My favorites ▼ | Sign in

openbiz-cubi

Rapid PHP Application Development Platform

Search projects

Project Home Downloads Wiki Issues Source Export to GitHub

Search Current pages ▼ for Search

OpenbizFrameworkUI

describe Openbiz view and form

Phase-Implementation

Updated Aug 10, 2011 by rockys...@gmail.com

Openbiz View and Form

Openbiz Data Object (DO) plays a data unit, and Openbiz Form plays as corresponding presentation unit. Each Form declares a DO name and mapping between DO Fields to Form Elements.

Openbiz View plays as a presentation container of Forms. In web technology, View is same as a web page and Form is a logic block within a page.

Openbiz View and Form Basics

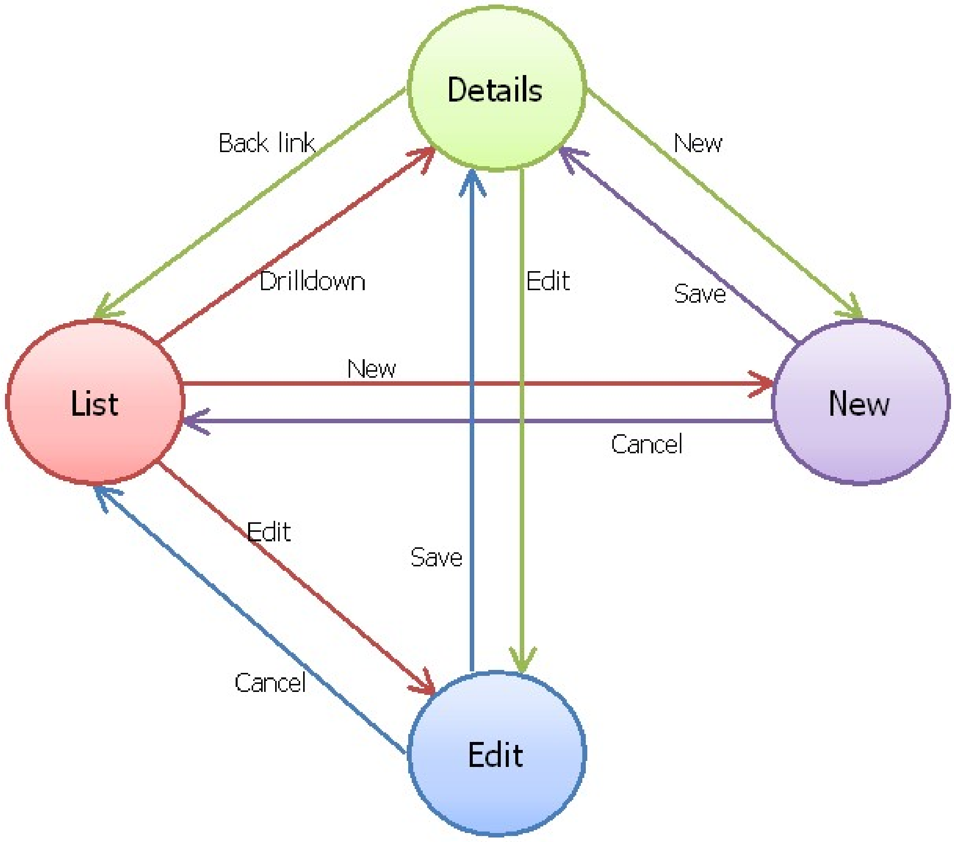
Before giving more detailed description of Openbiz Form and View, let's introduce some basic concepts first.

View and Form Navigation

In a data-driven web applications, data is usually presented in four types of screens.

1. screen for multi-record list or table. We call is as "List" screen
2. screen for single-record detail. We call is as "Detail" screen
3. screen for single-record editing. We call is as "Edit" screen
4. screen for single-record creation. We call is as "New" screen

Navigation among the 4 screens are described in the following diagram.



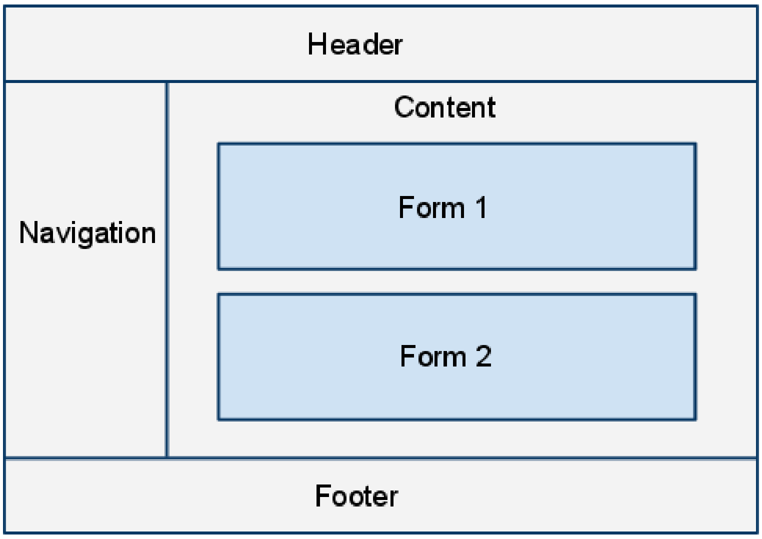
Openbiz recommends and supports two approaches to implement the above navigation flow.

