Menus and Actions

Menus

Menus are records in the `ir.ui.menu` table. In order to create a new menu entry, you can directly create a record using the `record` tag.

```
<record id="menu_xml_id" model="ir.ui.menu">
  <field name="name">My Menu</field>
  <field name="action" ref="action_xml_id"/>
  <field name="sequence" eval="<integer>"/>
  <field name="parent_id" ref="parent_menu_xml_id"/>
</record>
```

There is a shortcut by using the `menuitem` tag that **you should use preferentially**. It offers a flexible way to easily define the menu entry along with icons and other fields.

```
<menuitem id="menu_xml_id"
    name="My Menu"
    action="action_xml_id"
    icon="NAME_FROM_LIST"
    groups="groupname"
    sequence="<integer>"
    parent="parent_menu_xml_id"
/>
```

Where

- `id` specifies the **xml identifier** of the menu item in the menu items table. This identifier must be unique. Mandatory field.
- `name` defines the menu name that will be displayed in the client. Mandatory field.
- `action` specifies the identifier of the attached action defined in the action table (`ir.actions.act_window`). This field is not mandatory : you can define menu elements without associating actions to them. This is useful when defining custom icons for menu elements that will act as folders. This is how custom icons for “Projects” or “Human Resources” in OpenERP are defined).
- `groups` specifies which group of user can see the menu item. (example : groups=”admin”). See section ” Management of Access Rights” for more information. Multiple groups should be separated by a ‘,’ (example: groups=”admin,user”)
- `sequence` is an integer that is used to sort the menu item in the menu. The higher the sequence number, the downer the menu item. This argument is not mandatory: if sequence is not specified, the menu item gets a default sequence number of 10. Menu items with the same sequence numbers are sorted by order of creation (`_order = "*sequence,id*"`).

The main current limitation of using `menuitem` is that the menu action must be an `act_window` action. This kind of actions is the most used action in OpenERP. However for some menus you will use other actions. For example, the Feeds page that comes with the mail module is a client action. For this kind of menu entry, you can combine both declaration, as defined in the mail module :

```
<!-- toplevel menu -->
<menuitem id="mail_feeds_main" name="Feeds" sequence="0"
    web_icon="static/src/img/feeds.png"
    web_icon_hover="static/src/img/feeds-hover.png" />
<record id="mail_feeds_main" model="ir.ui.menu">
    <field name="action" ref="action_mail_all_feeds"/>
</record>
```

Actions

The actions define the behavior of the system in response to the actions of the users ; login of a new user, double-click on an invoice, click on the action button, ...

There are different types of simple actions:

- **Window**: Opening of a new window
- **Report**: The printing of a report
  - o Custom Report: The personalized reports o RML Report: The XSL:RML reports
- **Execute**: The execution of a method on the server side
- **Group**: Gather some actions in one group

The actions are used for the following events:

- User connection,
- The user clicks on a menu,
- The user clicks on the icon ‘print’ or ‘action’.

Opening of the menu

When the user open the option of the menu “Operations > Partners > Partners Contact”, the next steps are done to give the user information on the action to undertake.

1. Search the action in the IR.
2. Execution of the action
    1. If the action is the type Opening the Window; it indicates to the user that a new window must be opened for a selected object and it gives you the view (form or list) and the filed to use (only the pro-forma invoice).
    2. The user asks the object and receives information necessary to trace a form; the fields description and the XML view.

User connection

When a new user is connected to the server, the client must search the action to use for the first screen of this user. Generally, this action is: open the menu in the 'Operations' section.

The steps are:

1. Reading of a user file to obtain ACTION_ID
2. Reading of the action and execution of this one

The fields

Action Name
The action name

Action Type
Always 'ir.actions.act_window'

View Ref
The view used for showing the object

Model
The model of the object to post

Type of View
The type of view (Tree/Form)

Domain Value
The domain that decreases the visible data with this view

The view

The view describes how the edition form or the data tree/list appear on screen. The views can be of 'Form' or 'Tree' type, according to whether they represent a form for the edition or a list/tree for global data viewing.

A form can be called by an action opening in 'Tree' mode. The form view is generally opened from the list mode (like if the user pushes on 'switch view').

