Exility Page Reference Manual

EXILANT Technologies Pvt. Ltd.		
#27, P Kalinga Rao Road Bangalore – 560 027 INDIA	20195, Stevens Creek, Blvd #220 Cupertino, CA 95014 USA	Document Version.: 5.0.0 15-Jun-2015 Reference Exility Version: 5.0.0
Web site: www.exilant.com		

Exility Page Reference Manual

Trademarked names may appear in this document. Rather than use a trademark symbol with every occurrence of a trademarked name, we use the names only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

EXILANT is the company behind Exility, a rapid enterprise web application development framework.

EXILANT Technologies Private Limited provides Information Technology (IT) solutions with marked focus on measurable value. It offers services and products in enterprise mobility, business intelligence and analytics, user experience/interface (UI/UX) design and development, enterprise application integration to BFSI, Hi-Tech, Retail, Telco, Transportation/Logistics, Textiles, Apparel, Dairy and Construction industry segments.

EXILANT was founded in 2004 and is headquartered in Bangalore, India. It is ISO 9001:2008 & ISO 27001:2005 certified and employs over 1400 people with operations in India, USA, UK, and Singapore. For more information, visit www.exilant.com.

DISCLAIMER:
This document is for informational purposes only and is provided “AS IS.” The information set forth in this document is intended as a guide and not as a step-by-step process, and does not represent an assessment of any specific compliance with laws or regulations or constitute advice. We strongly recommend that you engage additional expertise in order to further evaluate applicable requirements for your specific environment. EXILANT MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND, EXPRESS OR IMPLIED, AS TO THE ACCURACY OR COMPLETENESS OF THE CONTENTS OF THIS DOCUMENT AND RESERVES THE RIGHT TO MAKE CHANGES TO SPECIFICATIONS AND PRODUCT/SERVICES DESCRIPTION AT ANY TIME WITHOUT NOTICE.

EXILANT RESERVES THE RIGHT TO DISCONTINUE OR MAKE CHANGES TO ITS SERVICES OFFERINGS AT ANY TIME WITHOUT NOTICE. USERS MUST TAKE FULL RESPONSIBILITY FOR APPLICATION OF ANY SERVICES AND/OR PROCESSES MENTIONED HEREIN. EXCEPT AS SET FORTH IN TERMS OF AGREEMENT YOU SIGN WITH EXILANT, EXILANT ASSUMES NO LIABILITY WHATSOEVER, AND DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO ITS SERVICES INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. Except as expressly provided in any written license agreement from EXILANT, the furnishing of this document does not give you any license to patents, trademarks, copyrights, or other intellectual property.

All other product names and trademarks used in this document are for identification purposes only to refer to either the entities claiming the marks and names or their products, and are property of their respective owners. We do not intend our use or display of other companies’ trade names, trademarks, or service marks to imply a relationship with, or endorsement or sponsorship of us by, these other companies.

HOW TO SUBMIT ERRORS, OMISSIONS AND SUGGESTIONS

If you find errors, omissions or have improvement suggestions, you can send an e-mail to zaglabs@exilant.com . Please clearly outline the following in your e-mail:

1. A detailed description of the error, omission or your suggestion
2. Your contact e-mail
3. Explicitly mention if you want your name included in credits list

We firmly believe in giving credit where it’s due. We would like to publish the names, of individuals who help us improve this document. If you want your name to appear in credits list, please do mention explicitly in your email. If your suggestion is accepted and published, and if you have explicitly given us permission to publish your name, we would add your name to the credits list in appropriate document.

Table of Contents

Overview

This document is a reference manual for the HTML client side of Exility-tech. It is intended for client-side designers and programmers of a project that uses Exility.

Page

Application interacts with the user through a ‘page’. Exility uses a standard html file to render a page. When we started Exility in 2004, cross-browser compatibility was a huge issue. Also, separation of html and css was not prevalent. While our initial version had difficulties in addressing these issues, current version is well tuned to generate standard HTML5 and CSS3 compliant web pages.

A page has four aspects to it. You should understand the page in terms of these aspects before you dive into using Exility to implement the page design.

1. **Visual aspects**: The layout including color, arrangement of elements such as labels, buttons, input fields, text etc. This is more of a ‘static’ layout of elements. At the highest level, page consists of panels that determine the overall layout. Each of these panels may contain other panels, or fields (that are data-dependent) or static elements. This is the traditional ‘prototype’ html page a designer would paint. Exility provides you with a set of facilities for all standard features within a project. It also provides you a way to extend this with your own html for any special effects in a given page.
2. **Data population**: A page has some content that would change at run time, typically based on some data that is fetched from the server. At run time, data would be fetched from the server, and the page has to be ‘rendered’ using data rather than show what was put in its ‘static’ stage. Exility uses a simple naming convention to bind visual elements to data elements.
3. **Actions**: Page has to respond to user inputs, keys and mouse movements. You may define different types of actions, like fetch data from server, navigate to another page etc. Exility allows you to associate these actions with standard events, like button-click.
4. **Parameterization**: A page can be designed to respond to more than one context. The context can be conveyed through a set of parameters. For example, a search page may optionally receive value to some fields and execute a search for those values on load. We strongly recommend that a page is designed as a stand-alone page, and not assume that it is always called from another specific page. If required, the calling page should pass appropriate parameters.

We recommend that a page be first visualized in terms of its look-and-feel (visual aspects). Next you should see how it behaves with respect to run time data. And then add actions to it.

1. Parts of a Page

A page consists of page attributes, a collection of panels, an optional collection of parameters, and an optional collection of actions. These are briefly explained below before taking you to a complete reference section.

1.1 Panel

A panel is an area within the page that contains visual elements in it i.e. the page elements are organized into panels. You can decide how elements are laid out within a page, how a panel itself should look like etc.. For example, a display panel spreads the elements from left to right and top to bottom. You can choose how many elements you want in each row. A list panel would arrange all its elements in a row, and adds a row for each row of input data.

A panel may contain fields/elements or other panels.

1.2 Static Element

Element is a displayable entity that is contained in a panel. It is a static entity, in that it does not receive data either from the server, or from the user. Text element and image elements are examples of an element.

1.3 Field

Field is a display element that can receive data from server or user. Text input box, check box, radio button, output field are examples of a field.

Panels, fields and static- element determine the visual aspects of a page, with fields providing the flexibility to change the display at run time based on the data that is received.

1.4 Action

Action has no visual aspect, but represents a response to some event on the page. Fetching data from the server, executing local scrip, navigating to another page are examples of actions.

1.5 Page Parameter

It is a name-value pair that is provided as input at the time of invoking the page. This concept is similar to a function or a procedure that is called with some parameters. The page behavior could depend on the parameters it receives.

```
<?xml version="1.0" encoding="utf-8" ?>
<page xmlns="http://com.exilant.exility/page"
      name="date" title="Testing date formatting " description="Exility" module="sample"
      onLoadActionNames="startupAction"      width="700"  height="700"
      scriptsToInclude="date.js.htm" >
  <actions>
    <closeAction name="closeAction"/>
    <serverAction name="submitAction"  serviceId="saveUserDetails" submitForm="true"/>
    <serverAction name="startupAction" serviceId="date"/>
    <localAction name="clickAction" functionName="clicked"/>
    <localAction name="changeAction" functionName="changed"/>
  </actions>
  <panels>
    <displayPanel requiresGroupOutline="true" >
      <elements>
        <textInputField name="field1" label="Future Date"
                        dataElementName="sample_futureDate" />
        <outputField name="field2" dataElementName="sample_date"
                     label="Output Date" />
        <checkBoxField onChangeActionName="changeAction"  name="myCheckBox"
                       dataElementName="sample_boolean"   checkedValue="abcd" uncheckedValue="efgh"
                       checkedValueIsTheDefault="true"    onClickActionName="clickAction"  />
      </elements>
    </displayPanel>
    <gridPanel requiresGroupOutline="true" label="dates in a grid"
               tableName="dates" rowsCanBeAdded="true"  rowsCanBeDeleted="true"
               idFieldName="field2" keyFieldName="field1" actionFieldName="action" >
      <elements>
        <textInputField name="field1" label="Past Date"  dataElementName="sample_pastDate" />
        <outputField name="field2" dataElementName="sample_date" label="Output Date" />
        <hiddenField name="action" dataElementName="internal_actionName"/>
      </elements>
    </gridPanel>
    <spacerPanel></spacerPanel>
    <buttonPanel elementsPerRow="3">
      <elements>
        <buttonElement label="Submit"  hoverText="View trace data to see the values getting submitted"
          onClickActionName="submitAction" />
        <buttonElement label="Close" onClickActionName="closeAction" />
      </elements>
    </buttonPanel>
  </panels>
</page>
```

Example 1 Page definition

1.8 What is expected of you?

Every project should have a lead designer. If you are the lead designer, you should have good conceptual understanding of HTML and CSS. Exility provides default.css that has a default rendering for all the tags that is used. You should be able change these to get the desired visual effects a projects wants. You should be able to create new CSS classes to be used for any special features a page may use.

If you are a page designer, you need not have extensive knowledge of HTML and CSS, but a conceptual understanding and basic working knowledge helps. However, you should be a good ‘programmer’. We expect that you:

- Read the manual and understand before you start your work. This manual is still in its draft stage. Your comments are of great help to improve this.
- Do not hesitate to ask others if you do not understand something. Exility is in its initial stages, and is exploring newer ways of developing software. These concepts may not be intuitive, and the documentation is in its nascent stage. So, it is not something against you if you do not understand. You should refer to the samples, or ask your team mate, or sought help from Exility team. Many a times, you can get away by simply copying something, but that is not what we want from you. (Last thing we want is to create great copy-cats) You should be able to understand how and why a feature is used.
- Create a page.xml that is valid as per the provided schema. You CAN create the file with a notepad, but that is a NO-NO. You should use an editor that can provide context-sensitive suggestions as well as validate the whole document based on the schema provided.
- Do not ignore errors as provided by page generator. Exility page generator generates a page even if there are some errors in the page for debugging purposes only.
- Do not ignore warning as a routine. Do look at them, and ensure that you are ready to live with them.
- Do not treat Exility as a black box, because it is not. It is nowhere near being an end-user out-of-the-box package. It is to be treated as your assistant that helps you in reducing your typing work. You are THE designer, and you decide what you want. Exility helps you to enter your design and automate some of the processes required to build the application based on your design.
- Be inquisitive. Have hunger to know how things work. Understand how things work at run time, and how Exility manages to create the components required at run time. If your page is not showing-up the way it should, may be take a look at the generated .htm file. Find out what is wrong with the HTML. That will help you improve your knowledge about HTML, and will help us in fixing it if it is beyond you.
- Review all relevant aspects of your components before concluding that the problem is beyond you. It could be typing error. May be it is the result of one of the warnings you have ignored.
- Take bottom line for your components. They MUST behave the way they are meant to be. If not, it is a mute point whether it is because there is a bug in Exility, or an error in your component. If you believe that the issue with Exility, raise an issue, and follow-up till you get it resolved.