- Navigation between Openbiz Views. Each View provides either list, detail, edit or new user interface. Changing between screens is actually navigation from one web page to another.
- Navigation between Openbiz Form within a View. One View contains 4 Forms - List Form, Detail Form, Edit Form and New Form. Changing between screens is switching from one form to another.

Basic View Layout

Openbiz View is the container of Forms. A Form can be treated as a Tile of a page. Each View has a template file to define it layout. A typical View layout is like

- View header area
- Navigation area
- Main content area. This area can include multiple Forms.
- View footer area

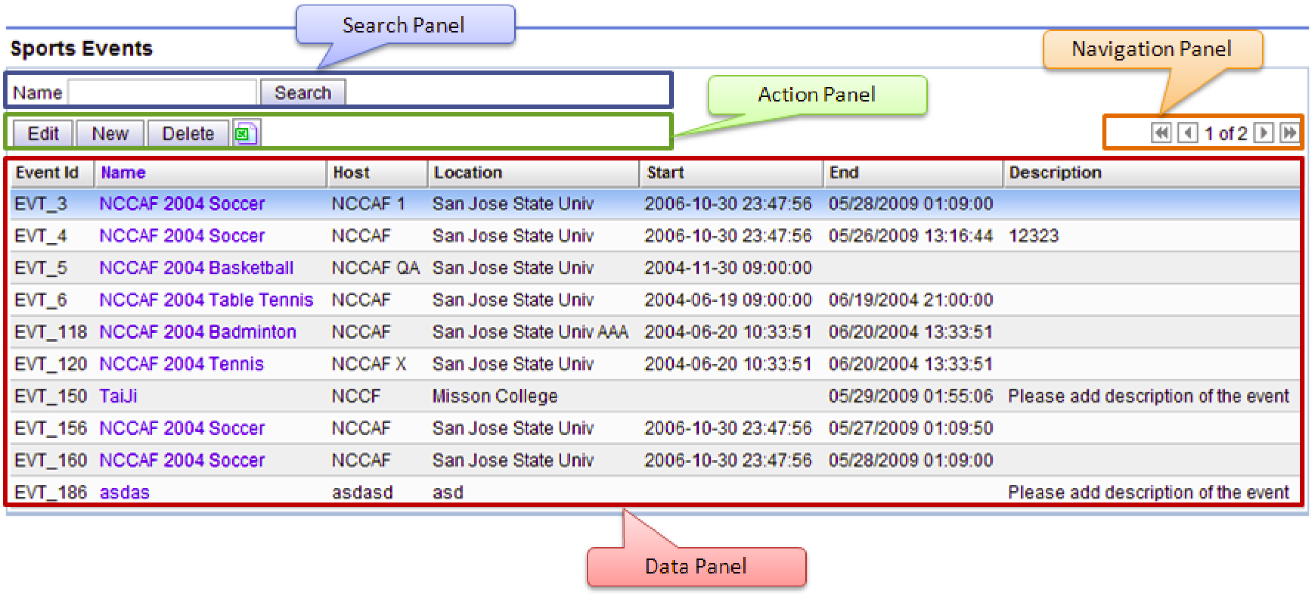


Basic Form Layout

Openbiz Form is composed with four parts - data panel, action panel, navigation panel and search panel

- Data panel is the area showing the data. It can be shown as list, table, form, tree and so on.
- Action panel is a list of controls where users can invoke command on the form.
- Navigation panel is a set of controls that controls data navigation such as paging, scrolling.
- Search panel usually includes the search entries and search button.

The screenshot below shows a sample of 4 panel layout of a Form.



Event Id	Name	Host	Location	Start	End	Description
EVT_3	NCCAF 2004 Soccer	NCCAF 1	San Jose State Univ	2006-10-30 23:47:56	05/28/2009 01:09:00	
EVT_4	NCCAF 2004 Soccer	NCCAF	San Jose State Univ	2006-10-30 23:47:56	05/26/2009 13:16:44	12323
EVT_5	NCCAF 2004 Basketball	NCCAF QA	San Jose State Univ	2004-11-30 09:00:00		
EVT_6	NCCAF 2004 Table Tennis	NCCAF	San Jose State Univ	2004-06-19 09:00:00	06/19/2004 21:00:00	
EVT_118	NCCAF 2004 Badminton	NCCAF	San Jose State Univ AAA	2004-06-20 10:33:51	06/20/2004 13:33:51	
EVT_120	NCCAF 2004 Tennis	NCCAF X	San Jose State Univ	2004-06-20 10:33:51	06/20/2004 13:33:51	
EVT_150	TaiJi	NCCF	Misson College		05/29/2009 01:55:06	Please add description of the event
EVT_156	NCCAF 2004 Soccer	NCCAF	San Jose State Univ	2006-10-30 23:47:56	05/27/2009 01:09:50	
EVT_160	NCCAF 2004 Soccer	NCCAF	San Jose State Univ	2006-10-30 23:47:56	05/28/2009 01:09:00	
EVT_186	asdas	asdasd	asd			Please add description of the event

