
and lifecyclehistory.action = 'Enter_Phase' and lifecyclehistory.state='RELEASED'
and wtdocument.latestiterationinfo = 1 and wtdocument.branchIditerationInfo in
(select max(A1.branchIditerationInfo) from wtdocument A1, wtdocumentmaster A2 where A1.ida3masterreference =
A2.ida2a2 group by A2.WTDOCUMENTNUMBER);
```

The corresponding QuerySpec code can be found inside ext package inside AdvancedQuerySpecTest class.

8.19 Read Global Enumeration Internal / Display name value

```
public static void getGlobalEnumeration() throws WTException {// Provide
                                                                // attribute
                                                                // name
        AttributeTypeIdentifier attTypeId = TypedUtility.getAttributeTypeIdentifier("CUSTOMER",
                TypedUtility.getTypeIdentifier("wt.part.WTPart"));
        System.out.println("attTypeId :: " + attTypeId);
        AttributeDefinitionReadView attDefReadView =
TypeDefinitionServiceHelper.service.getAttributeDefView(attTypeId);
        Collection<ConstraintDefinitionReadView> colConstrDefRV = attDefReadView.getAllConstraints();
        System.out.println("colConstrDefRV :: " + colConstrDefRV.size());
        ConstraintDefinitionReadView cdrv = null;
        for (ConstraintDefinitionReadView cdrvIt : colConstrDefRV) {
            System.out.println("reading contraint");
            if ((null != cdrvIt.getEnumDef()) && (!(cdrvIt.isDisabled()))) {
                cdrv = cdrvIt;
                Collection<ConstraintDefinitionReadView> col = new
HashSet<ConstraintDefinitionReadView>();
                col.add(cdrv);
                System.out.println("is active enum :: " + EnumerationConstraintHelper.isActiveEnum(cdrv));
                List<ConstraintDefinitionReadView> list =
EnumerationConstraintHelper.getEnumeratedConstraints(col,
                        true);
                for (ConstraintDefinitionReadView rdv : list) {
                    Set<ConstraintDefinitionReadView> allConditions = rdv.getAllConditions();
                    for (ConstraintDefinitionReadView condition : allConditions) {
                        System.out.println(condition.getAttName());
                        System.out.println(condition.getRuleData());
                    }
                    Map<String, EnumerationEntryReadView> map =
rdv.getEnumDef().getAllEnumerationEntries();
                    for (Entry<String, EnumerationEntryReadView> entry : map.entrySet()) {
                        System.out.println(entry.getKey());

    System.out.println(entry.getValue().getPropertyValueByName("displayName").getValue());
                    }
                }
            }
        }
```

8.20 To create a Project :

```
QuerySpec qs = new QuerySpec(WTOrganization.class);
                    WTOrganization localWTOrganization = null;
                    qs.appendWhere(new
SearchCondition(WTOrganization.class,WTOrganization.NAME,SearchCondition.EQUAL,"Demo Organization"),new int[] {
                    0 });
                    QueryResult qr = PersistenceHelper.manager.find((StatementSpec)qs);
                    System.out.println("Size" + qr.size());
                    while(qr.hasMoreElements()){
                            localWTOrganization = (WTOrganization)qr.nextElement();
                    }
                     OrgContainer localOrgContainer = null;
                    Project2 localProject2 = Project2.newProject2();
                    localProject2.setName("MyProject");
                    localOrgContainer = WTContainerHelper.service.getOrgContainer(localWTOrganization);
                    System.out.println(localOrgContainer.getContainerName());
                    localProject2.setContainer(localOrgContainer);
                    WTContainerHelper.setPrivateAccess(localProject2, false);
                    WTContainerRef localWTContainerRef = WTContainerRef.newWTContainerRef(localOrgContainer);
                    QueryResult localQueryResult =
ContainerTemplateHelper.service.getEnabledTemplateMasters(localWTContainerRef, Project2.class);
                    Enumeration localEnumeration = localQueryResult.getEnumeration();
                    WTContainerTemplateMaster localWTContainerTemplateMaster = null;
                    do {
                            if (!(localEnumeration.hasMoreElements())) break;
                           localWTContainerTemplateMaster = (WTContainerTemplateMaster)localEnumeration.nextElement(); }while
                      (!("General".equals(localWTContainerTemplateMaster.getName())));
                    WTContainerTemplate localWTContainerTemplate = null;
                    localWTContainerTemplate =
ContainerTemplateHelper.service.getContainerTemplate(localWTContainerTemplateMaster); if
                    (localWTContainerTemplate != null)
                        localProject2.setContainerTemplate(localWTContainerTemplate);
                    localProject2 = (Project2)WTContainerHelper.service.create(localProject2);
                    //to start the project
                    NmOid nmoid = new NmOid(localProject2);
                    NmProjectHelper.service.start(nmoid);
For more details decompile the class "LoadProject2"
```

8.21 To Create PartUsageLink between two part:

```
WTPart workingCopy;
                    WTPart part = WTPart.newWTPart("myPartNumber", "MyPartName");
                    //part.setContainer(nmCommandBean.getContainer());
                    PersistenceHelper.manager.save(part);
                    WTPart childPart = WTPart.newWTPart("myChildPartNumber", "MyChildPartName");
                    //childPart.setContainer(nmCommandBean.getContainer());
                    PersistenceHelper.manager.save(childPart);
                    workingCopy = (WTPart)WorkInProgressHelper.service.checkout(part,
WorkInProgressHelper.service.getCheckoutFolder(), null).getWorkingCopy();

                    WTPartUsageLink link = WTPartUsageLink.newWTPartUsageLink(workingCopy, (WTPartMaster)
childPart.getMaster());

                    Persistable per2 = PersistenceHelper.manager.save(link);
                    WorkInProgressHelper.service.checkin(workingCopy,null);
```

8.22 To Change the LifeCycle state programmatically:

LifeCycleHelper.*service*.setLifeCycleState((LifeCycleManaged)<object implementing LifeCycleManaged interface like WTPart,WTDocument etc.>,State.*toState*(<Key of the LifeCycle state>));

e.g :-

LifeCycleHelper.*service*.setLifeCycleState((LifeCycleManaged)wtPart,State.*toState*("CANCELLED"));

8.23 To get the deliverables from Activity (Project Link):

QueryResult qr = PersistenceHelper.manager.navigate(planActivity,
PlanDeliverableLink.PLAN_DELIVERABLE_ROLE, PlanDeliverableLink.class);

8.24 How to set the value of a Date and Time attribute via API :-

```
ReferenceFactory rf = new ReferenceFactory();
WTReference ref = rf.getReference("VR:wt.part.WTPart:162047"); Persistable p = ref.getObject();
p = WorkInProgressHelper.service.checkout( (Workable) p,
WorkInProgressHelper.service.getCheckoutFolder(),"").getWorkingCopy();
LWCNormalizedObject lwc = new LWCNormalizedObject(p, null,Locale.ENGLISH,new
UpdateOperationIdentifier()); lwc.load("time"); SimpleDateFormat dateformat = new
SimpleDateFormat();
```