1.12 Page Customization:

This feature is to enable having different labels depending on a “Key” specified at the page header level. The label value for the field is specified in the dataDictionary .The option in the page can be specified as follows

```
<page
  ...
  customLabelName = "key1"
  ...
  >
  ...
</page>
```

Example 2 Page Customization

At the time of rendering,

1. If there is a label specified for the field at the page level, it will be displayed.
2. If there is no label specified for the field, then customLabelName will be checked and if there exists a custom label for that data element with the key matching the "customLabelName", then its value will be displayed as label.
3. Otherwise, the default label of the data element will be displayed.

Custom labels for the dictionary can be added as follows,

```
<dictionary...>
 ...
 <datagroup ...>
  ...
  <dataElement ...>
   <customLabels>
    <customLabel name="key1" vaue="Label1" />
    <customLabel name="key2" vaue="Label2" />
    ...
   </customLabels>
  </dataElement>
  ...
 </datagroup>
 ...
</dictionary>
```

Example 3 Custom label in Dictionary

2. Details

2.1 Page Attributes

These are attributes of the page, at the page level, to be specific inside the <page …> tag.

Name	Type	Mandatory?	Default	Explanation
Name	string	yes	-	Name of the page. This should match the name of the file. E.g. if the file name is `samplePage.xml`, then field “name “must be set to “`samplePage`”.
Type	string	No	-	No specific meaning to Exility, but project may want to categorize pages into types. For example search, view, add-modify etc.. This field may also be used to do any specific customization in a project for different types of pages.
Module	string	No	-	Project module name to which this page belongs to. For documentation only, and not used by Exility. Your project will set a standard for this.
Description	String	No	-	Page description, for the designer’s documentation purpose.
Title	String	Yes	-	Title of the html page. This is not used for display as of now. However, a project can use it as a standard title to be displayed at the top of each page. Optionally, a project may also decide to use this as the title of the browser window. Page generator can be customized for a project to take care of these.
width, height	number	No	Application Parameter	Width and height of the page in pixels. Typically, your project should have a standard for most pages. Such a standard value should be specified in ApplicationParameters.xml as pageWidth and pageHeight. You specify these values for a page only if it has to be different for this page.
popupHeight, popupWidth, popupTop, popupLeft	number	No	-	If this page is used as a popup (either as a code picker, or explicitly in a navigation action) you can control the top corner where the window is positioned, and its size using these parameters. If omitted, pages size is used and the window is positioned at left top corner of the window.
minParameters	number	No	0	Minimum number of parameters with which this page should be invoked. Note that a page parameter may be either mandatory, or optional. This

Name	Type	Mandatory?	Default	Explanation
				number is the total number of parameters that must be supplied for the page. An error message is displayed if this page is not supplied with these many parameters. (refer to page parameter for further validations of parameters)
onLoadActionNames	String	No	-	Comma separated names of the actions to be executed when the page is loaded. Actions with these names are to be defined under actions collection.
onModifyModeActionNames	String	No	-	Comma separated names of the actions to be taken when page is loaded. And the page mode is determined to be 'modify'. These actions are performed instead of the ones specified above. (onLoadActionNames). A page is considered to be in modify mode if all the page parameters marked as primary key are supplied at the time of loading the page.
fieldsToDisableOnModifyMode	String	No	-	Many a times, some fields are not to be modified during update mode. They are entered during entry mode. Specify such fields separated by commas.
minParametersToFireOnLoad	number	No	0	This is different from minParameters. It should be greater or equal to minParamaters. This is applicable only if you have an action to be taken on load. The action will be taken only if a minimum of these many parameters are supplied. Note that this condition applies to onLoadActions and not onModifyModeActions.
scriptsToInclude	String	No	-	Comma separated list of project specific, as well as page specific pages to be included in this html page. Note that your project will have a standard on naming convention. Typically, if `samplePage.htm` is to be the name of html file, then `samplePage.js` or `samplePage.js.htm` ** could be the name of the script file for that page.
reloadActionName	String	No	-	Name of the action to be triggered when control returns to this page from a page you would have navigated with retainState option. Refer to navigationAction.

Name	Type	Mandatory?	Default	Explanation
buttonsToHideForPicker	String	No	-	If a page is used in 'normal' mode as well as code picker, you may want to hide some buttons in picker mode. e.g. You may use a search page for normal search, in which case you may have NEW button. But if the page is used as a code-picker pop-up, user should not be allowed to go to any other page from there. Provide a comma separated list of button names to be hidden in such a case.
trackFieldChanges	boolean	No	false	Set this parameter to true if you want Exility to track if user has modified any field and warn the user on either closing the page or navigating out of the page. Typically, you would like to warn if it is an entry page, but not bother if it is a search page. Note that this is based on any change. If the user changes a field back to its original value, page is still considered to be 'modified'.
validateOnlyOnUserChange	boolean	No	false	A field is validated, by default, even if user has not changed the value i.e. even when data is loaded from the server. This generally does not create any problem, except if you have lived with some issues with existing data on the server. I do not like this flag, but had to give this feature to take care of a situation in a project. If this is set to true, field values are validated only if the field is changed by user. Note that this attribute is also available at the field level. An unchanged field is not validated if this attribute is set to true either at page level, or at field level. This option can be implemented in two ways. In exilityParameters.js file we need to mention exilParms.trackFieldChanges = true;. But if we don't want at application level then at page level also we can specify the tarckFieldChanges attribute.
breadCrumpTitle	String	No		Now there is an option to show the page navigation path in Exility. If the user declares breadCrumpTitle="Page title", then page title will be shown in the navigation path.
formValidationFunction	String	No		Before sending data to the server, the client validates the data by using the associated data types and mandatory attributes only. If there is some business validation needs to be done before submitting the data to the server, a JavaScript function can be written and is specified as "formValidationFunction" in the page. Then, this particular function is

Name	Type	Mandatory?	Default	Explanation
				called before submitting the data to the server. Based on the return value of this function, the data is submitted to the server or else error message is displayed.
hasChartFields	boolean	No	False	If the page contains any kind of chart field, it is necessary that user has to declare this attribute at page level itself.
hotkeyFunction	String	No		This is a user function that helps the application developer to get the keyCode on a particular page. One of the scenarios where it can be useful is as follows. Developer can write a user function and specify it as hotkeyFunction in a page. Then the developer should be able to get each of the keyCodes that are pressed on that page. Based on the pressed keyCode, the developer can focus various fields within the page (like a short cut or hot key). There is also provision to have the cntrl alt and combinations of keys as hotkeys. Example for the JavaScript is as follows.

```
var isCtrlPressed = false;
var isAltPressed = false;
function onclickact(evt)
{
    var KeyCode = null;
    if(evt)
        KeyCode = evt.which;
    else
        KeyCode = event.keyCode;
    if(KeyCode == 17)
        isCtrlPressed = true;
    else if(KeyCode == 18)
        isAltPressed = true;
    else
    {
        if(isCtrlPressed)
        {
            alert(KeyCode + " pressed with ctrl");
```

Name	Type	Mandatory?	Default	Explanation

				isCtrlPressed = false;
				        }
				        else if(isAltPressed)
				        {
				            alert(KeyCode + " pressed with alt");
				            isAltPressed = false;
				        }
				        else
				            alert(KeyCode + " is pressed");
				    }
				
				    // This avoids the bubble of the event. Comment it if not needed
				    if(evt && evt.preventDefault)
				        evt.preventDefault();
				    else
				        event.returnValue = false;
				}
				
				hot key function can also be declared at the application level. In Exility parameters In exilityParameters.js, the application team can include a new parameter called 'exilParms.appHotKeyFunctionName'. If this is mentioned, then functions with same name need to be present in the page which loads Exility (usually the home page). This function should be hand crafted by the application team as per their needs. They need to communicate to the page that originated the events using the exilityPageStack.top() interface. If the hotkeyfunctionname is provided at the page level, it overrides the application level hot key function.

*** Note: Naming the file as “samplePage.js.htm” -- the “.htm” extension gets you the context sensitive help in MS Visual Studio*

Page will be modified by the user. But when we click on the menu after this we will not get warning message. As menu is implemented outside Exility, we could not capture the scenario where the page has been modified. At page level, when the page has been explicitly changed by the user, we will set a parameter, P2.pageChangedByUser = true; So on click of the menu, we need to check for this parameter using currentActiveWindow and then we need to show the warning message. Setting this variable to false has to be taken care by the application teams. Reference implementation is in our demo application.

2.2 Page parameter attributes

Name	Type	Mandatory	Default	Explanation
Name	string	yes	-	Name of the page parameter.
isRequired	Boolean	No	false	Whether this page parameter is mandatory or not. An error is generated at the time of loading the page if this parameter is mandatory and the caller is not supplying a non-zero non-empty string.
defaultValue	Appropriate type	No	-	If a value is not supplied, this value s assumed. Use this feature if your design needs a default value other than 0 or empty string.
dataElementName	String	No	name	Data dictionary element name that this parameter refers to. A dictionary element describes the business meaning of an element, in addition to defining its data type. The field defaults to the name of the parameter. That is, if the data element name is same as the parameter name, then you need not specify it.
setTo	String	No	name	Value received for this parameter is typically stored for use during the page execution. It could be either a field defined in the page, or a local (java script) variable. This field defaults to the parameter name. A field with the same name is first tried, failing which a local variable is created and assigned this value.
isPrimaryKey	Boolean	No	No	If this page is used in both new/Modify mode, is this parameter one of the primary keys?. If more than one parameter is specified as primary key, then you typically provide values for all of them in Modify mode, and none in New mode.

2.3 Basic element attributes

These attributes are applicable to element, field as well as panel. Attribute marked as html attribute is typically not processed by Exility, but is passed as an html attribute of the corresponding element. E.g. if you specify `width=600px` for a panel, the corresponding div tag will look like

```
<div style=”width: 600px; …..>
```

Name	Type	Mandatory	Default	Html	Explanation
Name	string	Yes/No	-	No	Name of the panel, field or element. It is mandatory for field, but optional for elements and panels. However, if a panel has some action associated with it, like tabbed panel, or expandable/collapsible you MUST specify a name. Also, name is mandatory for an element that is to be hidden/showed at run time. In general, it is a good practice to name all panels, while element can be named on a need basis.
Label	string	No	-	No	For a panel this is the heading. For field, this is the label with which the field is associated with. For an element, this does not have specific meaning.
Width, height	String	No	auto	Yes	Auto means let the browser decide the width. Using “auto” is appropriate in most situations. However if you have understood how the width works for html, you can use width and height to mange the rendering better. Once again, you must study the way browsers use the width and height attributes before using width and height attributes. Exility does not process this attribute, but passes it on as an attribute of the corresponding html element
Hidden	Boolean	No	False	-	A hidden element is not displayed when the page first loads. The way the element is hidden depends on the over-all layout. A collapsible panel (you will read about it later) is “hidden” by collapsing it. In other cases, the html attribute style “display” is set to ‘none’ to hide the element. You can use features associated with an action to hide/show elements at any time during the execution of the page, or you can control them through your custom script.
Align	String	No	auto	Yes	Possible values are left, right and center. This is how the element will be aligned with respect to its parent. If it is left as ‘auto’ the default alignment as specified by the browser will be used. This property is passed to the browser, and not processed