Bind Form with Data Object

A main usage of Form is to present data defined in DO. Form defines the mapping between elements in the Form and the fields in DO. Here's an example of a List Form.

```
<EasyForm Name="EventList" ... BizDataObj="EventDO" ...>
 <DataPanel>
   <Element Name="id" FieldName="Id" Class="ColumnText" Label="Event Id"/>
   <Element Name="event_name" FieldName="event_name" Class="ColumnText" Label="Event Name"/>
   <Element Name="event_time" FieldName="event_time" Class="ColumnText" Label="Event Time"/>
 </DataPanel>
</EasyForm >
```

View Metadata

Openbiz View has a core class “EasyView”. This class metadata is described below.

```
<EasyView ...>
    <FormReferences>
        <Reference ...> *
```

EasyView Element

EasyView element is the root element of a Form metadata. It has a child element - FormReferences. EasyView element has the following attributes:

Name

Name is the identifier of a View object. Name should be same as the View file name.

Description

Description describes the functions and behavior of the View.

Class

Class specifies the name of its implementation class.

TemplateEngine

TemplateEngine tells the type of template engine applied on the view. TemplateEngine can be either Smarty (this is default) or PHP.

TemplateFile

TemplateFile gives the path of template file which defines layout of the view.

MessageFile

MessageFile specifies the file path that provides message strings for the form. Message strings are used in the class source code for displaying messages like error, alert and log message.

CacheLifetime

CacheLifetime sets the View cache life time in seconds. If CacheLifetime is larger than 0, View rendering output is cached for the length given by the value. By default the cache is turned off.

Form Metadata

Openbiz Form has a core class “EasyForm”. This class metadata is described below.

```
<EasyForm ...>
    <DataPanel>
        <Element ...> *
            <EventHandler .../> *
    <ActionPanel>
        <Element ...> *
            <EventHandler .../> *
    <NavPanel>
        <Element ...> *
            <EventHandler .../> *

<SearchPanel>
        <Element ...> *
            <EventHandler .../> *
```

EasyForm Element

EasyForm element is the root element of a Form metadata. It has four children elements - DataPanel, ActionPanel, NavPanel and SearchPanel. EasyView element has the following attributes:

Name

Name is the identifier of a Form object. Name should be same as the View file name.

Description

Description describes the functions and behavior of the Form.

Title

Title specifies the title of the Form.

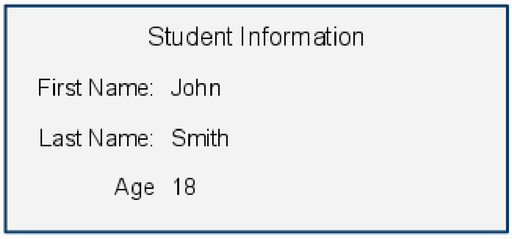
Class

Class specifies the name of its implementation class.

FormType

FormType specifies the type of the Form. Openbiz Form supports four type of form.

Detail Form Detail form is usually a read-only form that shows data of one data record.



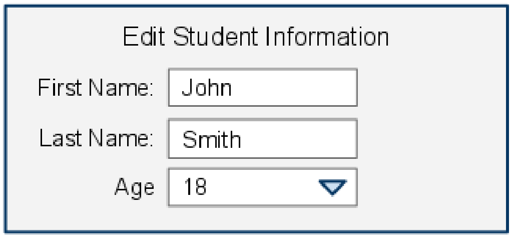
Student Information

First Name: John

Last Name: Smith

Age 18

Edit Form Edit form is a form that used for editing one data record



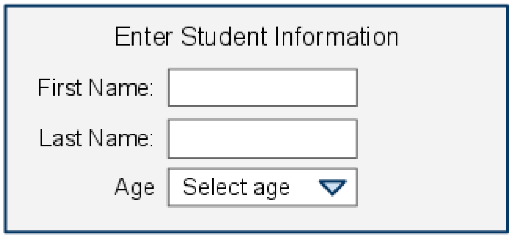
Edit Student Information

First Name: John

Last Name: Smith

Age 18

New Form New form is a form that used for entering a new data record



Enter Student Information

First Name:

Last Name:

Age Select age

List Form List form is a form that used for listing multiple data records. On browser, a list form is usually rendered as a html table.

Student Id	First Name	Last Name	Age
101212	John	Smith	18
101213	Alex	Johnson	19
101214	Mary	Williams	17

jsClass

jsClass specifies the name of its javascript implementation class. By default, Openbiz.Form should be used for forms with type as Detail, Edit or New; Openbiz.TableForm should be used for List forms. A custom javascript class can be developed and set in jsClass as well.