```
try {
        Date date1 = dateformat.parse("12/20/13 12:00 PM");
        Date date2 = dateformat.parse("1/20/14 1:15 PM");
        Timestamp[] dates = {new Timestamp(date1.getTime()),new Timestamp(date2.getTime())};
        lwc.set("time", dates);
        lwc.apply();
        PersistenceHelper.manager.modify(p);
        WorkInProgressHelper.service.checkin((Workable) p,"");
} catch (Exception e) {
e.printStackTrace();
}
```

8.25 How to retrieve and add a user or role to the Team of a TeamManaged object :

To retrieve the team (cast the return value to wt.team.WTRoleHolder2)

wt.team.TeamHelper.service.getTeam(wt.team.TeamManaged object)

To add a user or role to the team

wt.team.TeamHelper.service.addRolePrincipalMap(wt.project.Role role, wt.org.WTPrincipal principal, wt.team.WTRoleHolder2 team)

8.26 To create different kind of link between WTPart and CAD Document :

EPMBuildRule epm = EPMBuildRule.*newEPMBuildRule*((EPMDocument)obj, mpart);
epm.setBuildType(EPMBuildRule.*BUILD_ATTRIBUTES*);
PersistenceHelper.*manager*.save(epm);
To create Contributing link use build type as "EPMBuildRule.*BUILD_ATTRIBUTES*"

8.27 To change the name and number of CAD Document :-

```
EPMDocumentMaster epmMaster=(EPMDocumentMaster) epmDoc.getMaster();
EPMDocumentMasterIdentity epmMasterId =(EPMDocumentMasterIdentity) epmMaster.getIdentificationObject();
epmMasterId.setName("Equi_bar_45321.ASM");
IdentityHelper.service.changeIdentity(epmMaster,epmMasterId);
```

8.28 Undo Checkout

```
/**
   * Undo checkout an already checkout object
   * @param holder
   * @return Workable
   * @throws WTException
   */
  public static Workable undoCheckout (Workable holder) throws WTException {
    if(WorkInProgressHelper.isCheckedOut((Workable)holder)){
        try {
            holder = WorkInProgressHelper.service.undoCheckout(holder); }catch
        (WTPropertyVetoException e1){
            throw new WTException (e1);
        }
    }
   return holder;
  }
```

To get the Display Name for the Global Enumeration entry in Windchill PDMLink

Refer the PTC case section.

To filter types from the type picker in an object creation wizard in Windchill PDMLink

Refer the PTC case section.

8.29 To programmatically fetch all the Dependents of an EPMDocument in Windchill PDMLink

- **CADCollector** API can be used to fetch dependents and related objects in various configurations:
  - Dependents such as : assembly members, external references ...
  - Related drawings
  - Family instances and generics
  - Associated parts
- See **com.ptc.windchill.api.cad.CadCollector** in the JavaDoc for additional information.
- The CadCollector is a server side API.
- Snippet code :

```
/* Given a seed CAD Document, collects the related CAD objects in configuration AsStored */

ObjectIdentifier oid = ObjectIdentifier.newObjectIdentifier("wt.epm.EPMDocument:260910");
EPMDocument top = (EPMDocument) PersistenceHelper.manager.refresh(oid);

WTCollection seeds = new WTArrayList();
seeds.add(top);

EPMDocConfigSpec configSpec = new EPMDocConfigSpec();

EPMAsStoredConfigSpec asStored = EPMAsStoredConfigSpec.newEPMAsStoredConfigSpec(top);
configSpec.setAsStoredConfig(asStored);
configSpec.setAsStoredActive(true);
NavigationCriteria nc = NavigationCriteria.newNavigationCriteria();

List configpecs = new ArrayList();
configpecs.add(asStored);
nc.setConfigSpecs(configpecs);

CadCollectedResult ccr = CadCollector.newInstance(seeds, nc) .dependents(GatherDependents.ALL)
.familyGenerics(GatherFamilyGenerics.ALL) .familyMembers(GatherFamilyMembers.ALL)
.drawings(CadCollector.GatherDrawings.ALL) .collect();
Collection result = ccr.getCollectedObjects();
System.out.println("Number of objects Collected : " + result.size());
Iterator items = result.iterator();
while (items.hasNext())
{
   Object item = items.next();
   if (item instanceof Persistable)
        System.out.println("Related CAD object : " + (Persistable)item);
}
```

For further details visit: - https://www.ptc.com/appserver/cs/view/solution.jsp?n=CS81089

8.30 To create a Project picker using tags :-

```
<wctags:itemPicker id="ProjectId"
                componentId="ProjectId"
                objectType="wt.projmgmt.admin.Project2"
                inline="false"
                readOnlyPickerTextBox="true"
                showTypePicker="false"
                label="*Project Number"
                        displayAttribute="projectNumber"
                />
```

8.31 To get the PlanDeliverable associated with a WTPart :-

```
PlannableHelper.service.getDeliverables(part);//part is the object of WTPart
```

8.32 To read the value of an IBA of Occurrence Link :

```
private String getIbaValue(WTPartUsageLink paramWTPartUsageLink) {
        String str = "";
        if (paramWTPartUsageLink == null) {
                return "";
        }
        @SuppressWarnings("rawtypes")
        Vector localVector =
        paramWTPartUsageLink.getUsesOccurrenceVector(); if (localVector
        == null) {
                System.out.println("localvector is null" + localVector.size());
                return "";
        }
        int i = localVector.size();

        for (int j = 0; j < i; ++j) {
                Occurrence occur = (Occurrence) localVector.get(j);
                try {
                        Vector<Class<UsesOccurrenceUserIBAs>> classes = new
                                        Vector<Class<UsesOccurrenceUserIBAs>>( 1);
                        classes.add(UsesOccurrenceUserIBAs.class);
                        wt.fc.QueryResult result = wt.occurrence.OccurrenceHelper.service
                                        .getUsesOccurrenceData((UsesOccurrence) occur, classes);
                        //              System.out.println("size of result " +
                        result.size()); if (result.size() > 0) {
                                System.out.println("inside if");
                                wt.occurrence.UsesOccurrenceData data = (wt.occurrence.UsesOccurrenceData) result
                                                .nextElement();
                                if (data instanceof wt.occurrence.UsesOccurrenceUserIBAs) {
                                        LWCNormalizedObject obj = new LWCNormalizedObject(data,
                                                        null, null, null);
                                        obj.load("partno");// part no is the name of the IBA
                                        Object attributeValue = obj.get("partno");
                                        str += (String) attributeValue + ",";
                                        System.out.println("value " + str);
                                }
                        }
                } catch (WTException e) {
                        e.printStackTrace();
                }

        }

        if (!(str.equalsIgnoreCase("")))
                return ((String) str.substring(0, str.length() - 1));
        else
                return str;
        }
```

8.33 Programatically giving check-in comment :

```
VersionControlHelper.setNote(doc, "Comment");
PersistenceServerHelper.update(doc);
```

8.34 To get the current user (Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM):

```
private static WTUser getCurrentUser()
{
		WTUser wtuser = null;
		wt.org.WTPrincipal wtprincipal = null;
		try {
			wtprincipal = SessionHelper.getPrincipal();
		} catch (WTException e) {
			e.printStackTrace();
		}
		// get the current user
		wtuser = (WTUser) wtprincipal;
		return wtuser;
}
```

8.35 Without iteration updating multi-valued attributes value :- (vineet.dalal@itcinfotech.com)