Name	Type	Mandatory	Default	Html	Explanation
techNotes	String	No	-	No	Technical note about this element. For documentation purposes only.
businessValidation	String	No	-	No	Documentation purpose. Intention is to write the business validation as a business user understands, so that it can be verified and confirmed by the customer before coding it.
hoverText	String	No	-	Yes	Tool tip for the element. Browser displays this text on mouse over on this element.
numberOfUnitsToUse	Number	No	1	No	A panel displays each of its elements in one 'unit' by default. For a list panel, a unit is a column, where the label of the element is displayed in the header row, and the field values are displayed in subsequent rows. In a display panel, a unit means the label and the field value. This default behavior can be modified by specifying a value here. If it is zero, it means that the field will latch-on to the previous unit. In a display panel, both the label and the field values will be appended to the previous unit, while in a list panel the label is appended in the header row, and the data is appended in the data row. If you specify a value more than one, the element will be spread across as many units. This feature is useful if you need alignments other than the default one. Also, hidden fields have to use 0 units. In a display panel, a unit consists internally of two cells of a row, thus providing a way to align all labels and fields. When you choose a value other than one, this alignment will be disturbed, and you should manage that appropriately.
cssClassName	String	No	-	Yes	All elements of a page have a default css provided by Exility, and are provided the attribute in default.css file in a given project. A specific element can override this default to have its own look and feel. Note that the default.css should define this name. This feature is frequently used with textElement to render different types of texts in a project.
onClickActionName	String	No	-	No	Do you want any action when user clicks on this element? Actions collection should have an entry for this name. If you assign this attribute for a non-primitive element, you should be aware of the event propagation of html.
htmlAttributes	String	No	-	Yes	Any other html attribute you want the rendered element to have. This must be of the form attribute="value" (quotes required) and you should ensure that the attribute is the right one for the intended html element. If the attributes are for

Name	Type	Mandatory	Default	Html	Explanation
					style, you should specify style="attribute:value;…." etc..
bulkCheck	Boolean	No	No	-	If the user declares this attribute at grid or list level, one check box will be provided at the header level provided that grid or list contains a check box. On click of the header level check box, all the rows in that column will be checked or unchecked.
footerLabel	String	No	-	-	If the user declares this attribute at grid or list level, one footer row will be added to that grid or list and the mentioned footerLabel will be shown for that column.

3. Panels

3.1 Panel attributes

For aligning the panels properly we have provided an attribute called alignPanels in applicationParameters. If you give true for this variable then if the user did not mentioned the width for the panel then by default width will be 100%.

Name	Type	Mandatory	default	html	Explanation
Element attributes					Refer to basic element attributes. All of them are applicable for a panel.
requiresGroupOutline	boolean	No	false	-	A boundary is drawn around the panel using `fieldSet` html element.
isCollapsible	boolean	No	false	No	A collapsible panel will have a 'twisty' attached to that. Click on the twisty or the icon attached that will expand/collapse the panel. Hide/show of such a panel through an action will also result in expansion/collapse of the panel. Note that you MUST specify name for such a panel.
tabLabel	string	No	-	No	Applicable if this panel is part of a tabbed group. This text is used as the text on the tab, while the label is used as the heading for the panel.
columnWidths	String	No	Auto	Yes	This is a comma separated list of widths, one for each cell of a row in the panel. In a display panel, number of cells is twice the number of elements per row. For a list or grid panel, number of cells is counted as number of elements whose `numberOfUnitsToUse` is greater

Name	Type	Mandatory	default	html	Explanation
					than one. You must specify the widths in the right order. Note that you should either have predictable width for the element inside the column, like a text element or a text input box. If you have an output field in the cell, it may not wrap within the allotted width, resulting in some unexpected width adjustments. In such case you should use `maxLength` for output field, or specify appropriate style attributes to take care of it. Specifying width is advisable, but your project should have a standard/guideline to achieve this. Note: this feature is yet to be implemented, but you can go ahead and specify this in your xml.
noBorder	Boolean	No	False	Yes	By default, a border is rendered around the panel. You may use this attribute to avoid the border.
repeatOnFieldName	String	No	-	-	See sub-section on repeating panel
labelFieldName	String	No	repeatOnFieldName	-	See sub-section on repeating panel
repeatingPanelName	String	No	-	-	See sub-section on repeating panel
youngerBrother	String	No	-	-	See sub-section on repeating panel
elderBrother	String	No	-	-	See sub-section on repeating panel
slideEffect	string	No	-	-	Now there is an option to slide the panel.(it may be grid, or list or the whole panel itself.) We can hide the panel and while hiding a panel, slide effect will be there. This you can achieve in two ways. 1. fromRight – Panel will be hidden from right. 2. fromLeft. – Panel will be hidden from Left. 3. None. – There will now be any sliding effect.

3.2 Display Panel

The displayPanel is the most common panel that you use. It can be used to organize sub-panels within it, or directly contain fields or elements.

Name	Type	Mandatory	Default	Explanation
Panel attributes				Refer to basic panel attributes.
elementsPerRow	Number	No	1	Indicates how many elements/fields/sub-panels are to be laid out per row. For a display panel, this is typically set to 2, so that you have two sets of fields displayed per row.
tableName	string			Applicable if the fields within this panel are going to get their values from a grid rather than name/value pairs from server. All references to fields within this panel from any context outside the panel should use an 'absolute name' which is table name + '_' + name. for example, if you have a field named id inside with the table name as customerValues, other elements have to refer to this field as customerValues_id. Note that the fields within this panel will not prefix the table name.
linkedTableName	string			Table name of a list panel to which this display panel is linked to. This display panel acts as an editing area for rows in the list panel
recordName	string			

3.3 Button Panel

The buttonPanel is designed for rendering action buttons. Actions buttons may be rendered under display panel also, but having them in a separate panel provides the flexibility of controlling the way buttons are rendered in your project. If an element is placed beside a table, it used to come in next line. Hence instead of table we have put span for the button panel. But this will be an application level parameter. If you provide spanForButtonPanelRequires = True; then buttons will be aligned by using span element. This is applicable only for the page layout type 2.

Name	Type	Mandatory	Default	Explanation
Panel attributes				Refer to basic panel attributes.
renderingOption	String	No	nextToEachOther	Buttons can be rendered in the traditional manner next to each other as HTML input buttons. Alternately they can be rendered as options inside a drop-down with a GO button in front. This option is suitable when there are many buttons. (inADropDownList). If you don't want button label to be GO for dropdowlist then you can provide buttonLabel option. By default value is GO.
tableName	String	No	-	If the buttons to be displayed are determined at run time, you choose the name of the grid in which the valid button names are fetched at run time. Note that the possible list of buttons has to be known at design time, and you should have added button element corresponding to each of these buttons. You may choose to enable only a subset of this at run time. Refer to buttonManagement sample supplied by Exility for an example. The grid should have one column (with a heading row) and should contain one row per button to be enabled with name of the button as its first column.

3.4 List Panel

ListPanel is designed to display rows of data, and allow user to select row/s for any specific action. It also provides pagination of rows. If the number rows are less than the page size then page information will not be shown. You may either use the clickActionName to act on the row, or may design specific fields within the row to act as action buttons for that. For example, if you are displaying personal information, a click on mail-id may result in a mail-to action, while clicking on the employee id opens up a window with all details of the person. We do not recommend having input fields in list, as it is not designed for it. If the objective is to edit rows, use grid panel instead.

Name	Type	Mandatory	default	Explanation
Panel attributes				Refer to basic panel attributes.
tableName	string	yes	-	Name of the grid/table within the dc in which data is fetched from server. A list panel will have as many rows in it as many

Name	Type	Mandatory	default	Explanation
				data rows are there in the table in the dc.
showHeader	Boolean	No	True	To control the display of the header row ;default - the header is displayed
addSeqNo	Boolean	No	No	A sequence number is generated and prefixed to each of the row.
actionDisabled	Boolean	No	No	When mouse hovers on a row, the row is highlighted to provide a slight focus so that columns are easily grouped across the row. Also, an action may be triggered on click/double click on the row. In a certain context, these may not be valid. For example the page behavior is such that after clicking on a row, the data is edited. During this editing, user should not click on another row. In such a case, you may disable the action. If you want the list panel to be in such a disabled state on startup, use this attribute. During execution, use an API to enable/disable it.
multipleSelect	Boolean	No	No	Normally an action is taken immediately on click/dbl click of a row in a list. That is the row is selected and immediately acted upon. But you may have situations where you want to select several rows, and press a button to act on them, e.g. delete all selected rows, or include them for some action. In such a case, select on a row will select that row, in addition to the ones that are already selected. Click on a selected row to unselect it. If this option is false, when a row is selected, earlier selected row is automatically unselected.
onDblClickActionName	String	No	No	Name of the action taken on double click of a panel row. This action should be defined inside action collection.
messageNameOnNoData	String	No	No	A messageId, text of which is to be displayed instead of an empty table if there is no data to be displayed in a list panel. If you want the table to be just hidden with no data, you may specify an empty string in this attribute.

Name	Type	Mandatory	default	Explanation
				Refer to section on Handling No Data
simulateClickOnFirstRow	Bool	No	False	On loading the rows, should the first row to be clicked, so that any related actions trigger and fetch relevant data
paginationButtonType	String	No	linear	How should the pagination information be displayed. Linear shows them as “page nn of nn pages. <<first <prev next> last>>”. dropdown displays page numbers in a drop-down with a go button in front. In the future, we should be able to add other ways of displaying pagination. Also, this attribute signals pagination for the list.
pageSize	number	No	-	Number of rows per page. Default is taken from ApplicationParameters.
repeatingColumnName	String	No	-	See sub-section on repeating column
idFieldName	string	No	-	See sub-section on repeating column
childTableName	string	No	-	See sub-section on repeating column
childKeysTableName	string	no	-	See sub-section on repeating column
allColumnsAreFilterable	Boolean	no	-	If the user declares this attribute at grid or list level, all the columns will be provided with the filter option.
allColumnsAreSortable	Boolean	no	-	If the user declares this attribute at grid or list level, all the columns will be provided with the sort option.
columnSumCssClassName	string	no	-	User can have the different CSS for footer.
freezeColumn	string	no	-	A grid or list can be split into two parts. First part will be fixed and for the second part of the grid or list, horizontal scroll bar will be provided. To make use of this property, user need to have following properties declared at the grid or list level. 1. freezColumn 2. leftPanelWidth 3. rightPanelWidth 4. columnWidths 5. width 6. height 7. headerHeight 8. rowHeight

Name	Type	Mandatory	default	Explanation
				If the user declares a column name as freezColumn it means, till that column, grid or list will be fixed. After that, right panel will get the horizontal scroll bar.
leftPanelWidth	string	no	-	If the freezColumn is declared, then leftPanelWidth has to be declared.
rightPanelWidth	string	no	-	If the freezColumn is declared, then rightPanelWidth has to be declared.
simulateClickOnRow	string	no	-	It gives the option to select rows on load of the list data. There are five options. 1. None : No row will be selected 2. First : First row will be selected 3. Current: Current row will be selected. 4. Next: Next row will be selected 6. Last: Last row of the grid will be selected.
treeViewColumnName	string	no	-	All these 4 attributes are to be used together. The “treeViewColumnName” indicates which column is going to be the one that is having the treeView. The “treeViewKeyColumn” indicates the column that contains the key of the node that is going to be present in the tree. “treeViewParentKeyColumn” indicates the parent key of the current child. “treeViewHasChildColumn” indicates if the current node is a parent node or a child node.
treeViewKeyColumn	string	no	-	
treeViewParentKeyColumn	string	no	-	
treeViewHasChildColumn	string	no	-	
headerHeight	string	no	-	Header row height will be specified.
rowHeight				We can specify the height for the row in the grid or list.
localPagination	Boolean	No	-	If this attribute defined with true then all data for the table will be cached at the client even we have pagination defined. Use: 1. If you want to perform client -side filtering so get all data at once and do client filtering. 2. If you search detail kind of requirement.