BizDataObj

BizDataObj specifies the name of a DO that plays as data model of the Form. Form’s Elements usually map to the Fields of this DO.

SearchRule

SearchRule specifies the search rule that apply additional filter on the DO’s query.

TemplateEngine

TemplateEngine tells the type of template engine applied on the form. TemplateEngine can be either Smarty (this is default) or PHP.

TemplateFile

TemplateFile gives the path of template file which defines layout of the form.

PageSize

PageSize specifies the number of records displayed in the list form. So this attribute only used in “List” type of form.

MessageFile

MessageFile specifies the file path that provides message strings for the form. Message strings are used in the class source code for displaying messages like error, alert and log message.

CacheLifetime

CacheLifetime sets the Form cache life time in seconds. If CacheLifetime is larger than 0, Form rendering output is cached for the length given by the value. By default the cache is turned off.

Element Metadata

Openbiz Form Element is a basic UI element. By applying different implementation class, one element can become different type of UI unit. Openbiz core provides more than 20 element classes which are usually enough for normal web applications. There are many element attributes are common across all element types, while each element type might have its own special attributes.

On a typical user interface, the smallest unit can be classified into the following categories:

- Text
- Input
- Selection
- Button
- Other elements

Element’s Common Attributes

Some attributes are common across all Openbiz Form Elements.

- Name. The name of the element. Element name should be unique in a Form.
- Description. Content describing the element.
- Class. The name of the implementation class of the element.
- CssClass. The name of css class applied on the element.
- Style. The in-line css style for the element.
- Height. The height of the element.
- Width. The width of the element.
- HTMLAttr. Html attributes applied to the elements.

Text

Developers can use following classes to present text messages.

- LabelText. LabelText displays a data field with its name (as label) and value (as text).
- ColumnText. ColumnText displays a data field with its name (as a table column header) and value (in table cell).

Text elements can have attributes like

- FieldName. The name of the Form’s DO Field. This Field provides data for Element to display.
- Label. The label of the displayed text. For ColumnText, “Label” is the table column header.
- Link. If an url is given to the Link attribute, a hyperlink will be added on the text value.
- Sortable. This attribute apply only ColumnText class and its sub-classes. Sortable can have 2 values - “Y” or “N”.
  - “Y” means the column is sortable. Clicking the column header will resort the list on that column.
  - “N” means the column is not sortable. No action is triggered on clicking the column header.

Input

Developers can use following classes to present input elements.

- InputText. InputText displays a data field with its name (as label) and value in an edit box.
- Textarea. Textarea displays a data field with its name (as label) and value in an textarea.
- Password. Password displays a password type data field with its name (as label) and value in a password input box.
- InputDate. InputDate displays a date field with its name (as label) and value in an edit box. It also comes with a date picker for user to pick a date from a popup calendar.
- InputDatetime. InputDatetime displays a datetime field with its name (as label) and value in an edit box. It also comes with a date and time picker for user to pick a date/time from a popup calendar.
- CKEditor. CKEditor displays a data field with its name (as label) and value in a rich text editor called CKEditor.

Input elements can have attributes like

- FieldName. The name of the Form’s DO Field. This Field provides data for Element to edit.
- Label. The label of the input element.
- Enabled. If Enabled is set to “N”, the element becomes a read only one.
- Hidden. If Hidden is set to “Y”, the element becomes invisible on page.
- DefaultValue. If DefaultValue is set as non-empty string, it will be used as pre-fill value on new record creation (in New Form).
- Required. If Required is set to “Y”, Form will check if there is no-empty input on the element.
- Validator. Validator can be given Simple Expression to invoke advanced or custom validation logic.

Selection

Developers can use following classes to present selection inputs.

- Listbox. Listbox displays a data field with its name (as label) and list of values in a standard list box.
- DropdownList. DropdownList displays a data field with its name (as label) and list of value in a dhtml dropdown list.
- Checkbox. Checbox displays a data field with its name (as label) and a checkbox.
- Radio. Radio displays a data field with its name (as label) and list of radio buttons.
- InputPicker. InputPicker displays a data field with its name (as label), value in a input box and a picker button. Clicking on the button will launch a popup for picking values into the input box (and other inputs).
- AutoSuggest. AutoSuggest displays a data field with its name (as label) and value in an input box. Typing in the input box will trigger a dropdown list that lists related values to the user input.
- EditCombobox. EditCombox displays a data field with its name (as label) and list of values in a list box. User can type in value as well as pick a value from the given list.