```
private static void updateIBAs(Persistable part, String attrName,
		String[] attrValue) throws WTException, RemoteException,
		WTPropertyVetoException {
	IBAHolder ibaHolder = IBAValueHelper.service.refreshAttributeContainer(
			(IBAHolder) part, null, null, null);
	DefaultAttributeContainer defAttrContainer = (DefaultAttributeContainer) ibaHolder
			.getAttributeContainer();
	if (defAttrContainer != null) {
		AttributeDefDefaultView attrDef = IBADefinitionHelper.service
				.getAttributeDefDefaultViewByPath(attrName);
		AbstractValueView abstractview[] = defAttrContainer
				.getAttributeValues(attrDef);
		for (int i = 0; i < abstractview.length; i++) {
			if (attrDef instanceof StringDefView) {
			defAttrContainer.deleteAttributeValue(abstractview[i]);
			}
		}
		if (attrValue != null) {
			for (int j = 0; j < attrValue.length; j++) {
				if (attrValue[j] != null) {
					if (attrValue[j].length() > 0) {
						AbstractValueView attrValueView = null;
						attrValueView = new StringValueDefaultView(
								(StringDefView) attrDef, attrValue[j]);
						defAttrContainer.addAttributeValue(attrValueView);
					}
				}
			}
		}
	}
	Manager dbm = ManagerServiceFactory.getDefault().getManager(
			IBAValueService.class);

	((MultiObjIBAValueService) dbm).theStandardIBAValueService.theIBAValueDBService
			.updateAttributeContainer(ibaHolder, null, null, null);
}
```

8.36 From Part to object id :- (Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM)

String partobid=OidHelper.getOidAsString(part);

8.37 To get the user from any Role :- (Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM)

Here role menas workflow role not the container team role.

The following example explains how to get the user name(s) assigned to Product Manager from the team of a Change Request.

The code will even work for Custom Roles also.

```
public static String getQcAssigneeTeam(WTChangeRequest2 chanRequest) throws
TeamException, WTException, IOException {
        Team team=TeamHelper.service.getTeam((TeamManaged) chanRequest);
        /* Team refer to the workflow participant team.If you want context team then use
                container Team */
        Enumeration<?> members =
      team.getPrincipalTarget(Role.toRole("QC_Validator")); //QC_Validator
        StringBuffer qcAssinees = new StringBuffer();
        //int i = 1;
        while(members.hasMoreElements())
        {
          WTPrincipalReference ref=(WTPrincipalReference) members.nextElement();

          qcAssinees=qcAssinees.append( ( (WTUser)ref.getPrincipal() ).getFullName()
                        ); qcAssinees.append("\n");
        }
        return qcAssinees.toString();

      }
     private static String getPoductManager(WTChangeRequest2 changeRequest)throws
TeamException, WTException, IOException {
                Team team=TeamHelper.service.getTeam((TeamManaged) changeRequest);
        Enumeration<?> prodManagers = team.getPrincipalTarget(Role.toRole("PRODUCT
        MANAGER"));
        StringBuffer prodManager = new StringBuffer();
        while(prodManagers.hasMoreElements())
        {
          WTPrincipalReference ref=(WTPrincipalReference) prodManagers.nextElement();
          prodManager=prodManager.append(( (WTUser)ref.getPrincipal() ).getFullName()
          );
        }
        return prodManager.toString();
        }
```

8.38 Get describe by document and reference document linked to a WTPart :- (*Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM*)

```
QueryResult qr = WTPartHelper.service.getReferencesWTDocumentMasters(parentPart);
                    while (qr.hasMoreElements()) {
                                        WTDocumentMaster localWTDocument = (WTDocumentMaster)
qr.nextElement();
                    }


QueryResult qr = WTPartHelper.service.getDescribedByDocuments(parentPart); while
                                    (qr.hasMoreElements()) {
                                WTDocument localWTDocument = (WTDocument) qr.nextElement();
                    }
```

8.39 Read value of WTProperties through code :- (Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM)

```
        /**
                * Taking the key as a parameter this method will return the corresponding.
                * value from WT.properties
                *
                * @param key
                *          key of the property
*
                * @return the corresponding value
*
                * @throws IOException
                *           throws {@link IOException}
                */
               public static String readWTProperties(String key) throws IOException
                     { return WTProperties.getLocalProperties().getProperty(key);
               }
```

8.40 Update IBA without iterating the Object :

```
void updateIba(WTPart pt, String var, String value) throws WTException,
                RemoteException, WTPropertyVetoException {
    wt.iba.value.IBAHolder ibaHolder = IBAValueHelper.service
                .refreshAttributeContainer((wt.iba.value.IBAHolder) pt, null,
                        null, null);
    System.out.println("IBAHOLDER" + ibaHolder);
    DefaultAttributeContainer defaultattributecontainer = (DefaultAttributeContainer) ibaHolder
                .getAttributeContainer();

    System.out.println("defaultattributecontainer"
                + defaultattributecontainer);

    AttributeDefDefaultView addv = IBADefinitionHelper.service
                .getAttributeDefDefaultViewByPath(var);// it should be logical
                                                                    // identifier value
    System.out.println("iba name" + addv.getDisplayName());
    AbstractValueView[] abstractvalueview =
    defaultattributecontainer
                .getAttributeValues(addv);
    for (int k = 0; k < abstractvalueview.length; k++) {

        ((StringValueDefaultView)
        abstractvalueview[k]).setValue(value); defaultattributecontainer
                        .updateAttributeValue(abstractvalueview[k]);
    }
    StandardIBAValueService.theIBAValueDBService.updateAttributeContainer
                ( ibaHolder, null, null, null);
}
```

8.41 Start Workflow through code :- (Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM)

```
startWorkflow("Part_Catalogue_Report_Download_Workflow",paramNmCommandBean.getContainer().getConta
inerReference(), part);

public static void startWorkflow(String workflowName,WTContainerRef wtcontainerref, WTPart part) {
		//		deaclare wfprocessdefinition
		WfProcessDefinition wfprocessdefinition;
		//		deaclare wfprocesstemplate
		WfProcessTemplate wfprocesstemplate = null; WfProcess wfprocess = null;
		try {
			// get wf defination
			wfprocessdefinition = WfDefinerHelper.service.getProcessDefinition(workflowName,
wtcontainerref);
			// get wf template
			wfprocesstemplate = wfprocessdefinition.getProcessTemplate(); //
			get wf process
			wfprocess = WfEngineHelper.service.createProcess(wfprocesstemplate,null, wtcontainerref); }
		catch (WTException e) {
			logger.info("" + e);
		}

		// get timestamp
		Calendar calendar = Calendar.getInstance(TimeZone.getTimeZone("GMT")); //
		get timestamp
		Timestamp timestamp = new Timestamp(calendar.getTimeInMillis());
		String s3 = null;
		if (wfprocesstemplate != null) {
			s3 = (new
StringBuilder()).append(wfprocesstemplate.getName()).append("_").append(getCurrentUser().getName()).app
end("_").append(timestamp.toString()).toString();
		}
		//		declare processdata
		ProcessData processdata = null; if
		(wfprocess != null) {
			//		set wfprocess name
			wfprocess.setName(s3);
			//		set wfprocess desc
			wfprocess.setDescription(wfprocesstemplate.getIdentity());
			// set wfprocess priority
			wfprocess.setPriority(1L);
			processdata = wfprocess.getContext();
			try {
				//		set pbo
				processdata.setValue("primaryBusinessObject", part);
			} catch (InvalidDataException e) {
				logger.info("" + e);
			}