3.5 gridPanel

gridPanel is a collection of zero to any number of fields/elements. It’s used to represent a data grid. Unlike listPanel, gridPanel can have input fields. Following attributes are defined for a gridPanel. We can add a row when we tab out of the last input field of the row inside a grid. But this is optional. If the application team want to have add rows on tab out then they need to mention lastKeyEventTrigger="true" applicationParameters.xml and in exilityParameters.js file they need to mention exilParms.doNotAddRowsOnTabOut = false; If the application team does not want have add rows on tab out then they need mention lastKeyEventTrigger="false" applicationParameters.xml and in exilityParameters.js file they need to mention exilParms.doNotAddRowsOnTabOut = true; If we want to navigate across the rows then we can use alt + up or down arrow key.

Name	Type	Mandatory	default	Explanation
Panel attributes				Refer to basic panel attributes.
tableName	string	yes	yes	Name of the table/grid inside the dc in which data is fetched/sent for this grid.
rowsCanBeAdded	boolean	yes	yes	Can the user add a row at run time?. If yes, a button for this sake is generated by Exility. Also, a tab key pressed on the last field of the last row automatically adds a row.
functionBeforeAddRow	String	No	-	Do you want a local script function to be executed before adding a row? Specify the name of the function. You should define this function in your JavaScript file.
functionAfterAddRow	String	No	-	See above
rowsCanBeDeleted	boolean	yes	yes	Can the user delete a row at run time? If so, a check box/icon is provided to mark a row for deletion. An existing row that is marked for deletion is rendered in different color as an indication. However, if a user deletes a row that she had just added, that row is removed from display. Use can undelete a row that is marked for deletion.

Name	Type	Mandatory	default	Explanation
				Note: a row that is marked for deletion, and subsequently undeleted will be marked as 'modified' and will be sent to server for modification even if no data in the row could be modified by the user.
idFieldName	string	No	No	What is the internal unique key with which a row can be identified? Most designers use an internally generated one-up number for this, and do not display this to user. In such a case, this field would be a hidden field. Exility uses this field to determine whether a row as fetched from the server or added locally.
keyFieldName	string	No	No	what is that all important field that a user sees as the key on the grid? This has relevance only if rows can be added. If user adds a row, but leaves all fields empty, then she does not expect us to add that row. keyField is used as THE field that determines if she entered data to this row or not.
actionFieldName	string	No	No	This is the name of the filed that keeps track of what happened to the row. This is required if you use a bulk action on the server. Value in this field could be empty, (nothing happened) or add, modify, or delete. The value in the field is automatically set by Exility based on the activity on the row. However, you MUST define this field as one of the fields within the panel, and you better define it as a hidden field. Note: Exility assumes that you have hidden this field, and hence does a small optimization of not populating the Dom element with the value, though it internally keeps it.
uniqueColumns	string	No	No	Is there one or more columns, whose values should be unique across all rows? If two columns together have to be unique, specify them as comma separated fields. In case

Name	Type	Mandatory	default	Explanation

				you have more than one such constraint, put then semicolon separated.
				for example:
				uniqueColumns = "col1,col2;col5,col6,col9"

minRows	number	No	No	Minimum number of rows of the panel at any point of time. Using 'deleteRowButton' rows cannot be deleted if current rows go below `minRows`.
maxRows	number	No	No	Maximum number of rows of the panel.
initialNumberOfRows	Number	No	-	Number of rows to be displayed on form load
sendAffectedRowsOnly				
repeatingColumnName	String	No	-	See sub-section on repeating column
childTableName	string	No	-	See sub-section on repeating column
childKeysTableName	string	no	-	See sub-section on repeating column
allColumnsAreFilterable	Boolean	no	-	If the user declares this attribute at grid or list level, all the columns will be provided with the filter option.
allColumnsAreSortable	Boolean	no	-	If the user declares this attribute at grid or list level, all the columns will be provided with the sort option.
columnSumCssClassName	String	no	-	User can have the different CSS for footer.
dataForNewRowToBeClonedFromFirstRow	Boolean	no	-	When we click on the add row or clone row button, data will be copied to new row from the first row of the grid.
dataForNewRowToBeClonedFromRow	Boolean	no	-	We can specify from which row data should be copied to the newly added row. 1. None : Data will not be copied 2. First : First row will be copied from first row 3. Current: Copied from the current row. 4. Next: Copied from the immediate next row 6. Last: Will be copied form the last row.
deleteRowsImmediately	String	no	-	Row will be deleted automatically even if ID is there.
freezeColumn	string	no	-	A grid or list can be split into two parts. First part will be fixed and for the second part of the grid or list, horizontal scroll bar will be provided. To make use of this property,

Name	Type	Mandatory	default	Explanation
				user need to have following properties declared at the grid or list level. 1. freezColumn 2. leftPanelWidth 3. rightPanelWidth 4. columnWidths 5. width 6. height 7. headerHeight 8. rowHeight If the user declares a column name as freezColumn it means, till that column, grid or list will be fixed. After that, right panel will get the horizontal scroll bar.
functionAfterDeleteRow	string	no	-	Local action can be called after deleting the row.
functionBeforeDeleteRow	string	no	-	Local action can be called before deleting the row.
headerHeight	string	no	-	Header row height will be specified.
labelForAddRowButton	string	no	-	User can specify his own label for add row button.
labelForBulkDeleteCheckBox	string	no	-	User can specify his own label for bulk delete check box.
leftPanelWidth	string	no	-	If the freezColumn is declared, then leftPanelWidth has to be declared.
newRowColumnsNotToBePopulatedWithData	string	no	-	User can mention the column names which he doesn’t want to copy from the parent row to cloned row when clone row happens.
rightPanelWidth	string	no	-	If the freezColumn is declared, then rightPanelWidth has to be declared.
rowsCanBeCloned	Boolean	no	-	One new row will be created with all the data copied from the parent row.
confirmOnRowDelete	Boolean	No	False	If the user needs confirmation message on delete, then he can give this option as true.
RowHeight	Integer	No		Height for the grid can be specified here.
isRequired	Boolean	No		If we want to show * for mandatory fields in the grid, then

Name	Type	Mandatory	default	Explanation
				we have to specify a parameter in ApplicationParameter.xml i.e. “showRequiredLabelinGrid = true”. When * appears on the grid headers, alignment of the grid might be varied. Applications are requested align their grid so that alignment is proper.
autoSaveServiceName	String	No		Valid when you have linked panel, and you want a row to be saved as and when it is modified. This service will be called with a grid with the modified row in it.

3.6 Tab Panel

The tabPanel is designed to display several panels in a tabbed manner. That is only one panel is visible at any time, and the panel can be viewed by clicking on its tab. It is desirable to set the width of a tabbed panel. If not set, it may change based on the selected panel. The UI effect is not pleasant. All the panels under a tab panel should have comparable size (height as well as width) for the tabbed panel to look good.

It is not advisable to have action buttons inside tabbed panels, except if you make that a standard, and each tab has its own button.

Tab panels will have round bottom edges. For page layout 3 only this is applicable (for other page layouts, the edges are square.).

Name	Type	Mandatory	default	Explanation
Panel attributes				Refer to basic panel attributes.
tabAreaHeight	string	No	Auto	Auto height is not desirable as it may lead to the page taking different shape in tab clicks. Specify this in absolute measures, like px, and not in %.
tabAreaWidth	string	No	Auto	Width of the tabPanel.

3.7 Spacer Panel

A text rows gap is maintained by default between two panels. If you need more gap, you can user a spacer panel with a specified height. Typically you specify just the **height**, (thru basic Element attribute) and no other attribute.

Name	Type	Mandatory	Explanation

Panel attributes			Refer to basic panel attributes.

3.8 Sub-window Panel

The subWindowPanel is used to render another page within a panel. The difference between display panel and sub-window panel is that the subwindow panel has its own page. That is, data that is fetched for the main window is not pushed to the subwindow. Subwindow's page will get its own data and manage, as it is an independent page. There would be some communication required between main window and sub window. Such facilities will be provided as and when you come up with such requirements.

A subwindow has to have its height and width. It is mandatory. There is no 'auto' height/width available in html for this. In other words, the size of the subwindow is fixed, irrespective of the size of the page it renders. Use htmlAttributes to set visual attributes like border, scrolling needs etc..

Name	Type	Mandatory	default	Explanation
Panel attributes				Refer to basic panel attributes.
Src	string	No	About:blank	url/path of the page to be displayed in this subWindowPanel. If source is not known at design time, leave it to the default. A blank page is displayed. You may set its ***src*** at any time during the execution of the page. Let Exility team know about your requirements to handle ***src***, and we will create declarative syntax for you to manage them.

3.9 Message Panel

The messagePanel is meant to display messages that are sent from server. The design of this depends on the project requirement. As of now, this is not used. Exility displays the message as an alert at this time. You may come up with your requirement for the message panel, and Exility will implement them.

3.10 Handling 'No Data' in Grid or List Panel

If you want to hide the regular grid/list and display something different, you have two options. You may set `messageNameOnNoData`="" a message name in you messages.xml. This message will be displayed instead of the regular panel. Advantage of this approach is that the page will be able to handle multiple languages at run time.

Alternately, you can create another panel with the name as tableName+NoData. Note that it is the tableName of the panel, and not the panel name itself. E.g. a panel with name=”myPanel” with tableName=”myTable” can have a corresponding messagePanel with name=”myTableNoData”.

3.11 Repeating Panel

Normally, you display rows of data; you display them as rows in a table. However, if the same set of data were to be just one row, you prefer to display that in a normal panel with pairs of label-field laid out rather than a header row and a data row. There are times when you may have more than one row of data, but you may want to display each row as if it were a normal display panel. i.e. each row is displayed in a (possibly collapsible) panel either laid out one below the other, or inside a tab panel.

Repeating panel is meant for such a situation. Any panel can be repeated. Note that the data input as well output to server will be in a grid, as if all the rows were part of one table. You can use even grid panel with addRow option. Exility takes care of adding the right key for each row.
To design repeating panel, you specify repeatOnFieldName=”” for the panel. This is the column name in the inkling data table that determines how many times the panel is to be repeated. Note that if the panel being repeated is a display panel, then this column value is unique. If panel being repeated is a grid panel or a list panel, grid is repeated for each distinct values of this column. In a repeating grid panel with option to add rows, this column value is set automatically for all the rows added in that specific instance of the grid.

If the panel has a label, the same column value is used as the label, except if you choose a different one specifically with labelFieldName=””

Also you should note the data requirements of repeating rows. The data table has to have all the columns for the designed fields as well as the one for the one based on which panels are repeated. Data table MUST have rows grouped by this column. That is all rows with the same value of this column must be grouped together. (Simple way is to just have the table sorted on this column).

With such a design, repeating panel is a UI feature, with no implication for the server side. Server side of the application gets to send a regular data table, and will get back such a one.