Selection elements can have attributes like

- FieldName. The name of the Form’s DO Field. This Field provides data for Element to display.
- Label. The label of the selection element.
- SelectFrom. SelectFrom provides a list of option to be picked as the element value. Details is covered in a separate chapter.
- ValuePicker. This attribute is only used by InputPicker (and its subclass) element. ValuePicker specifies a popup Form where provides a list of records for user to pick.
- PickerMap. This attribute is only used by InputPicker (and its subclass) element. PickerMap specifies the mapping between picked record and to-be-changed fields.
- Enabled. If Enabled is set to “N”, the element becomes a read only one.
- Hidden. If Hidden is set to “Y”, the element becomes invisible on page.
- DefaultValue. If DefaultValue is set as non-empty string, it will be used as pre-fill value on new record creation (in New Form).
- Required. If Required is set to “Y”, Form will check if there is no-empty input on the element.
- Validator. Validator can be given Simple Expression to invoke advanced or custom validation logic.

Button

Developers can use following classes to present buttons.

- Button. Button is a general button that support both caption and image on the button.
- HTMLButton. HTMLButton is the standard html button with Caption.
- ResetButton. ResetButton is the standard form reset button
- SubmitButton. SubmitButton is the standard form submit button

Button elements can have attributes like

- Text. Text is the actual caption displayed on a button.
- Image. Image is used to display on a button. This attribute is only available to “Button”.
- Enabled. If Enabled is set to “N”, the element becomes a read only one.
- Hidden. If Hidden is set to “Y”, the element becomes invisible on page.

Other Elements

There are many other element types which give more choices of UI rendering.

Hidden

Hidden type of element is same as “Hidden” input.

File

File type of element is used to uploading a file.

HTMLBlock

HTMLBlock is actually a block of HTML code.

Element EventHandler Metadata

Openbiz Form Element can declare more than1 Event Handlers. In each Event Handler, action can be defined for certain event type. An EventHandler element have following attributes.

- Name. The name of an event handler. Different event handlers at same Form should have different names.
- Event. The name of event. Openbiz supports all events list at http://www.w3.org/TR/html4/interact/scripts.html#h-18.2.3. The most commonly used event is “onclick”.
- Function. Function defines the action function to be invoked on server side when the event happens.Typically, an event handler function points to a public method of the Form object.
- FunctionType. This attribute defines how the function is invoked. “RPC” is the default value. RPC indicates that browser client invokes server side function through AJAX.
- RedirectPage. This attribute defines the which view, form or url will be presented to user after the function is executed successfully.
- ShortcutKey. It maps a combination of shortcut key to the function.
- ContextMenu. It maps a context menu item to the function.

Form API and Samples

Openbiz Form provides a list of public methods that can complete common requests sent from browser client. The widely used Form methods are listed below. Please refer to the API doc for details.

Query and show results

- runSearch. This method takes the search inputs from the client, Issue the query to underline DO and re-render the form with query results.

Insert a new record

- newRecord. This method shows the new record screen with default values filled.
- insertRecord. This method insert the user input data as a new record.

Update a record

- editRecord. This method shows the edit record page on current focused record

- updateRecord. This method saves current edited record with user input

Delete a record

- deleteRecord. This method deletes the current focused record

Sort records

- sortRecord. This method sorts record list on given column and re-render the form

Page Navigation

- gotoPage. This method takes user to the given screen and re-render it.
- switchForm. This methods switches from current Form to another.

Event Handling

Openbiz allows binding event handling functions to element. Event handling is defined in following syntax.

```
<Element Name="" Class="">
  <EventHandler Name="..." Event="..." Function="..." FunctionType="..."
RedirectPage="..." ShortcutKey="..." ContextMenu="..."/> *
</Element>
```

More than event handlers can be associated to an element. The following xml is a sample event handler metadata.

```
<Element Name="btn_save" Class="Button">
  <EventHandler Name="save_onclick" Event="onclick" Function="updateRecord" .../>
</Element>
```

Define an event

“Event” is the name of event. E.g. onclick, onchange … Openbiz supports all events list at http://www.w3.org/TR/html4/interact/scripts.html#h-18.2.3

Define an action

“Function” is used to define the action function to be invoked on server side when the event happens.Typically, an event handler function points to a public method of the Form object. The syntax of a function is

```
<EventHandler ... Function="objectName.methodName(arg1, ...)" .../>
```

The objectName can be empty, this means it calls method of current Form. For example,

```
<EventHandler ... Function="SaveRecord()" .../>.
```

Define how action is invoked

“FunctionType” define how client communicate with server. FunctionType can be either RPC, Page, Form or Popup.

FunctionType	Description	Example
RPC	RPC (Remote Procedure Call) indicates that browser client invokes server side function through AJAX. It is default value if FunctionType is empty	New record button: `<EventHandler Name="btn_new" Event="onclick" Function="newRecord()" />`
Page	the function with type Page is invoked on server side with page reload, Form user input data is not passed to server	Page the function with type Page is invoked on server side with page reload, Form user input data is not passed to server Export button to bring up a file save dialog in the page `<EventHandler Name="btn_exp" Event="onclick" Function="CallService(ioService,exportXML)" FunctionType="Page"/>`
Form	the function with type Form is invoked on server side by form submission	Upload button for file upload in the form: `<EventHandler Name="btn_upload" Event="onclick" Function="uploadFile()" FunctionType="Form"/>`
Popup	the function with type Popup is invoked on server side targeting to a new popup window	Excel output button to show excel format in a popup: `<EventHandler Name="onclick" Event="onclick" Function="CallService(excelService,renderHTML)" FunctionType="Popup"/>`

Define post action

Developers can use “RedirectPage” to define the redirected view or form after an action is executed successfully.