			//		processdata.setValue("creator",
			getCurrentUser()); try {
				// get the session manager
				SessionHelper.manager.setAdministrator();
			} catch (WTException e) {
				logger.info("" + e);
			}
			try {
				//		start the workflow
				WfEngineHelper.service.startProcessImmediate(wfprocess,processdata,
wfprocess.getPriority());
			} catch (WTException e) {
				logger.info("" + e);
			}

		}
	}
```

8.42 Code to read custom property file :- (Code Courtesy :- Rahul.Jadhav@ITCINFOTECH.COM)

```
public final class CustomPropertyHelper {

	/**
	 * Private variable to store the path to gear.properties.
	 */
	private static Properties customProp = new Properties();

	/**
	 * Default comment for field localProp.
	 */
	private static WTProperties localProp;

	/**
	 * static initialisation block to initialize the static variables.
	 */
	static {
		try {
			/*
			 * Storing the WTProperties.
			 */
			localProp = WTProperties.getLocalProperties();

			String dirSep = readWTProperties("dir.sep");

			customProp.load(WTContext.getContext().getResourceAsStream(
					("ext") + (dirSep) + ("custom") + (dirSep)
							+ ("custom.properties")));
			//path to the properties file.
		} catch (IOException ex) {
			ex.printStackTrace();
		}
}

/**
 * private constructor for Sonar.
 */
private CustomPropertyHelper() {
	// TODO Auto-generated constructor stub
}
/**
 * Taking the key as a parameter this method will return the corresponding.
 * value from WT.properties.
 *
 * @param key
 *            key of the property
 *
 * @return the corresponding value
 *
 * @throws IOException
 *             throws {@link IOException}
 */
public static String readWTProperties(String key) throws IOException {
	return localProp.getProperty(key);
}

/**
 * Read the value for the given key.
 *
 * @param key
 *            the value of the key for properties.
 *
 * @return the value for the key.
 */
	public static String readCustomProperties(String key) {
		/*
		 * The value of corresponding KEY.
		 */
		return customProp.getProperty(key);
	}
}
```

8.43 Get Lifecycle History :- (Code courtesy :- Nitin.darekar@itcinfotech.com)

```
public static void getHistoryInfo(WTPart parentPart) {
		QueryResult qrParent;
	try {
	qrParent = LifeCycleHelper.service.getHistory(parentPart);
	System.out.println(qrParent.size() + " Get Info ::: " +qrParent.hasMoreElements());
	while (qrParent.hasMoreElements()) {
	LifeCycleHistory lifeCycleHistoryObj = (LifeCycleHistory)qrParent.nextElement();
	System.out.println("lifeCycleHistoryObj " + lifeCycleHistoryObj.getState());
			}
		} catch (LifeCycleException e | WTException e) {
			e.printStackTrace();
		}
	}
```

8.44 API to get Recently Accessed objects list for current user

```
java.util.Vector recentObjs= wt.recent.RecentlyVisitedHelper.service.getRecentlyVisitedObjectStack();

for (Iterator iterator = recentObjs.iterator(); iterator.hasNext();) {

    Object object = (Object) iterator.next();

    System.out.println("Recently Accessed Object -- "+((wt.recent.ObjectVisitedInfo)object).getName());

}
```

8.45 Complete activity of WorkFlow :

```
public static String closeActivity(Object primaryBusinessObject,
                    wt.workflow.engine.WfProcess wfprocess, String activityName) throws
                    WTException {
             try {

                          wt.query.QuerySpec qspec = new wt.query.QuerySpec(
                                       wt.workflow.engine.WfActivity.class);
                    qspec.appendWhere(new wt.query.SearchCondition(
                                 wt.workflow.engine.WfActivity.class,
                    "parentProcessRef.key", "=", wt.fc.PersistenceHelper
                                 .getObjectIdentifier((wt.fc.Persistable) wfprocess)));
wt.fc.QueryResult qr1 = wt.fc.PersistenceHelper.manager.find(qspec);

while (qr1.hasMoreElements()) {
       wt.workflow.engine.WfActivity activity = (wt.workflow.engine.WfActivity) qr1
                    .nextElement();
       if (activity.getName().trim().equalsIgnoreCase(activityName)) {
             wt.fc.QueryResult qr2 = wt.fc.PersistenceHelper.manager
                          .navigate(
                                       activity,
                                       "assignment",
                                       wt.workflow.work.ActivityAssignmentLink.class,
                                       true);
             while (qr2.hasMoreElements()) {
                    java.util.Vector vector = new java.util.Vector(); wt.workflow.work.WfAssignedActivity
                    wfassignedactivity1 = null; wt.workflow.work.WfAssignment assignment =
                    (wt.workflow.work.WfAssignment) qr2
                                 .nextElement();
                    wt.fc.QueryResult qr3 = wt.fc.PersistenceHelper.manager
                                 .navigate(assignment, "workItem",
                                              wt.workflow.work.WorkItemLink.class,
                                              true);
                    while (qr3.hasMoreElements()) {
                          wt.workflow.work.WorkItem workitem = (wt.workflow.work.WorkItem) qr3
                                       .nextElement();
                          if (workitem.getStatus().toString()
                                       .equals("POTENTIAL")
                                       || workitem.getStatus().toString()
                                                    .equals("ACCEPTED")) {
                                 wfassignedactivity1 = (wt.workflow.work.WfAssignedActivity) workitem
                                              .getSource().getObject();
                                 wt.fc.PersistenceHelper.manager
                                              .delete(workitem);
                                 vector = wfassignedactivity1.getAllEvents();
                          }
                          if (wfassignedactivity1 != null) {
                                 wt.workflow.engine.WfEngineHelper.service
                                              .complete(wfassignedactivity1, vector);
                          }
                    }
             }
       }

       return "SUCCESS";
} catch (WTException wex) {
       System.out.println("4. Error in retrieving the object because "
                    + wex.getLocalizedMessage());
       return "ERROR";
}
}
```

8.46 To increase the search limit of 50 for Find Participants :

- Change the letter n to the desired maximum value and run following command in a Windchill shell: **xconfmanager -s com.ptc.netmarkets.userSearch.maxCount=n -t codebase\wt.properties -p**
- Restart Windchill

9. SONRA RULES

As we all know that we need to standardize our code as per sonar guideline.
So, it may be helpful that at the time of writing the code if we can write it as per the guideline rather than change/restructure (if required) later.
Rule violation can be categorize into four types Critical, Major, Minor , Info.(There is another type called Blocker but it's occur rarely)
If your code is free of Critical, Major, Minor violation then your Rule Compliance will be the magical number **100**
Below are some violation related to sonar rules.

9.1 If Stmts Must Use Braces (Major)

Usually if there is only one statement after For , While , For Each, Do While loop or If statement we don't use braces.
Like the code below.