Refer to repeatingPanels.xml in sample pages supplied by Exility.
<< example .xml, data and html to be inserted here >>

3.12 Repeating Column in Grid or List Panel

Sometime, you need a list or grid panel with one of the columns that may repeat based on run time data. For example you may want to show booked quantity for each of the possible sizes for a style. You may want to show scheduled numbers for each of the days in the selected time frame. Here a specific column (quantity by size of scheduled number for day etc...) repeats itself, but the number of times it repeats is not known at design time.

Refer to 'repeatingColumn.xml' example supplied with Exility.

Specify the grid/list as you would do if the column were not repeating. For the grid/list panel, specify the following attribute.
repeatingColumnName: Note that you would have defined this as a column (field for the panel) the same way that you define any other column. This attribute is in addition to that definition.

At run time, you will get data in two or three grids for such an arrangement.

1. Header table. This is the table where you get data for all columns except the one that repeats. Name of this table is what you specify as tableName for the panel. Note that you MUST specify idFieldName for such a table.
2. ChildTableName. This table has the following columns
    a. idField of the parent table.
    b. Key of this row. For example if it is qty by size, then this is the internal key of size.
    c. Description of above key. This column is optional if a third table is provided. This field is what is used as the column heading for the repeated column.
    d. Value that is displayed in the repeated column.
3. keysTable: this table has all possible repetition of the column. For example it has all possible sizes across all rows. First column is the internal key value and the second one is what is displayed. For example you may have sizes in a coded table with a number as the key and a code as its value (1=Small, 2=medium etc...) If this table is not supplied, then the possible list is computed based on distinct values in childTable.

<<Example .xml, example data and display to be inserted here >>

3.13 htmlPanel

Sometime, we need to have option where we can define our own HTML pages and render that HTML inside our page. In this case we can make use of this property.

Name	Type	Mandatory	default	Explanation
fileName	string	yes		Through this path we are taking HTML page and rendering it inside our page
Name	string	No		Name of the panel.

3.14 includePanel

It is not necessary that we have to declare all the panels inside a single page. We can have multiple pages where different panels will be declared.

Then in one of the pages we can include all those panels and we can generate the pages. This feature can be implemented using includePanel attribute. The pages which we are including through includePanel should not contain Page tag.

Name	Type	Mandatory	default	Explanation
panelNameToBeIncluded	string	yes		We are mentioning the filename which contains the desired panel.
Name	string	No		Name of the panel.

3.15 spacerPanel

If we want space between two panels then we can make use of spacer panel. No attributes are required for spacerPanel.

3.16 xmlTreePanel

Name	Type	Mandatory	default	Explanation
Panel attributes				Refer to basic panel attributes.
fieldName	string	Yes		This field contains the path of the xml file.
expandAllOnLoad	Boolean	No	False	When we give `expandAllOnLoad=true`, all the values will be shown with the tree structure
showValues	Boolean	No	False	When we give `showValues = true`, node value will be shown along with node name.

XML treePanel can be loaded from the database and it can be loaded from xml file also.

The usage is as follows.
1. xmltreepanel should have the value for the attribute "fieldName" in page.xml
2. One of the server actions should return the grid "fieldName" with the following columns in the same order
   a. id of the parent node
   b. id of the node
   c. text to be displayed for the node
   d. flat to indicate whether the node is a parent node or not (valid values are 'y', 'Y', 'n', 'N')
   e. link to the page for navigation
3. The page should include a JavaScript which contain the function with name 'menuNavigateToPage' and should accept the link 'url' specified in sub-point 'e' of point '2' as its parameter. This function should be written by the application team for navigating to different pages.

On click of parent node also, the same function "menuNavigateToPage" will be triggered. So, application team can use appropriate links at parent and child levels to navigate to different pages. Some of the following features like,
1) Mouse down
2) Mouse Up
3) onmouse over
4) onmouse out

5) Double click
6) right click can be implemented as follows. This can be used by specifying the "childHtmlAttributes" in the page.xml on the "xmlTreePanel" element.

```
<xmlTreePanel name="xmlPanel"
                      bulkCheck="true"
                      slideEffect="fromLeft"
                      width="300px"
                      height="300px"
                      label="XmlTree"
                      fieldName="xmlpath"
                      expandAllOnLoad="true"
                      showValues="true"
onClickActionName="testingAction"
childHtmlAttributes="onmousedown=alert(this);
onmouseup=alert('up');"/>
```

Like this, any of the html attributes can be provided on all the child nodes.

PS: Please make sure that the "double quotes" are not added for the function call.

XML tree has been used to build the menu of the application. When we click on any menu item, previous page is not going off. Hence the new page is not getting loaded. When we click on any XML tree element we are clearing any of the pages. Hence this issue is arising. Only way we can fix this issue is, we should not store the page in cache. If we don't have any pages in cache then page should load on click of the page. One way we can achieve the same is by setting cache size equal to zero in the browser settings. By doing so, end user has to change this setting every time. Instead of this method, we can add meta tag in html, <meta http-equiv="Cache-Control" content="no-cache"/>. But if we introduce this tag, then there is a possibility of performance overhead as each and every time page has to reload. So we have provided an application parameter level option, httpNoCacheTagRequires="true". If you set this tag then this this tag will be added.

4. Static Element

All static elements have attributes as defined in the table “Basic element attributes”. Some elements may have additional attributes. This section explains elements and the additional attributes they may have.

Page elements are used to render something on the page, but no data binding happens to them. However actions can be attached to the elements that triggers on click of the element. Following types of element are defined for the page.

1. buttonElement
2. breakElement
3. formElement
4. htmlElement
5. spacerElement
6. staticImageElement
7. textElement

4.1 Button Element

Those of you know about html form, know that button is a form field. We have categorized it as an element, because the button neither changes based on data, nor does it send any data to the server because of the way Exility works. (In normal html, button does send data to server, and hence it is apt in that context to be called a field.) In Exility, a button is implemented as ‘static element’ with its rendering details known at design time. onClickActionName is required for a buttonElement, without which is not a button ☺. You do not need any more attributes than the basic element attributes to define a button

Name	Type	Mandatory	Default	Explanation
Element attributes				Refer to basic element attributes.
imageName	string	No		Image to be displayed in the button; the gif address to be provided from the current base of the Htm file.
isDefaultButton	Boolean	No		In each page there can be one default button which gets the “click” on

Name	Type	Mandatory	Default	Explanation
				hitting “Enter” key
whatToDoOnFormChange	String	No	leaveMeAlone	If you set trackChanges to true at the page level, the form state changes from ‘clean’ to ‘dirty’ the moment user changes any one field. Do you want to enable/disable this button when this happens? For example, you may have submit and cancel button disabled and close button enabled on page load. When user changes any field, you may want to disable close, but enable submit and cancel.

4.2 Text Element

The textElement is used to render some fixed text on the page. Use cssClassName for this text to be rendered the way you want. This element does not require any additional attributes.

Name	Type	Mandatory	Default	Explanation
Element attributes				Refer to basic element attributes.

4.3 Static Image Element

Why are we using the word staticImageElement and not just imageElement. That is because we have another field called imageField. To stress the fact that this image element has no dynamic data behavior, we named it that way. A static image element is designed like the classic img element of html. You specify the src, the way its border is to be considered, and you are done. Then what is image field? Well wait till we reach there.

Name	Type	Mandatory	Default	Explanation
Element attributes				Refer to basic element attributes.
Src	string	Yes	-	Image file name with path for the element.
Border	string	No	none	Width of the border to draw around the image. Defaults to having no border.

4.4 Spacer Element

The spacerElement is used to have some areas unused in a panel lay out. For example, if you want to keep the second row of a panel empty and render the next field as the first element in the next row, you can throw a spacerElement there. Spacer element uses a space character. However, if you need to control the width and height in a better way, you can use an image instead. Exility provides a single pixel image by name `spacer.gif`. However, if you want to use your image, use the spacerImage attribute. Note that spacer element does not expose all attributes of a basic element, but has limited attributes as follows.

Name	Type	Mandatory	Default	Explanation
numberOfUnitsToUse	number	No	1	As in element
spacerImage	String	No	-	Name of the image file to be used as background for this space.
Width	String	No	-	Width of this element.

4.5 Break Element

Elements by default are rendered in-line, one in front of the other within the width available. Browser may decide to wrap if the width is not adequate. However, if you do want to render the next element on the next line, use breakElement. No attributes are allowed. It simply translates to a <br /> html element.

4.6 htmlElement

Exility generates an html based on the xml you have provided. It is designed to provide a simple, declarative way of expressing your page design. It is not cluttered with details of an html implementation. While it takes care of most of the requirements, it is not ‘complete’ in all respect. If there are html features that you want to use in addition to what Exility is providing, you can use htmlElement to directly insert a syntactically correct html text into the generated page.

Name	Type	Mandatory	Explanation
htmlText	string	yes	A syntactically correct html text. Note that the text has to be XML escaped. Example a double quote is represented by &quot;. Use CDATA section to avoid such unreadable text.

5. Field

Field is a page element that can receive data or send data. It may or may not have a displayable attribute. For example a hidden field can receive and send data, but is not displayed on the page.

We have started with the normal fields that a browser provides in an html page. However, we intend to provide more user-friendly fields in the future.

Exility lays out fields in such a way that the labels are aligned by default. To achieve this, it relies on table, tr and td elements of html. It is better that you know about how this works, so that you are in a good position to use some of the flexibilities provided to you.

Let us say that you have a display panel with numberOfUnitsPerRow = 2. That is, two fields per row. The display requirement is that there are two columns of fields; each column has two ‘elements’ - label and field. For aesthetic reasons, the labels across all the rows are to be aligned. To achieve this, Exility opens one td for the label, and another for field. That is, in a tr, there are four tds. So, one unit is two td. Exility provides the flexibility to lay a given field or set of fields differently, by allowing you to specify what to do with the label, how many units a field should take as explained in “basic element attributes”.

Validation:

All input fields are automatically validated by Exility. The data type of the dataElement that is associated with the field is used for validation. In addition to this you can do the following validation on the client:

1. Must Enter /Required field.
2. Conditional must enter : A field is a must enter, if another field has non-empty, non-zero value
3. From-to validation : a field value cannot be less than another
4. unique values in a column : one or more columns together has to be unique across all rows of a grid.

In addition to these client side validations, Exility also allows these server-side assisted validations.

1. value restricted to a drop-down – User has to choose one of the values from the drop-down/selection list.
2. Existing code – When the value to e entered has to exist in the data base: For example a customer code. Such a validation is achieved through description service.
3. New unique code - User has to enter a code that is not there in the data base. This validation is also achieved thru description service.

Description Service and Code Picker:

Primary objective of a description service is to validate a code that is entered by the user, and get the corresponding name/description if the value exists. Typically, you use this when the list of valid values is not small, and hence a drop-down box is not suitable. Experienced users may know the code, in which case, the code they have entered can be immediately validated and the name/description displayed immediately, so that the user can verify that the code entered is the right one. (An entered code may be valid, but not the one that the user intended).

It would be convenient if we provide flexibility for users either to type such a code, or allow selection of such a code through a search operation. Code Picker provides such a facility. You can use an appropriate search page as the pop-up page that come on when user clicks on an icon next to the field. User can use the page to select the required record based on the information she has. Use codePickerSrc attribute of attached to an input field for this.