- RedirectPage can be a normal url that points to a View or a non-Openbiz page.
- If RedirectPage starts with "form=...", it will switch to the given form (not a full page/view)

Associate action with key and mouse

ShortbutKey is used to map a key combination to the action. If ContextMenu is defined, the event handler will appear as a menu item on mouse right click. For example:

```
<Element Name="btn_cancel" Class="Button" Text="Cancel">
   <EventHandler Name="cancel_onclick" Event="onclick" Function="SwitchForm(backup.form.BackupListForm)" ShortcutKey="Esc
</Element>
```

Bind Data to Listbox

Openbiz allows binding static or dynamic data to Listbox element as well as other elements that derived from OptionElement.

Bind simple list to Listbox

In a Form, developers can assign `SelectFrom="Selection(Option1|Option2|Option3|...)"` to an Element. Then the listbox will have Option1, Option2 and Option3 in the list.

Bind static list from xml file to Listbox

In a Form, developers can assign `SelectFrom="LOV(ABC)"` to an Element. This means this field element is a Listbox whose data is from the elements ABC in LOV xml file.

For example, `SelectFrom="Enum(Chart)"` will list all “Chart” element in Enum.xml file. Enum.xml

```
<Chart Value="Column Chart"/>
<Chart Value="Line Chart"/>
<Chart Value="Bar Chart"/>
```

Also, you can give both value and text. If only give value, Openbiz uses the value as the display text.

```
<Chart Value="Column_Chart" Text=”Column Chart”/>
<Chart Value="Line_Chart" Text=”Line Chart”/>
<Chart Value="Bar_Chart" Text=”Bar Chart”/>
```

Developers can make up their own xml file that has list of values for selection.

Bind dynamic list (Table column) to Listbox

In a Form, developers can assign `SelectFrom="DO[Field]"` to an Element. This means the list of data is from the table column mapping to the Field of the DO. To avoid same values appear in the Listbox, please make sure the DO query returns an unique list.

The following format `SelectFrom="DO[Field4Text:Field4Value]"` is also supported so that the listbox displays Field4Text field as texts and Field4Value as values.

Further more, a SearchRule can be applied on the SelectFrom with syntax as `SelectFrom="DO[Field4Text:Field4Value], DO_SearchRule"` For example,

```
<Element Name="fld_conference" Class=”Listbox” SelectFrom="EventDO[Name], [Name] LIKE '%Conference%'"/>
```

Input Validation

Input validation can be set in Form metadata. This is mainly to check the user data input in record insert and update. Openbiz Form use “Required” and “Validator” attribute at Element level to control the input. If validation fails, Form throws a validation exception to the caller.

Required Check

Openbiz checks if an element can accept empty input value if “Required” attribute is set in Element metadata. Syntax:

```
<Element Name="..." Required="Y|N" .../>
```

Example:

```
<Element Name="fld_name" FieldName="name" Required="Y" />
```

Validator

Element can use “Validator” to set flexible validation rules for this element.

Validate field with simple expression

Simple Expression can be used in validator. Example:

```
<Element Name="fld_age" FieldName=”age” Validator="{[age]<15}">
```

Validate field with validation service

Validation service (Openbiz uses Zend Validation) can be use to validate the input data in Validator attribute. Examples:

```
<Element Name="Email" Validator="{@validate:email('[Email]')}"/>
<Element Name="Phone" Validator="{@validate:phone('[Phone]')}"/>
```

Form Element Dependency

In a complex form, it happens that some elements values depend on the values of other elements. A good example is one form has country and state list boxes. When user changes the pick of country, the state list changes accordingly. This is so called form element dependency.

With the help of Simple Expression, elements' relationship can be specified in Form metadata file. The following example shows how to link multiple elements within a form.

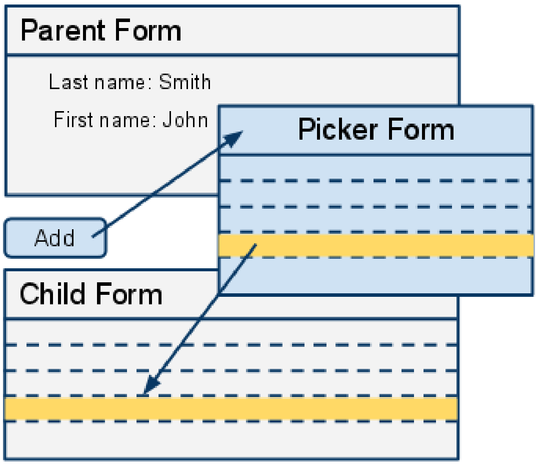
```
<EasyForm>
 <DataPanel>
      <Element Name="fld_combo1" Class="Listbox" FieldName="" Label="Event test combo box" SelectFrom="easy.Selection(Tes
             <EventHandler Name="onchange" Event="onchange" Function="UpdateForm()"/>
      </Element>
      <Element Name="fld_edit1" Class="InputText" FieldName="" Label="Enable/disable switch" Enabled="{(@:Elem[fld_combo1
      </Element>
      <Element Name="fld_edit2" Class="InputText" FieldName="" Label="Hide/show switch" Hidden="{(@:Elem[fld_combo1].Valu
      </Element>
      <Element Name="fld_combo2" Class="Listbox" FieldName="" label="cascade combobox" SelectFrom="easy.Selection({@:Elem
      </Element>
 </DataPanel>
</EasyForm>
```