```
String name = "98300 - Document of Inspection";
String message;
if(name.matches("[0-9]+"))
    message = "Digit is there";
else
    message = "Digit is not there";

return (String.format("%02d", count) + "-" + number);
```

Though the code is perfectly fine still it will create two major violations in Sonar cause as per sonar guideline we must use braces even if there is only one statement.

After correction

```
String name = "98300 - Document of Inspection";
String message;
if (name.matches("[0-9]+")) {
    message = "Digit is there";
} else {
    message = "Digit is not there";
}
```

Though it seems a bit ridiculous but it increases Readability and reduces the chances of other violations in case of long file(s).

9.2 System Println (Major)

As the name suggest never use System. (out|err).print in your code which will be deployed in server.
Instead of SysOut use Logger.

9.3 Hide Utility Class Constructor (Major)

Make sure that utility classes (classes that contain only static methods) do not have a public constructor.
Apart from the private constructor declare the class as Final also.

9.4 Unused local variable (Major)

This major violation occurs when a local variable is declared and/or assigned, but not used.
Comment or delete the local variables which falls under above mentioned criteria.
At the time of declaring local variable you should also declare a variable as final if any local variable is assigned only once.

9.5 Broken Null Check (Critical)

The null check is broken since it will throw a Nullpointer itself. The reason is that a method is called on the object when it is null. It is likely that you used || instead of && or vice versa.

Look at the below code

```
List<String> list = new ArrayList<String>();
if (!(list.isEmpty()) && (list != null))

{
System.out.println(list.size());
}
```

The intension is to call the System.out.println() if the list is not null and the list is not empty.

But the above code will give you the critical violation because first you are checking whether the list is empty and then you are checking whether it's null or not.
If the list is null then in spite of your null check it will throw NullPointerException.

The correct statement will be

if ((list != null)) && !(list.isEmpty())
It's advisable to create a blank collection rather then set it to null.

9.6 Correctness - Possible null pointer dereference (Critical)

There is a branch of statement that, if executed, guarantees that a null value will be dereferenced, which would generate a NullPointerException when the code is executed. Of course, the problem might be that the branch or statement is

```
public static int cardinality(Object obj, final Collection col) {
int count = 0;
  if (col == null) {
    return count;

  }
  Iterator it = col.iterator();
  while (it.hasNext()) {
    Object elt = it.next();
    if ((null == obj && null == elt) || obj.equals(elt)) { // null pointer dereference
    count++;
}
```

infeasible and that the null pointer exception can't ever be executed

The above code will generate this critical violation as the second condition of if statement can generate NullPointerExeption.

```
public static int cardinality(Object obj, final Collection col) {
  int count = 0;
  if (col == null) {
    return count;
  }
  Iterator it = col.iterator();
  while (it.hasNext()) {
    Object elt = it.next();
    if ((null == obj && null == elt) ||
        (null != obj && obj.equals(elt))) {
      count++;
    }
  }
```

9.7 Loose coupling (Major)

Avoid using implementation types (i.e., HashSet); use the interface (i.e. Set) instead.
Create any collection object(s) by using the interface like.

List<String> list = new ArrayList<String>();

Set<Object> set = new HashSet<Object>();

9.8 Avoid Catching Generic Exception / Avoid Catching Throwable (Major / Critical)

Avoid catching generic exceptions such as NullPointerException, RuntimeException, Exception in try-catch block.
If you are throwing Exception or Throwable then you will get a critical error.
This is dangerous because it casts too wide a net; it can catch things like OutOfMemoryError.

9.9 Local Variable Name (Major)

All variable name should maintain a special format.
Like local variable name should start with small case irrespective of the access modifier.
String[] Number is wrong String[] number is right.
WTPart parentpart and WTPart parentPart both are right.
If the variable is static as well as final then the name should be all capital letter.

```
/**
Private variable for Logger.
 */
private static final Logger LOG = LogR
            .getLogger(CustomPROPLFormProcessor.class.getName());
```

9.10 Unused Private Field (Major)

Detects when a private field is declared and/or assigned a value, but not used.
The solution is similar to Unused Local Variable.

9.11 Avoid Duplicate Literals (Major)

Code containing duplicate String literals can usually be improved by declaring the String as a constant field.
Suppose you want to show user a pop up message "Please check the selected object".

Now based on five different condition you are showing the message by throwing WTException.
Instead of writing it like

```
if(condition1)
                        {
                                throw new WTException("Check the selected object");
                            }else if(condition2){
                                throw new WTException("Check the selected object");
                            }else if(condition3){
                                throw new WTException("Check the selected object");
                            }else if(condition4){
                                throw new WTException("Check the selected object");
                            }else if(condition5){
                                throw new WTException("Check the selected object");
                            }
```

Write it like

```
final String errorMessage= "Check the selected object";
                        if(condition1)
                        {
                                throw new WTException(errorMessage);
                            }else if(condition2){
                                throw new WTException(errorMessage);
                            }else if(condition3){
                                throw new WTException(errorMessage);
                            }else if(condition4){
                                throw new WTException(errorMessage);
                            }else if(condition5){
                                throw new WTException(errorMessage);
                            }
```

9.12 Performance - Unused field (Major)

The violation is similar to Unused Private Field.
The main difference is the access modifier is not private in this case.

9.13 Parameter Name (Major)

The same rule of Local Variable Name should follow in case of parameter also.
private void checkDigit(String FolderName) is wrong
private void checkDigit(String foldername) is right
private void checkDigit(String folderName) is right

9.14 Replace Vector With List (Major)

Consider replacing the Vector with the newer java.util.List
If any OOTB method is returning Vector you still can convert it in List by using **addAll** method.

9.15 Replace Hashtable With Map (Major)

Consider replacing the Hashtable with the newer java.util.Map.
If you are using OOTB loader APIs then you may don't have any choice.
But try to avoid HashTable whenever possible.

9.16 Parameter Assignment (Major)

Sonar disallow parameter assignments.

```
int increment(int salary)
{
    return salary + 1300000;
}
```

is right but

```
int increment(int salary)
{
    salary = salary + 1300000;
    return salary ;
}
```

is wrong.

9.17 Collapsible If Statements (Minor)

Sometimes two 'if' statements can be consolidated by separating their conditions with a boolean short-circuit operator.

```
int increment(int salary)
{
    if(salary < 100000)
    {
        if(salary > 500)
        {
            return salary * 2;
        }
    }
    return salary;
}
```

this can be simplified to

```
int increment(int salary)
{
    if(salary > 500 && salary < 100000)
    {
        return salary * 2;
    }
    return salary;
}
```

9.18 Method Length(Major)

This violation occurs when the method length exceeds **150** lines.

9.19 Ncss Method Count / Ncss Type Count (Major)

This rule uses the NCSS (Non Commenting Source Statements) algorithm to determine the number of lines of code for a given method. NCSS ignores comments, and counts actual statements. Using this algorithm, lines of code that are split are counted as one.
High Ncss Method Count violated the rule.
I am not sure about the max allowed limit as the violation description only shows the Ncss Method Count, not the limit.
Ncss Type Count is also same the only difference is instead of method it calculates no of lines in a class.

9.20 Naming - Suspicious constant field name (Major)

A field name is all in uppercase characters, which in Sun's Java naming conventions indicate a constant. However, the field is not final

```
private static Hashtable<Locale, EnumeratedType[]> LOCALSETS;
```

is wrong

```
private static final Hashtable<Locale, EnumeratedType[]> LOCALSETS = new Hashtable();
```

9.21 Visibility Modifier (Major)

Checks visibility of class members. Only static final members may be public; other class members must be private.
To access other class's private field(s) use setter method(s) if necessary.

9.22 Coupling - excessive imports (Major)

A high number of imports can indicate a high degree of coupling within an object. Rule counts the number of unique imports and reports a violation if the count is above the user defined threshold.
Not sure about the threshold value in the sonar configured by ITC Infotech as the value didn't show in the violation description but I suspect the value is near around 30.
You should break the class into smaller classes in order to avoid this rule violation.
Anyway you can easily fool the tool by the star import technique.