Description service can be used to get any number of fields. Its advantage is that it is aware of the grid row. That is, all the fields fetched are delivered to the same row as the triggering field, if they were to be in a grid. With this additional advantage, attributes related to description are available to selection fields also

.

Description Service and Code picker must be used together to provide flexibility to the user.

An alternative to this approach is the combo-box. Details about this will be provided in the next version.

The common attributes of the “Field” are explained below:

5.1 Field Attributes

Name	Type	Mandatory	Default	Explanation
Element attributes				Refer to basic element attributes.
labelPosition	string	No	right	How do you want the label to be displayed? Default is right, as labels are better aligned right. Some projects use left justified. Label on top gives look of a row of label and a row of field. Hide means you do not want to display the label, but the td that is meant for the label should be left as it is. Merge means that the two tds, (one for label, and one for field) are merged into one cell, and label (if any) and the field are rendered next to each other.
defaultValue	string	No	-	Default value of the field if no value is set.
technicalDescription	string	No	No	Technical description of the field in the page for the designer.
businessDescription	string	No	No	Description of the field in the page from business user perspective.
dataElementName	string	No	No	By default the field name is same as the data dictionary entry name. If the designer wants to the field to refer to some other data dictionary entry, it’s specified through dataElementName attribute.
isRequired	boolean	No	No	If true, the user MUST enter some value for this field.
altKey	char	No	-	Short cut key for the field. E.g. “k” means alt+k will get the focus to this field. ‘k’ in the label is also highlighted. This feature is yet to be implemented, but you can go ahead and design this feature.
breakToNextLine	boolean	No	False	Field is rendered in the next line. Useful if you want to accommodate more than one field in a table cell, and you want one to appear below the other. That is, you would like fields to be rendered in more than one lines within a row.
aliasName	boolean	No	False	Do you have a need to send two fields to the server, but their values have to be the same? When this field is submitted, two fields, one with

Name	Type	Mandatory	Default	Explanation
				name and other with aliasName go to the server with same value in both.
breakToNextLine	boolean	No	False	Field is rendered in the next line. Useful if you want to accommodate more than one field in a table cell, and you want one to appear below the other.
rowSum	boolean	No	No	This attribute is applicable only if the field is inside a list/gridpanel. when rowSum is set to "true" for a field, sum of all rows for that particular column is rendered in a footer row.
rowAverage	boolean	No	No	This attribute is applicable only if the field is inside a list/gridpanel. when rowAverage is set to "true" for a field, average of all rows for that particular column is rendered in a footer row.

columnSum	string	No	No	This attribute is applicable only if the field is inside a list/gridpanel.
				when colSum are set for some fields, sum of all those columns for each row would be rendered in an additional column in the table. These additional col elements need to be specified in the page xml.
				The value of colSum should be same as the name of the sum col element.
				Also for the sum col element, doNotReceiveData="true" need to be set.
				Example:
				(page xml extract for colSum in a grid)
				`<inputField name="buyerName"  label="Buyer Style Name"  colSum="sum1" />`
				`<inputField name="quantity"  label="Quantity" rowSum="true"  colSum="sum1"/>`
				`<outputField name="sum1"  doNotReceiveData="true"  label="Sum"/>`

columnAverage	string	No	No	This attribute is applicable only if the field is inside a list/gridpanel.

Name	Type	Mandatory	Default	Explanation

				when colAverage are set for some fields, average of all those columns for each row would be rendered in an additional column in the table. These additional col elements need to be specified in the page xml.
				The value of colAverage should be same as the name of the average col element.
				Also for the average col element, doNotReceiveData="true" need to be set.
				
				Example:
				(page xml extract for colAverage in a grid)
				
				`<inputField name="buyerName"  label="Buyer Style Name"  colAverage="avg1" />`
				`<inputField name="quantity"  label="Quantity" rowSum="true"`
				`colAverage="avg1"/>`
				`<outputField name="avg1"  doNotReceiveData="true"  label="Average"/>`
				
				Column sum feature cannot be overridden for average calculations. These are present as attributes of grid elements. A JavaScript function can be written, which should be included as the script at page level. User has to provide the function name as value for the aggregate function. This local function will take entire grid as the parameter. In all the cases the function should return a value. In case of a columnSum or columnAverage on a repeating field, the function should return an array of values one for each of the repeated instances.

Formatter	String	No	-	lcase -> convert text to lower case ucase -> convert text to upper case inr -> for number field: use commas as in lacs etc.. but no decimal places. e.g. 34,45,678 inr2 -> have two decimal places, in addition to the commas as explained above usd, usd2 -> similar to inr and inr2 but comma for millions etc..

Name	Type	Mandatory	Default	Explanation
globalFieldName	String	No		
globalFieldLabelName	String	No		
isFilterable	Boolean	No	False	Rows can be filtered at the list or grid or field level. Field should be part of the grid or the list. It can be specified at grid or list or field level. If we mention at grid or list level it will be applied to all the fields in the grid or list. To make this option work, we need to declare FilterArea div in the application home.
isSortable	Boolean	No	False	Data can be sorted at the grid, field or list level. Field should be part of the grid or the list. If we mention at grid or list level it will be applied to all the fields in the grid or list.
maxRowsToDisplay	Int	No		How many rows of data user wants to show for the field/
onBlurActionName	String	No		When the focus is out of the field, an action can be called for the field.
onFocusActionName	String	No		When the user focusses on the field, an action can be called
onUserChangeActionName	String	No		This action will be called only when user changes something for the particular field.
otherField				
supressDescOnLoad	Boolean	No	False	When the page is loaded, description service will not be called automatically for the particular field. It will be called only if user changes something
validateOnlyOnUserChange	Boolean	No	False	Validate for the field only if the user explicitly changes the value for the field.
comboDisplayFields				List of fields to be displayed in combo. The fields must be specified as comma separated.

5.2 Input Field attributes

In addition to the field attributes, following attributes are common to all input fields.

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
onChangeActionName	string	No	Action to be taken on change of this field. This is triggered when the field changes, either by user, or when a value is changed internally. Values are changed internally either

Name	Type	Mandatory	Explanation
			based on other fields, or when data is fetched from the server
onUserChangeActionName	string	No	Action to be taken when user changes the field. Note that this is triggered after the user gets out of the field, and not when the user is still editing the field.
basedOnField	string	No	By default, user need not enter value in this field. But if the 'basedOnField' has value, then this field also should have value.
basedOnFieldValue	string	No	Same as above, but only if the field has this specific value. Note that this attribute is meaningful only if you use the previous attribute also.
copyTo	string	No	Name of the other field that is populated with the same value as of this field, when this field value is changed.
doNotValidate	boolean	No	If true, this field bypasses the validation test.
isLocalField	boolean	No	This field is used locally by the client, and is not to be sent to the server. This field may still get data from server, but its value is not sent to the server.
dependentSelectionField	string	No	Another selection field, whose list of possible values would depend on this field. For example, if this field is country code, another field that gives list of states would be dependent on this. In such a case, stateCode is a dependent SelectionField of countryCode. You may have more than one dependent field, in which case, specify them separated by comma. Attribute name is not plural because of historic reasons. (You will see this in several other attributes also)
codePickerSrc	string	No	This is the name of the page that is to be used as code picker for this field. A code picker is a user aid to select a code, rather than typing one. You need a code picker if the field has a finite set of valid values (like customer number) but the number of such entries too large to be shown in a drop-down. Also, if a user knows the code, she may type it, else click on the picker icon to select from a search page. Page name is to be specified as folder/paegName.htm. This facility is to be used along with description service. (refer to note on code-picker)
descServiceId	string	No	Name of the description service that is used to validate the entered value and get the description/name for the entered value/code.
descFields	string	No	Comma separated field names that are to be populated when the description service returns value.

Name	Type	Mandatory	Explanation
descQueryFields	string	No	Query fields to be sent along with the request for the description service.
descQueryFieldSources	string	No	Field names from which values for descQueryFields to be assigned from. Ensure that the names are in the right order. descQueryFieldSource name should be present in dictionary
doNotMatchDescNames	boolean	No	When the description service returns, fields are assigned values based on field names. When you try to reuse a service across similar needs, the field names may not match, but they may be in the same order. Use this attribute to assign values in the order of their names, but not by name.
validateQueryFields	boolean	No	By default, query fields are not validated before they are used. You can change it using this attribute.
isUniqueField	boolean	No	This is to be used only when the field has description service. Its meaning is that the value to be entered in this field has to be unique, and should not exist already. Attached description service is then used to validate whether the entry already exists. User-name for a new registered user is a good example of this feature. In such a case you want the user to type a value that does not exist. You define the descServiceId that tries to get an existing entry. You may notice that specifying 'isUniqueField' flips the validation. That is if this is false, the entered value MUST exist. If this field is true, then the entered value MUST NOT exists.
toField	string	No	If this field is used as part of a range, toField is the name of the field that specifies the upper limit.
fromField	string	No	If this field is used as part of a range, fromField is the name of the field that specifies the lower limit. If the fromField and toField validation messages has to be different then user has to provide `exilParms.diffToFrom = true;` in exilityParameters.js file.
Size	number	No	Size of this field in the html page.
minCharsToTriggerService	Number	No	Specify this if you need combo-box behavior to this field. Note that this is relevant only if you have specified descServiceId. Once the user types min chars, desc service is used to fetch list of valid values and offered in a drop-down list.
columnSumFunction, columnAverageFunction,	string	No	These are present as attributes of grid elements. A JavaScript function can be written, which should be included as the script at page level. User has to provide the

Name	Type	Mandatory	Explanation
rowSumFunction, rowAverageFunction			function name as value for the aggregate function. This local function will take entire grid as the parameter. In all the cases the function should return a value. In case of a columnSum or columnAverage on a repeating field, the function should return an array of values one for each of the repeated instances.
fieldToFocusAfterExecution	String	No	For a field element "fieldToFocusAfterExecution" has to be used in combination with "descServiceId". This attribute can be provided with the name of the field to be focused as value. The field that gets focus cannot be a part of list or grid

5.3 checkBoxField

Following attributes are defined for the checkBoxField.

Name	Type	Mandatory	Explanation
input field attributes			Refer to basic input field attributes.
checkedValueIsTheDefault	boolean	yes	If true, the field is checked by default when the page is rendered.
checkedValue	string	No	When checked, checkedValue is the value of the field.
uncheckedValue	string	No	When unchecked, uncheckedValue is the value of the field.

5.4 filterField

A filter filed is a misnomer. It is a group of fields that help user to specify selection criterion, or filtering conditions. For example you want all customers, where the credit limits is in between 10,000 to 20,000. Filter field provides a convenient way to implement such features on both client side and server side. Note that the filter field is not valid for Boolean valued fields. If the mandatory option is required for BETWEEN option of a date field then we have to set “exilParms.validateDatesForBetween = true” in Exility parameters.js file.

Name	Type	Mandatory	Explanation
input field attributes			Refer to basic input field attributes.
defaultComparisonType	string	No	Default comparison type selected when the page is rendered.
Size	number	No	Size of this field in the html page.

5.5 hiddenField

The hiddenFields are used to store some value that the user does not want to render on the page but at the same time wants to use the value to build some logic at page level for rendering/submitting the data.
The following attributes are defined for the hiddenField.