The first combobox is the driving control. Based on its value, - 2nd control will be enabled or disabled (by setting simple expression on its “Enabled” attribute). - 3rd control will be hidden or shown (by setting simple expression on its “Hidden” attribute). - 4th control will have different lists (by setting simple expression on its “SelectFrom” attribute).

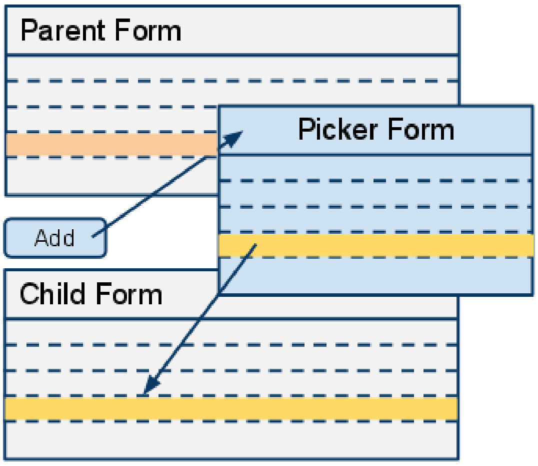
Data Record Picker

Openbiz allows users to pick record from a different Form on the following cases:

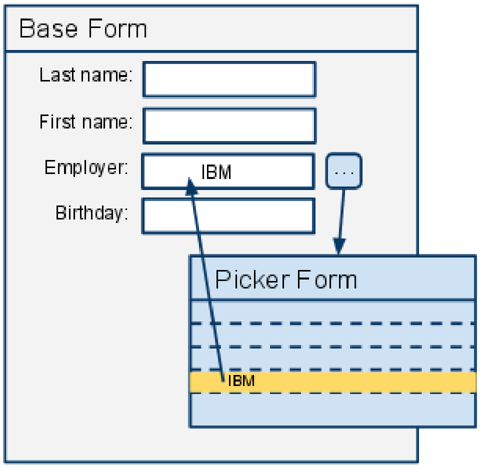
- Add a child record in the child form (parent-child relationship is 1-M) by picking an existing child record



- Add a child record in the child form (parent-child relationship is M-M) by picking an existing child record



- Add or change the joined fields by picking a record from joined dataobj.



Note: Openbiz keeps only one instance of a metadata object (of DataObj, Form, View) in a user session, so always configure a different object when two instances of same entity needed in one view. For example, on an Event form (EventForm / !EventDO), we usually configure an event popup as (EventPopForm / !EventPopDO). There are two ways to configure a record-selecting popup Form.

Configure Element to show popup

To configure element to show popup form, developers can set "LoadDialog(FormName)" in the EventHandler Function attribute. This configuration can be used in picking a child record (M-1 or M-M) in the child form.

Sample:

```
<Element Name="btn_add" Class="Button" text="Add">
 <EventHandler Name="add_clk" Event="onclick" Function="LoadDialog(UserPicker)"/>
```

On the popup form, which is usually based on PickerForm class, set a button to call AddToParent() to add the current selected record to the base form (the form launches the popup).

Configure InputPicker to show popup

In order to pick values from a popup window and populate them to base form elements, you can use InputPicker element that has the pick button at the right side of an input box. To configure the InputPicker element, ValuePicker and PickerMap need to be specified correctly.

- ValuePicker attribute. ValuePicker gives the view and form to be shown in the popup.
  - It has syntax as `ValuePicker="picker form name"`.
- PickerMap attribute. PickerMap is used to map the picked record from a popup form to the base record that initiates the popup. Openbiz will pick values from popup form according to the mapping.
  - The syntax of PickerMap is `PickerMap="thisform_element_1:popup_element_1, thisform_element_2:popup_element_2, ..."`

Sample:

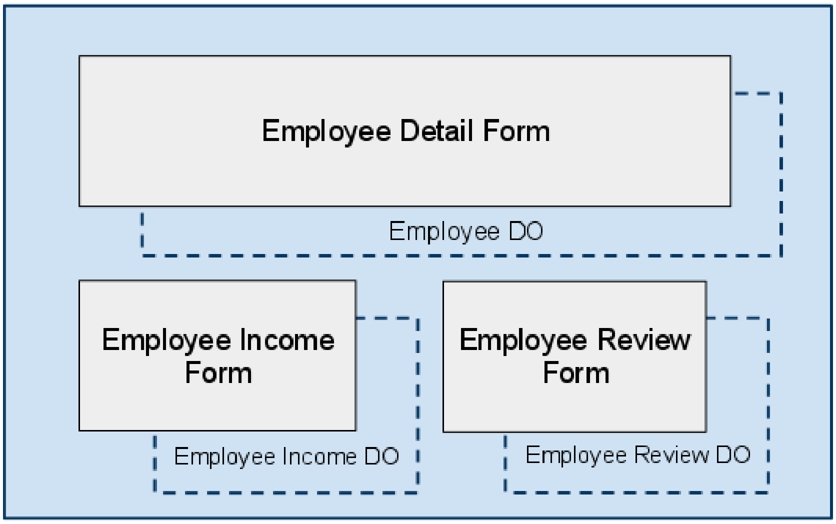
```
<Element Name="fld_to" Class="InputPicker" FieldName="to" Label="To ..." ValuePicker=”UserPickForm” PickerMap="fld_to_user
```

On the popup form, add a button to call JoinToParent() to join the current selected record to the base form (the form launches the popup).

Form Parent-Child Relationship

A complex View can have more than one Forms. In many scenario, one Form is the main one and other Forms data depends on the data of main Form. It is usually called parent-child relationship.

For example, in an Employee detail View, the main Form is employee detail Form, while there are employee profile Form and employee income Form as children Forms.



Each Form object has a corresponding DO behind it. The DO of the parent Form is "main DO", the DO of a child form is "reference DO".

Define Form Relationship

Parent child relationship between forms is defined in View metadata. For the above Employee detail View, the metadata is like

```
<EasyView Name="EmployeeDetailView" ...>
 <FormReferences>
   <Reference Name="EmployeeDetailForm" SubForm="EmployeeIncomeForm,EmployeeReviewForm"/>
   <Reference Name="EmployeeIncomeForm"/>
   <Reference Name="EmployeeReviewForm"/>
 </FormReferences>
</EasyView>
```

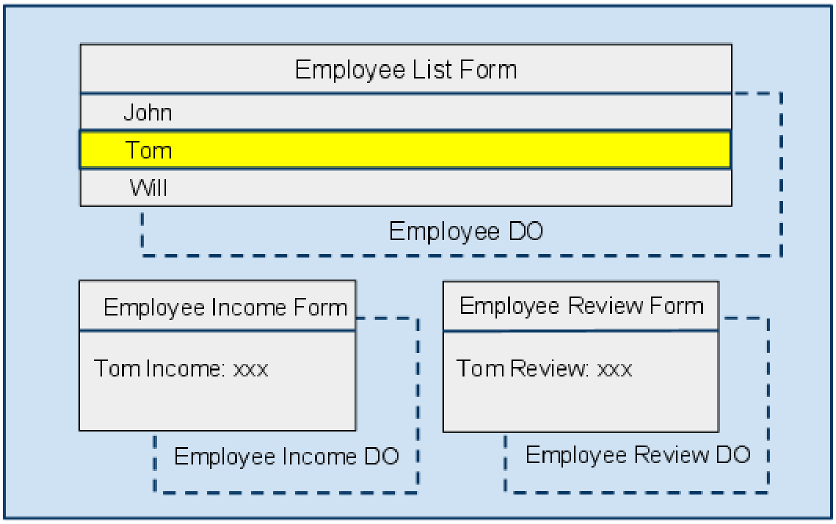
Link Forms

So far even if EmployeeDetailForm has 2 children Forms, they are still separated - the child Income Form can list income of all employees. While we want to display income of the employee on the parent employee detail Form. In order to link Forms at data level, the child Form's DO must be a reference DO of the parent Form's DO. It means in the Employee DO metadata needs to have !EmployeeIncomeDO and !EmployeeReviewDO in its ObjReference. Its metadata is like

```
<BizDataObj Name="EmployeeDO" Class="BizDataObj" Table="employee" ...>
...
 <ObjReferences>
   <Object Name="EmployeeIncomeDO" Relationship="1-M" Table="emp_income" Column="id" FieldRef="emp_id"/>
   <Object Name="EmployeeReviewDO" Relationship="1-M" Table="emp_review" Column="id" FieldRef="emp_id"/>
 </ObjReferences>
</BizDataObj>
```

Cascade Change on Child Form

In certain case, the parent Form is a list Form. When a user picks a record on the parent Form, we want to the data at child Forms changes accordingly.



As discussed in previous paragraphs, with correct settings of SubForm in View and ObjReference in Form's DO, this type of advanced UI features can be done easily in Openbiz.

Comment by FirmanRi...@gmail.com, Jun 8, 2012 where is the place for xml file if i want to bind static list from xml file to listBox????

► Sign in to add a comment