```
import java.text.ParseException;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Enumeration;
import java.util.HashMap;
import java.util.Iterator;
import java.util.Locale;
import java.util.Map;
import java.util.Set;
import java.util.Vector;
import java.util.Date;
import org.apache.logj.Logger;
import wt.change.WTChangeOrder;
import wt.doc.WTDocument;
import wt.epm.EPMDocument;
import wt.esi.ESITargetFacade;
import wt.esi.ESITransactionFacade;
import
wt.esi.doc.ESIDocumentsRendererInterface;
import
wt.esi.tgt.delegate.ESITargetProcessor;
import wt.facade.mpmlink.MPMLinkFacade;
import wt.fc.Persistable;
import wt.fc.PersistenceHelper;

import wt.identity.IdentityFactory;
import wt.inf.container.WTContained;
import wt.logj.LogR;
import wt.maturity.PromotionNotice;
import wt.org.OrganizationServicesHelper;
import wt.part.WTPart;
import wt.pom.PersistenceException;
import wt.pom.Transaction;
import wt.session.SessionContext;
import wt.session.SessionHelper;
import wt.session.SessionMgr;
import wt.session.SessionThread;
import wt.util.WTException;
import wt.util.WTPropertyVetoException;
```

```
import java.text.ParseException;
import java.util.*;
import org.apache.logj.Logger;
import wt.change.WTChangeOrder;
import wt.doc.WTDocument;
import wt.epm.EPMDocument;
import wt.esi.*;
import wt.facade.mpmlink.MPMLinkFacade;
import wt.fc.*;
import wt.identity.IdentityFactory;
import wt.inf.container.WTContained;
import wt.logj.LogR;
import wt.maturity.PromotionNotice;
import wt.org.OrganizationServicesHelper;
import wt.part.WTPart;
import wt.pom.PersistenceException;
import wt.pom.Transaction;

import wt.session.*;

import wt.util.WTException;
import wt.util.WTPropertyVetoException;

/**
Star import also lead to Rule violation but that is of Info
type.
But use this technique only if it's required.
Don't make this as a general practice. **/
```

9.23 Illegal Throws (Major)

Throwing java.lang.Error or java.lang.RuntimeException or Throwable is almost never acceptable.

9.24 Don't Import Java Lang (Minor)

Avoid importing anything from the package 'java.lang'. These classes are automatically imported

9.25 Final Field Could Be Static (Minor)

If a final field is assigned to a compile-time constant, it could be made static, thus saving overhead in each object at runtime.

9.26 Dodgy - Write to static field from instance method (Critical)

This instance method writes to a static field. This is tricky to get correct if multiple instances are being manipulated, and generally bad practice.
You have to avoid it at any cost. This is very dangerous practice.
Below code is an example of such violation where static field is manipulated from a non-static method.

```
public class GettingSequence implements RuleAlgorithm {

    private static List<String> names;

    public GettingSequence() {
        names = new ArrayList<String>();
        // TODO Auto-generated constructor stub
    }

    public void init(){
        names = new ArrayList<String>();
    }
```

The steps to avoid this is very simple.

```
public class GettingSequence implements RuleAlgorithm {

    private static List<String> names = new ArrayList<String>();
```

But what if we are instantiating the field by calling a method and that method is throwing some exception.
Then the best way to do it by using static initialize block.

```
public class GettingSequence implements RuleAlgorithm {

    private static String propertyValue;

    static {
        try {
            propertyValue = getPropertyValue("wt.codebase.location");
            /*
             * you can write it like this also. propertyValue =
             * WTProperties.getLocalProperties
             * ().getProperty("wt.codebase.location");
             */
        } catch (IOException ex) {
            /*
             * Handle the exception here.
             */
        }
    }

    private static String getPropertyValue(String key) throws IOException {
        if (key == null)
            return "";
        return (WTProperties.getLocalProperties()).getProperty(key);
    }
```

9.27 Performance - Method concatenates strings using + in a loop (Critical)

The method seems to be building a String using concatenation in a loop. In each iteration, the String is converted to a StringBuffer/StringBuilder, appended to, and converted back to a String. This can lead to a cost quadratic in the number of iterations, as the growing string is recopied in each iteration.

Better performance can be obtained by using a StringBuffer (or StringBuilder in Java 1.5) explicitly.

For example:

```
 //              This   is   bad
 String s = "";
 for (int i = 0; i < field.length; ++i) {
 s = s + field[i];
 }

// This is also very bad practice
String s = “”;
for (int i = 0; i < field.length; ++i) {
                s = buf.new StringBuffer().append(field[i]) .toString();
}
 // Better rule compliance code
 StringBuffer buf = new StringBuffer();
 for (int i = 0; i < field.length; ++i) {
   buf.append(field[i]);
 }
```

String s = buf.toString();

9.28 Dodgy - Redundant nullcheck of value known to be non-null (Critical)

This method contains a redundant check of a known non-null value against the constant null.
Consider below code snippet it's clearly showing if con == null it will throw exception so there is no need to check whether con is null or not.

```
try{
        /*
         * Get the Connection
         */
        con = ConnectionManager.getConnection();

        if(con == null)
        {
                if(DEBUG){LCSLog.debug("Unable to Connect the database");}
                throw new SQLException();
        }

        try {

                if (con != null) {
                        StringBuffer insertQuery = new StringBuffer("INSERT INTO ");
                        insertQuery.append(getTableName());
                        insertQuery.append("(");
```

9.29 Empty Catch Block (Critical)

Avoid empty catch blocks at any cost.
If you are catching an exception you must handle it or show it.
If you left it blank then there will be no stack trace also.

9.30 Empty If Stmt / Unconditional If Statement (Critical)

Empty If Statement finds instances where a condition is checked but nothing is done about it.
Do not use if statements that are always true or always false.

9.31 Correctness - Value is null and guaranteed to be dereferenced on exception path (Critical)

There is a statement or branch on an exception path that if executed guarantees that a value is null at this point, and that value that is guaranteed to be dereferenced (except on forward paths involving runtime exceptions).

Usually this violation happens at the time of closing connection or PreparedStatement.

```
private static void closedConnections1(Connection connection,
        PreparedStatement pstmt) {
    try {
            pstmt.close();
            connection.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

is wrong

```
private static void closedConnections(Connection connection,
        PreparedStatement pstmt) {
    try {
        if ((pstmt != null) && (!pstmt.isClosed())) {
            pstmt.close();
        }
        if ((connection != null) && (!connection.isClosed())) {
            connection.close();
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

is right

9.32 Dodgy - Questionable cast to concrete collection (Critical)

This code casts an abstract collection (such as a Collection, List, or Set) to a specific concrete implementation (such as an ArrayList or HashSet). This might not be correct, and it may make your code fragile, since it makes it harder to switch to other concrete implementations at a future point. Unless you have a particular reason to do so, just use the abstract collection class.

It's better to avoid casting as the code below

```
private static List consolidatedError(Collection col) throws IOException {
    List<String> list = new ArrayList<String>();
    if(col instanceof ArrayList){
        list= (List<String>) col;
    }
    return list;
}
```

Instead of that it's better to use addAll method or constructor.