Name	Type	Mandatory	Explanation
input field attributes			Refer to basic input field attributes.
numberOfUnitsToUse			Defaults to 0. For all other fields, default is 1. While this convention is convenient, it may lead to a problem if you put a hidden field as the first field of a panel. Always ensure that hidden field comes after specifying at least one field with numberOfUnitsToUse > 0.

5.6 imageField

An image field allows an image to be displayed at run time based on some data value. For example you want to show the image of the employee. You would have stored all images with a naming convention, say empId.jpg. You need to set the src of an image control to such a value at run time. imageField is the right control for such situations. You may put a place holder in the src, and choose a field name from which to get the value at run time and put that in the place holder.

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
Border	String	No	Width of the border to draw around the image.
baseSrc	String	No	Path of the folder in which the image resides.
imageExtension	String	No	Extension of the image (e.g. gif, jpg etc.)

5.7 outputField

The outputFields are used to render text output data on the page. Following attributes are defined for the outputField. The outputField is different from textElement. The Output data can be bound to an outputField but not to a textElement. The following attributes are defined for the outputField.

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
maxCharacters	number	No	Maximum characters allowed in the output field.
toBeSentToServer	boolean	No	When form is submitted, this field is returned to the server

5.8 radioButtonField

The radioButtonField is used when the user wants to select only one value out of many options. The following attributes are defined for the radioButtonField.

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
valueList	string	No	Specifies semicolon separated list of options where each option is comma separated value and label. Example: `valueList="1,Option1;2,Option2;3,Option3"`

5.9 selectionField

The selectionField is used when the user wants to select one or more options out of many options. The following attributes are defined for the selectionField.

Name	Type	Mandatory	Explanation
input field attributes			Refer to basic input field attributes.
listServiceId	String	No	Name of the list service to fetch the list options.
listServiceQueryFields	String	No	Fields to be sent along with the serviceId
listServiceQueryFieldSources	String	No	Names of fields whose value is used for the fields mentioned above
keyValue	String	No	The parameter with which the list service is invoked. By parameter passing, same list service can be used with different fields using different keyValues.
blankOption	String	No	String to be displayed when no option is selected.
noAutoLoad	Boolean	No	If noAutoLoad is false and listServiceId is specified, then list service is invoked on page load and dropdown options are populated. If noAutoLoad is true, then dropdown options are not populated on page load.
valueList	String	No	Specifies semicolon separated list of options where each option is comma separated value and label. Example: valueList="1,Option1;2,Option2;3,Option3"
Size	Number	No	Size of this field in the html page.
multipleSelection	Boolean	No	Whether multiple option selection is allowed.
selectFirstOption	Boolean	No	If true, first option is selected by default.
dependentSelectionField	String	No	Name of the other selection field that depends on this selection field. On change of this selection field, the dependent selection field is invoked with its own listServiceId and keyValue equals to the value of this field.
sameListForAllRows	Boolean	No	When this field is inside a grid, should they all have the same list of options, or will they be different? This is used when this field is a dependent on another field. Default behavior is that only the current row is affected. Set this attribute to true to apply that list of values to all the rows.

5.10 textAreaField

The textAreaField is used when the user wants to enter multi line text. The following attributes are defined for the textAreaField.

Name	Type	Mandatory	Explanation
input field attributes			Refer to basic input field attributes.
numberOfRows	number	yes	Text area height: number of rows.
numberOfCharactersPerRow	number	yes	Text area width: number of characters per row.
isProtected	boolean	No	Whether the user can edit this field.

5.11 textInputField

The textInputField is used to accept character input that is typed by the user. The following attributes are defined for the textInputField.
If a textinput field is date field, then that value can be right aligned. The CSS needs to be updated as follows.
1. create a copy of the class "inputfield"
2. rename the copied instance as "dateinputfield"
3. provide the alignment property for this class name as needed.

Name	Type	Mandatory	Explanation
input field attributes			Refer to basic input field attributes.
toField	string	No	If this field is used as part of a range, toField is the name of the field that specifies the upper limit.
fromField	string	No	If this field is used as part of a range, fromField is the name of the field that specifies the lower limit.
Size	number	No	Size of this field in the html page.
isProtected	boolean	No	Whether the user can edit this field.

5.12 booleanOutputField

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
trueValue	string	No	If this field is used as part of a range, toField is the name of the field that specifies the upper limit.

falseValue	string	No	If this field is used as part of a range, fromField is the name of the field that specifies the lower limit.
toBeSentToServer	boolean	No	When form is submitted, this field is returned to the server
allowHtmlFormattedText			
valueList	string	No	Specifies semicolon separated list of options where each option is comma separated value and label. Example: valueList="1,Option1;2,Option2;3,Option3"

5.13 chartField

Chart Fields are used to draw the charts in the page. There are some mandatory things which the user should follow while designing the pages. Chart field should not be hidden, when rendering the data for that chart field. User should always specify the height and width for the chart field at the design time itself.

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
reportServiceId	string	No	User can mention the service ID which brings the data for the specified chart field.
chartType	string	No	There are 5 types of graphs. 1.DEFAULT 2.PIE 3.SPEEDOMETER 4.BAR 5.RADAR 6. HORIZONTALSTACKED 7. RUNCHART 8. STACKED 9. HORIZONTALBAR 10. BUBBLE 11. SCATTER
Xaxiscolumn	string	No	The column name which the user should refer from the grid to render it on the X axis
Yaxiscolumn	string	No	The column name which the user should refer from the grid to render it on the Y axis. Earlier only X axis was the dynamic column. Now Y axis can also be loaded dynamically.

isMultiDataSet	Boolean	No	If the user wants to group by some column, then it is necessary that he should mention isMultiDataSet=”true”
Groupbycolumn	string	No	If the user is grouping the data based on some column, then he should mention the column name.
Xaxislabel	string	No	Label for the X axis.
Yaxislabel	string	No	Label for the Y axis
Xaxislabels	string	No	If the user wants to hard code X axis unit label, he can specify the X axis unit labels here.
Yaxislabels	string	No	If the user wants to hard code X axis unit label, he can specify the Y axis unit labels here.
Minx	string	No	Chart will start from this value on X axis. It is mandatory that if we give minx, miny has to be given and vice versa.
Maxx	string	No	Chart will end on this value on X axis. It is mandatory that if we give maxx, maxy has to be given and vice versa.
Miny	string	No	Chart will start from this value on Y axis.
Maxy	string	No	Chart will end on this value on Y axis.
noAutoLoad	Boolean	No	On load of the page data will not be loaded.
showLegend	Boolean	No	Legends will be shown for the chart. For bubble chart if set this attribute to false, value will be hidden on the bubble.
yaxislabelformatterid	String	No	If the Y axis labels are to be formatted, set this to 'inr', 'usd' or 'dem'
bubbleColumn	String	No	Bubble will be drawn based on this column.
bubbleradiusdenominator	String	No	Size of the bubble will be decided by the following column.

Whenever mouse moves over the data points of a chart field line, the value at that of point can be displayed on the right-bottom corner. The functionality is available in Bar and Line Charts only.

5.14 checkBoxGroupField

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
listServiceId	string	No	Name of the list service to fetch the list options.
listServiceQueryFieldNames	string	No	Fields to be sent along with the serviceId
listServiceQueryFieldSources	string	No	Names of fields whose value is used for the fields mentioned above

keyValue	string	No	The parameter with which the list service is invoked. By parameter passing, same list service can be used with different fields using different keyValues.
noAutoLoad	Boolean	No	If noAutoLoad is false and listServiceId is specified, then list service is invoked on page load and dropdown options are populated. If noAutoLoad is true, then dropdown options are not populated on page load.
valueList	string	No	Specifies semicolon separated list of options where each option is comma separated value and label. Example: valueList="1,Option1;2,Option2;3,Option3"
sameListForAllRows	Boolean	No	When this field is inside a grid, should they all have the same list of options, or will they be different? This is used when this field is a dependent on another field. Default behavior is that only the current row is affected. Set this attribute to true to apply that list of values to all the rows.
selectionValueType	string	No	It decides how should the values be sent/recd, it will be in three formats. 1. Grid 2. List 3 Text
minSelections	string	No	Min number of check boxes to be selected
maxSelections	string	No	max number of check boxes to be selected

5.15 fileField

This attribute is used where the user wants to upload the files.

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.

5.16 passwordField

This attribute will be used where the entered data should be in the binary format so that entered data cannot be visible for the normal user. This feature will be useful where the user wants to design some password fields.

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
Size	Number	No	This attribute will decide how many bytes of data this field can allow.

5.17 shadeInputField

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
defaultColor	String	No	Default color for the field.

5.18 shadeOutputField

Name	Type	Mandatory	Explanation
field attributes			Refer to basic field attributes.
defaultColor	String	No	Default color for the field.

5.19 comboBox

Name	Type	Mandatory	Explanation
input field attributes			Refer to basic field attributes.
toField	string	No	If this field is used as part of a range, toField is the name of the field that specifies the upper limit.
fromField	string	No	If this field is used as part of a range, fromField is the name of the field that specifies the lower limit.
Size	Number	No	This attribute will decide how many bytes of data this field can allow.
minCharsToTriggerService	Number	No	Specify this if you need combo-box behavior to this field. Note that this is relevant only if you have specified descServiceId. Once the user types min chars, desc service is used to fetch list of valid values and offered in a drop-down list.
maxRowsToDisplay	Number	No	Maximum number of rows of the panel.

5.20 AssistedInputField

This is the dynamic field which will be rendered based on data type and attributes defined.
For example if data type for this field name is date then it will behave as date field and date calendar will be suggested for selecting to date.

Some more examples:

1. If listService defined for this field then it will behave as selection field and all attributes from selection field are applicable.
2. If suggestionService defined for this field then it will behave as Google suggestion drop-down.

If none of the mentioned above defined, then it will behave as textInputField.

Name	Type	Mandatory	Explanation
input field attributes			Refer to basic field attributes.
suggestionServiceId	string	No	Name of the suggestion service to fetch the list options.

suggestionServiceFields	string	No	Fields to be sent along with the serviceId
suggestionServiceFieldSources	Number	No	Names of fields whose value is used for the fields mentioned above
suggestAfterMinChars	Number	No	Once the user types min chars, suggestion service is used to fetch list of valid values and offered in a drop-down list. 0(zero) is not valid, because, in that case, you can go in for listService rather than suggestionService
columnIndexesToShow	String	No	If you want to display more than one column as a nice table, use this comma separated, zero-based column indexes. You have to manage the appearance with your CSS settings
suggestionCss	String	No	This will be the CSS class assigned to the div that houses suggestions.
listServiceId	String	No	If listService is given then this behaves same as selectionField and all attributes inherits of selectionField.(refer selectionField)

Example:

How to use assisted input field in XML page.

```
<assistedInputField name="empCodes"
                            label="Emp Code List"
                            listServiceId="internal.getEmployeeCodes"/>
```

Explanation:
In above example since listService is defined for the assistedInputField so it will work as selection field.

6. Actions

Action describes what is to be done, either as a response to a user action, (like pressing a button, or changing a field), or to a system event (like server returning data).