```
private static List consolidatedError(Collection col) throws IOException {
    List<String> list = new ArrayList<String>();
    if(col instanceof ArrayList){
        list.addAll(col);
    }
    return list;
}
private static List consolidatedError(Collection col) throws IOException {
    if(col instanceof ArrayList){
        return new ArrayList(col);
    }
    return new ArrayList();
}
```

9.33 Correctness - Field only ever set to null (Critical)

All writes to this field are of the constant value null, and thus all reads of the field will return null. Check for errors, or remove it if it is useless.
Try to avoid assigning null to any field specially if it's static.
At the time of declaration you may assign the field null but through constructor or initialize block or by setter method make sure the value is not null throughout the class.

9.34 Signature Declare Throws Exception (Major)

It is unclear which exceptions that can be thrown from the methods. It might be difficult to document and understand the vague interfaces. Use either a class derived from RuntimeException or a checked exception.
Don't throw **Exception** in a method declaration also.

```
private static void findOrganizationByName(String orgName) throws Exception {
    QuerySpec qs = new QuerySpec(WTOrganization.class);
    WTOrganization localWTOrganization = null;
    qs.appendWhere(new SearchCondition(WTOrganization.class,WTOrganization.NAME,SearchCondition.EQUAL,orgName),new int[]
    { 0 });
    QueryResult qr = PersistenceHelper.manager.find((StatementSpec)qs);
    while(qr.hasMoreElements()){
        localWTOrganization = (WTOrganization)qr.nextElement();
    }
}
```

Not good considered as bad practice.

```
private static void findOrganizationByName(String orgName) throws WTException {
    QuerySpec qs = new QuerySpec(WTOrganization.class);
    WTOrganization localWTOrganization = null;
    qs.appendWhere(new SearchCondition(WTOrganization.class,WTOrganization.NAME,SearchCondition.EQUAL,orgName),new int[]
    { 0 });
    QueryResult qr = PersistenceHelper.manager.find((StatementSpec)qs);
    while(qr.hasMoreElements()){
        localWTOrganization = (WTOrganization)qr.nextElement();
    }
}
```

Better.

9.35 Use Index Of Char (Major)

Use String.indexOf(char) when checking for the index of a single character; it executes faster.

```
private static int findOccurence(String emailId) throws WTException {
    return emailId.indexOf(".");
}
```

Takes more time to execute.

```
private static int findOccurence(String emailId){
    return emailId.indexOf('.');
}
```

Takes considerably less time to execute

9.36 Boolean Expression Complexity (Major)

Restricts nested boolean operators (&&, || and ^) to a specified depth (default = 3).
The below code will trigger this kind of violation

```
private static boolean checkMailId(String emailId) {

    if (emailId.contains(".") && !(emailId.contains("\\"))
            && !(emailId.contains("/")) && emailId.endsWith("com")
            && !(emailId.contains("*"))) {
        return false;
    }
    return true;
}
```

The compliance code

```
private static boolean checkMailId(String emailId) {
    return (checkChar(emailId) && checkString(emailId));
}

private static boolean checkChar(String emailId) {
    return emailId.contains(".")
            && !(emailId.contains("\\") && !(emailId.contains("/")));
}
```

```
private static boolean checkString(String emailId){
    return emailId.endsWith("com")
    && !(emailId.contains("*"));
}
```

9.37 Method Name (Major)

Method name must start with small letter , may contain upper case but must not have any symbol like "_","-" etc.

9.38 Hidden Field (Major)

Checks that a local variable or a parameter does not shadow a field that is defined in the same class.

Local variable name and parameter name should be different than the field name.

```
private double salary = 5.95;

private void calculateSalary()
{
    double salary;
```

should avoid.

9.39 Inefficient String Buffering (Major)

Avoid concatenating non literals in a StringBuffer constructor or append().

```
private void extractMailId(String mailIds) {
    final String[] mail = mailIds.split("|");
    final StringBuffer sb = new StringBuffer();
    for (String str : mail) {
        sb.append(str + "\n");
    }
}
```

very bad practice, violates the rule

```
private void extractMailId(String mailIds) {
    final String[] mail = mailIds.split("|");
    final StringBuffer sb = new StringBuffer();
    for (String str : mail) {
        sb.append(str).append("\n");
    }
}
```

Rule compliance code.

9.40 Cyclomatic Complexity (Major)

Checks cyclomatic complexity of methods against a specified limit. The complexity is measured by the number of if, while, do, for, ?:, catch, switch, case statements, and operators && and || (plus one) in the body of a constructor, method, static initializer, or instance initializer. It is a measure of the minimum number of possible paths through the source and therefore the number of required tests. Generally 1-4 is considered good, 5-7 ok, 8-10 consider re-factoring, and 11+ re-factor now !

9.41 Javadoc Method (Major)

Checks the Javadoc of a method or constructor. By default, does not check for unused throws. To allow documented java.lang.RuntimeExceptions that are not declared, set property allowUndeclaredRTE to true. The scope to verify is specified using the Scope class and defaults to Scope.PRIVATE. To verify another scope, set property scope to a different scope.

In order to avoid this violation use proper java doc comments.Below are some examples.

For class, non-parametarized constructor , fields

```
/**
 * Returns the serial number to be used for numbering of Place holder part.
 *
 * @version 'true' 1.0
 *
 * @author 'true' Viraj Koli
 *
 * @author 'true' 12815 Shambhavi Kumar
 **/
public final class CADNumberingUtil {
    /**
     * Private variable for logger.
     */
    private static final Logger LOG = LogR.getLogger(CADNumberingUtil.class
            .getName());

    /**
     * To hide the default constructor.
     */
    private CADNumberingUtil() {
        // TODO Auto-generated constructor stub
    }
```

It's also important that your comment must end with a (.)

```
/** Static filed to store serial version UID. **/
private static final long serialVersionUID = 8980193532760592607L;

/** class name of this class. **/
private static final String CLASSNAME = CADDocumentCreationListenerService.class
        .getName();

/**
 * Private variable for logger.
 */
private static final org.apache.log4j.Logger LOG = wt.log4j.LogR
        .getLogger(CLASSNAME);

/**
 * Consolidated error message for incorrect user input.
 */
private List<String> consolidatedError = new ArrayList<String>();

/**
 * Static final variable to store next line character.
 */
private static final String NEXTLINE = "\n";

/**
 * Static final variable to store the internal name of End Item Part.
 */
private static String endItemPart;

/**
 * Static variable for Separator.
 */
private static final String SEPARATOR = " , ";
```

For Parametarized constructor

```
/**
 * Constructor of {@link ████TransferBOM}.
 *
 * @param filterTypeReference
 *            filter the specified type reference
 *
 * @param viewName
 *            view name in which the Part will change
 *
 * @param folderOID
 *            the folder oid in which the newly created part will be saved
 */
public ███TransferBOM(String filterTypeReference, String viewName,
        String folderOID) {
    this.filterTypeReference = filterTypeReference;
    this.viewName = viewName;
    this.folderOID = folderOID;
}
```

For methods.

```
/**
 * Data utility for Item(WTPart) Picker.
 *
 * @author 'true' 12805 kaushik.das@itcinfotech.com
 *
 * @version 'true' 1.0
 *
 */
public class ItemPicker extends AbstractDataUtility {

    /**
     * Private variable for Logger.
     */
    private static final Logger LOG = LogR
            .getLogger(ItemPicker.class.getName());

    /**
     * Over-ridden method of {@link AbstractDataUtility}.
     *
     * @param paramString String
     *
     * @param paramObject Obejct
     *
     * @param paramModelContext {@link ModelContext}
     *
     * @return object
     *
     * @exception WTException throws {@link WTException}
     */
    public Object getDataValue(String paramString, Object paramObject,
            ModelContext paramModelContext) throws WTException {
        LOG.debug("Entering Item Picker .......");
```

9.42 Avoid instantiating objects in loops (Major)

Detects when a new object is created inside a loop.

As the name suggest avoid constructing object inside loop.

9.43 Avoid Print Stack Trace (Major)

Avoid printStackTrace(); use a logger call instead.
Instead of Print Stack Trace use logger.error method.

9.44 Integer Instantiation (Major)

Avoid instantiating Integer, calling new Integer() causes memory allocation. Integer.valueOf() is more memory friendly.

9.45 Unused formal parameter (Major)

Avoid passing parameters to methods or constructors and then not using those parameters.

9.46 Boolean Instantiation (Major)

Avoid instantiating Boolean objects; reference Boolean.TRUE or Boolean.FALSE or call Boolean.valueOf() instead.

9.47 Preserve Stack Trace (Major)

Avoid throwing New exception in catch block as the original stack trace may be lost because of this.

9.48 Simplify Conditional (Major)

No need to check for null before an instanceof; the instanceof keyword returns false when given a null argument.
Below code will generate this violation
if(obj != null && obj instanceof ArrayList)

9.49 Constructor Calls Overridable Method (Major)

Calling overridable methods during construction poses a risk of invoking methods on an incompletely constructed object and can be difficult to discern.
It may leave the sub-class unable to construct its superclass or forced to replicate the construction process completely within itself, losing the ability to call super().

If the default constructor contains a call to an overridable method, the subclass may be completely uninstantiable.
Note that this includes method calls throughout the control flow graph - i.e., if a constructor Foo() calls a private method bar() that calls a public method buz(), this denotes a problem.

Consider the example below

```
public class SeniorClass {
  public SeniorClass(){
      toString(); //may throw NullPointerException if overridden
  }
  public String toString(){
    return "IAmSeniorClass";
  }
}
public class JuniorClass extends SeniorClass {
  private String name;
  public JuniorClass(){
    super(); //Automatic call leads to NullPointerException
    name = "JuniorClass";
  }
  public String toString(){
    return name.toUpperCase();
  }
}
```

9.50 String Instantiation (Major)

Avoid instantiating String objects; this is usually unnecessary.

9.51 Close Resource (Major)

Ensure that resources (like Connection, Statement, and ResultSet objects) are always closed after use. It does this by looking for code patterned like :

Connection c = openConnection();
try {
  // do stuff, and maybe catch
something } finally {
  c.close();
}
Prior to Java 7 try to use try with resource block.

Inner Assignment (Major)

Checks for assignments in subexpressions.

```
String s;
if((s = bufferedreader.readLine()) == null)
```

9.52 Trailing Comment (Minor)

The check to ensure that requires that comments be the only thing on a line. For the case of // comments that means that the only thing that should precede it is whitespace.
It doesn't check comments if they do not end line, i.e. it accept the following: Thread.sleep(10 <some comment here>); Format property is intended to deal with the "} // while" example.

```
a = b + c; // Some insightful comment
d = e / f; // Another comment for this line
```

Above lines will lead to this type of violation.

9.53 Avoid Instanceof Checks In Catch Clause (Minor)

Each caught exception type should be handled in its own catch clause.

```
catch(WTException ex)
{
    if(ex instanceof EPMException)
    {
        //do something
    }
    else if(ex instanceof WTException)
    {
        //do something
    }
}
```

Bad practice lead to the violation.

```
catch(EPMException ex)
{
    //do something
}
catch(WTException ex)
{
    //do something
}
```

Better Rule compliance code.

Unused Imports (Info)

Checks for unused import statements.
If you are using Eclipse then type ctrl + shift + O. It will remove unused import and add necessary import statement(s). Similarly press ctrl + shift + F for formatting.

9.54 Modifier Order (Minor)

Checks that the order of modifiers conforms to the suggestions in the Java Language specification, sections 8.1.1, 8.3.1 and 8.4.3. The correct order is : public, protected, private, abstract, static, final, transient, volatile, synchronized, native, strictfp.

Usually this rule violation happen because of writing access modifier like

- static private String str
- private final static Logger LOG

9.55 Final Field Could Be Static (Minor)

If a final field is assigned to a compile-time constant, it could be made static, thus saving overhead in each object at runtime.
For better performance you may declare any constant field (global variable or class variable) as static.

9.56 Magic Number (Minor)

Checks for magic numbers.
Only 0 , 1, 2 are not magic.Replace other integer number with constant.(If Possible)

9.57 Redundant Throws (Minor)

Checks for redundant exceptions declared in throws clause such as duplicates, unchecked exceptions or subclasses of another declared exception.

Check http://plmsonarprod.itcinfotech.com:9000/rules_configuration/index/16 for list of total 587 rules.

10. PTC CASES

Below are some useful ptc case link.

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS139867 :- GC Thread

https://www.ptc.com/appserver/cs/view/solution.jsp?n=CS98135 :- **provides Indices for Performance that are missing out of the box for Windchill 10.0 and Windchill 10.1.**

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS45858 :- **Change the password of WCADMIN when WBR is installed**

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS97931
https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS71489 :- Sizing of cache

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS51570 :- Save As operation from workspace is slow in Windchill PDMLink 10.0, 10.2 (M010)

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS179494 :- The "Set Location" window is slow in Windchill 10.2

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS151136 :- Search is slow when using leading and trailing wildcard, or selecting "All Types" in Windchill

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS94989 :- Search with multiple IBA is very slow in Windchill 10.0 and 10.1

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS170265 :- Setting Equivalent link in MPSE is much slower in Windchill 10.2 compared to 9.1

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS72994 :- **How does LDAP synchronization work in Windchill**

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS37293 :- Windchill installation fails with return code = -1073741819

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS21895 :- **How to convert a vault to use a Root Folder in Windchill PDMLink**

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS133237 **:- How to get the Display Name for the Global Enumeration entry in Windchill PDMLink**

https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS69640 **:- How to filter types from the type picker in an object creation wizard in Windchill PDMLink**

https://support.ptc.com/appserver/cs/view/solution.jsp?source=subscription&n=CS176561 **permission of modifying Name and Number for WTDocument and WTPart**

How to restrict web access to Windchill PDMLink / How to temporarily disable user access to Windchill :-https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS61527
https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS71201
https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS45373

Typical steps necessary for purging CAD Document :-
https://support.ptc.com/appserver/cs/view/solution.jsp?n=CS44609

Source codes are available in src.zip