6.1 Action attributes

Name	Type	Mandatory	Explanation
Name	String	Yes	Identifier for the action. Use this field to refer to the action in the xml
showPanels	String	No	Comma separated names of panels/elements/fields that are to be displayed when this action is executed.
hidePanels	String	No	Comma separated panel/field/element names to be hidden when this action is executed. Note that a panel is twisted or tabbed if possible. Else, it is hidden using the style display property of the element.
enableFields	String	No	Comma separated names of fields to be enabled before this action is taken.
disableFields	String	No	Comma separated names of fields to be disabled before this action is taken.
warnIfFormIsModified	boolean	No	User might have entered some values in the form before asking for this action, Should you go ahead this action just lie that or warn the user about loosing those changes and take a confirmation? You set this to true if the action you want to take would result in user's input being lost. Exility manages this automatically for close/reset actions.
resetFormModifiedState	boolean	No	State of the form is maintained if trackChanges is set to true at the page level. State is changed to 'dirty' or 'changed' the moment user changes any field. Should this state be reset to 'unchanged' after this action is taken?
fieldToFocusAfterExecution	String	No	For a field element "fieldToFocusAfterExecution" has to be used in combination with "descServiceId". This attribute can be provided with the name of the field to be focused as value. The field that gets focus cannot be a part of list or grid .

6.2 localAction

The localAction provides the mechanism to execute user defined JavaScript function.
The function is called with three parameters.

1. obj – Dom object that triggered the action
2. fieldname, - name of the field of which the obj is an instance. For a field that is not part of any grid, the field name is same as obj.id.
3. parameters – string value that you specify as part of this action definition (parameter attribute)

Name	Type	Mandatory	Explanation
Action attributes			Refer to the basic action attributes.
functionName	string	Yes	Name of the user defined function to be invoked as action. This function is called with

			the following parameters: 1. obj: the DOM object that triggered this action. “null” if the action is not associated with any object. 2. fieldName: name of field that triggered this action. “null” if the field is not associated with any field. 3. params: refer to the next field. For example if the action is onChange action for field aField and the function name is foo, and params=”abcd”, then foo(ele, ‘aField’, ‘abcd’) is called You may use a function with arguments, instead of just the function name. e.g. foo(‘magic’, 2). In such case, the function is invoked the way have specified it, and not with parameters mentioned above.
Parameter	string	No	Optional parameter with which the function is to be called

6.3 serverAction

The serverAction is an action on the server. A request is sent to the server for the specified service. You may choose to either wait for the response before allowing the user to interact with the page, or block it. You may choose another action to be executed when the server returns a response to you. By default, Exility pushes the data that has come from server to the page. You may choose to do your own action, either in addition to this, or instead of this.

Name	**Type**	**Mandatory**	**Explanation**
Action attributes			Refer to the basic action attributes.
serviceId	string	Yes	Name of the service invoked as action.
queryFieldNames	String	No	Comma separated list of field names that need to be passed as argument for this service request.
queryFieldSources	String	No	If the field names in the queryFieldNames are different from form field names, queryFieldSources tells the comma separated list of form fields whose value is passed to the service request.
validateQueryFields	Boolean	No	By default, query fields not validated, but you can change that using this attribute
submitForm	Boolean	No	If submitForm is true, entire page is validated before sending the service request. And all form fields are sent as argument to the service.
submitFields	string	No	Comma separated set of fields to be submitted instead of the entire page. Only these

Name	Type	Mandatory	Explanation
			fields are submitted. Note that the server action is not taken if any of the fields fail validation.
callBackActionName	String	No	Name of the action that is to be executed when the server action returns.
waitForResponse	Boolean	No	If set to true, user action is disabled for the page. User has to wait till the server returns. A time-out is triggered by Exility in case the server does not respond within specified time. You set this to true for submit/modify kind of actions. You set this to false if you do some additional data fetching during data entry.
toRefreshPage	string	No	To indicate if the data received from server needs to be pushed onto the page . This field can take the following indicators ➢ beforemyaction : push data to page BEFORE invoking callBackAction ➢ afterMyAction: push data to page AFTER invoking callBackAction ➢ none: Do NOT push data to page
atLeastOneFieldIsRequired	Boolean	No	If set to true, server action is not taken if value is not entered for any of the fields.
disableForm	Boolean	No	Should the form be disabled till this action returns?
callBackEvenOnError	Boolean	No	Should the call back action happen even if server returns with errors?

6.4 navigationAction

You want the action to shift to another page. You may be either done with this page, and start a new page in this window, or you may want the focus to shift another concurrent window.

Name	Type	Mandatory	Explanation
Action attributes			Refer to the basic action attributes.
pageToGo	string	Yes	URL of the page to go, as action.
windowToGo	String	No	The window frame in which the new page is rendered.
substituteValueFrom	String	No	Do you have some run time variable in the name of the page to go to? In that case, you insert #value# in the pageToGo attribute. Specify the name of the field, whose value is to be used to replace that token at run time. Note that this name could be either a field, or a global JavaScript variable you have defined.
queryFieldNames	string	No	Comma separated list of field names that need to be passed as page parameter for the page ‘pageToGo’.

queryFieldSources	String	No	If the field names in the queryFieldNames are different from form field names, queryFieldSources tells the comma separated list of form fields.
validateQueryFields	Boolean	No	By default, query fields not validated, but you can change that using this attribute
windowDisposal	String	Yes	➢ replace – replace the current page with the newpage. ➢ Popup – open a new window on top of the current window as a 'modal' window. That is, the opened window will stay on top till the user closes it. Current window is retained as it is, and the control returns to it once the user closes the opened window. ➢ retainState- Current window disappears and a new window appears with the new page. However, the current window is retained, and the control comes back to it once the user closes the opened window. This is similar to popup except that the current window is not visible in this case. ➢ reset – when the next page closes, current page is displayed, but is reset. That is the page is assumed to be called again with the same values of query string that it got when it was first invoked.
passDc	Boolean	No	Should the last set of data that is received from the server be passed to the next page. In that case, the next page will use it to refresh the page.

6.5 mailToAction

This action is used to send a mail to the mailing address.

Name	Type	Mandatory	Explanation
Action attributes			Refer to the basic action attributes.
mailto	string	Yes	Mail-id to which mail is sent as action.
substituteValueFrom	String	No	Refer to the attribute with the same name for navigationAction

6.6 closeAction

Current window is to be closed. A warnIfFormIsModified is automatically set to true. The control will go back to the one that was last not replaced. If there is no such window, control will go to home page.
You may use functionName as an attribute. This function is invoked before closing the window.

You should choose the label for the button that triggers this action as per the standards in your project.

6.7 resetAction

Traditionally HTML pages have reset option; Browser resets values in all fields to the value that was supplied when the page was open. That is before user changed them. In Exility, this meaning is bit difficult to implement, as the designer would have called one or more actions before the page is ready for the user.

ResetAction in Exility means restart the page from the time it was called, possibly with some query fields. After a reset, you should see a page that is just like the way it was displayed before user started interacting with it.

However, you can change this behavior by setting fieldsToReset=”all” in which case all fields are reset to their default values (as you have set at design time) or provide comma separated fields.

6.8 DownloadFileAction

This action can be used to download files through the given path. The user has to provide two parameters for the download action.

1. download_fileType: Type of the file which the user is downloading.
2. download_fileName: Name of the file which the user wants to download.

Name	Type	Mandatory	Explanation
Action attributes			Refer to the basic action attributes.

6.9 dummyAction

If the user does want to perform page validation without calling any page actions, he can use this action. Following actions can be performed here.

Name	Type	Mandatory	Explanation
disableFields	String	No	Comma separated names of fields to be enabled before this action is taken.
enableFields	String	No	Comma separated names of fields to be disabled before this action is taken.
hidePanels	String	No	Comma separated panel/field/element names to be hidden when this action is executed. Note that a panel is twisted or tabbed if possible. Else, it is hidden using the style display property of the element.
Name	String	Yes	Identifier for the action. Use this field to refer to the action in the xml
resetFormModifiedState	boolean	No	State of the form is maintained if trackChanges is set to true at the page level. State is changed to 'dirty' or 'changed' the moment user changes any field. Should this state be reset to 'unchanged' after this action is taken?
showPanels	String	No	Comma separated names of panels/elements/fields that are to be displayed when this action is executed.
warnIfFormIsModified	boolean	No	User might have entered some values in the form before asking for this action, Should you go ahead this action just lie that or warn the user about losing those changes and take a confirmation? You set this to true if the action you want to take would result in user's input being lost. Exility manages this automatically for close/reset actions.

6.10 SaveAsXlsAction

Data coming from the DC can be saved into a XL sheet.

Name	Type	Mandatory	Explanation
Action attributes			Refer to the basic action attributes.
reportName	String	Yes	Name of the XL sheet through which the user is saving the data.

6.11 UploadFileAction

This action can be used to upload files through the given path. User has to provide two parameters for the upload action. When the users are trying to upload the same files through different session then we should handle it so that it should not create any conflict with the filename.

UploadFileAction which is supported by Exility will upload the files to a folder which is mentioned applicationParameters.xml. (For ex: File Path is /UploadFiles) Files uploaded in this folder are temporary. After uploading the files to this directory, application teams have to take care of those uploaded files. When the user uploads a file to this temporary path, file name will be suffixed with the current session id through which file is getting uploaded. In dc, file field though which file is getting uploaded, will contain the value of the original file path where the file was being selected. During upload file operation one more variable will be added to dc and that variable name will be the file field name suffixed with ExilityFilePath. And this variable name will contain the value which is having the temporary file path after the file gets uploaded. Using these values, application team needs to write their own tasks to move these folders from this temporary path to the directory of their own choice. Following variables will be used by the Exility during file upload operation. Hence application teams are requested not to use those parameters in their coding. Parameters are as follows,

1. ExilityFilePath
2. ExilityFileItemList
3. ExilityIsMultiPartRequest

Name	Type	Mandatory	Explanation
Action attributes			Refer to the basic action attributes.
closeWindow	Boolean	No	If we give True then upload window will be closed. If we give false then window will not be closed.

7. Look & Feel -- CSS

Most of the look-and-feel is controlled though CSS in Exility. Following is a list of class names used by Exility.

button	form button <input type=button>
Imagebutton	form button <input type=image>
label	<span> label of a field
field	outputField <span>
Headerpanel	not used any more. Used to be a header panel<div>
Pageheader	header of the page inside a fieldset <legend>
expandedfieldset	fieldset used to produce outline, when it is expanded <fieldset>
collapsedfieldset	fieldset used to produce outline, when it is collapsed. lower border with looks good <fieldset>
twister	text of the twister for a collapsible panel <legend> if outline is used, <span> otherwise
Tablelabel	label of a panel if no twistie <span>
Buttonpanel	earlier called tools, for button panel <div>
displaypanel	earlier called inputpanel <div>
Fieldstable	table inside display panel <table>
gridpanel	outer div of gridpanel <div>
gridtable	table of grid <table>
listpanel	outer div of listpanel <div>
listtable	table of list <table>
Spacerpanel	div inside a table that takes 100% <div>
subwindowpanel	outerpanel <div>
tabnav	outermost panel of tabbed panel (group)
activetablabel	label of the tab that is active <span>
passivetablabel	label of a tab that is not visible <span>
row1	first row of list/grid <tr>
row2	second row of list/grid <tr>
nodatamessage	A message that is displayed instead of an empty table when a list panel has no data.