The domain

This parameter allows you to regulate which resources are visible in a selected view.(restriction)

For example, in the invoice case, you can define an action that opens a view that shows only invoices not paid.

The domains are written in python; list of tuples. The tuples have three elements;

- the field on which the test must be done
- the operator used for the test (<, >, =, like)
- the tested value

For example, if you want to obtain only 'Draft' invoice, use the following domain; [('state','=','draft')]

In the case of a simple view, the domain define the resources which are the roots of the tree. The other resources, even if they are not from a part of the domain will be posted if the user develop the branches of the tree.

Window Action

Actions are explained in more detail in section "Administration Modules - Actions". Here's the template of an action XML record :

```
<record model="ir.actions.act_window" id="action_id_1">
    <field name="name">action.name</field>
    <field name="view_id" ref="view_id_1"/>
    <field name="domain">["list of 3-tuples (max 250 characters)"]</field>
    <field name="context">{"context dictionary (max 250 characters)"}</field>
    <field name="res_model">Open.object</field>
    <field name="view_type">form|tree</field>
    <field name="view_mode">form,tree|tree,form|form|tree</field>
    <field name="usage">menu</field>
    <field name="target">new</field>
</record>
```

Where

- **id** is the identifier of the action in the table "ir.actions.act_window". It must be unique.
- **name** is the name of the action (mandatory).

- **view_id** is the name of the view to display when the action is activated. If this field is not defined, the view of a kind (list or form) associated to the object res_model with the highest priority field is used (if two views have the same priority, the first defined view of a kind is used).
- **domain** is a list of constraints used to refine the results of a selection, and hence to get less records displayed in the view. Constraints of the list are linked together with an AND clause : a record of the table will be displayed in the view only if all the constraints are satisfied.
- **context** is the context dictionary which will be visible in the view that will be opened when the action is activated. Context dictionaries are declared with the same syntax as Python dictionaries in the XML file. For more information about context dictionaries, see section " The context Dictionary".
- **res_model** is the name of the object on which the action operates.
- **view_type** is set to form when the action must open a new form view, and is set to tree when the action must open a new tree view.
- **view_mode** is only considered if view_type is form, and ignored otherwise. The four possibilities are :
    - **form,tree** : the view is first displayed as a form, the list view can be displayed by clicking the "alternate view button" ;
    - **tree,form** : the view is first displayed as a list, the form view can be displayed by clicking the "alternate view button" ;
    - **form** : the view is displayed as a form and there is no way to switch to list view ;
    - **tree** : the view is displayed as a list and there is no way to switch to form view.

(version 5 introduced **graph** and **calendar** views)

- **usage** is used [+ ***TODO*** +]
- **target** the view will open in new window like wizard.
- **context** will be passed to the action itself and added to its global context

```
        <record model="ir.actions.act_window" id="a">
            <field name="name">account.account.tree1</field>
            <field name="res_model">account.account</field>
            <field name="view_type">tree</field>
            <field name="view_mode">form,tree</field>
            <field name="view_id" ref="v"/>
            <field name="domain">[('code','=','0')]</field>
            <field name="context">{'project_id': active_id}</field>
        </record>
```

They indicate at the user that he has to open a new window in a new 'tab'.

Administration > Custom > Low Level > Base > Action > Window Actions

Examples of actions

This action is declared in server/bin/addons/project/project_view.xml.

```
<record model="ir.actions.act_window" id="open_view_my_project">
    <field name="name">project.project</field>
    <field name="res_model">project.project</field>
    <field name="view_type">tree</field>
    <field name="domain">[('parent_id','=',False), ('manager', '=', uid)]</field>
    <field name="view_id" ref="view_my_project" />
</record>
```

This action is declared in server/bin/addons/stock/stock_view.xml.

```
<record model="ir.actions.act_window" id="action_picking_form">
    <field name="name">stock.picking</field>
    <field name="res_model">stock.picking</field>
    <field name="type">ir.actions.act_window</field>
    <field name="view_type">form</field>
    <field name="view_id" ref="view_picking_form"/>
    <field name="context">{'contact_display': 'partner'}</field>
</record>
```

Url Action

Report Action

Report declaration

Reports in OpenERP are explained in chapter "Reports Reporting". Here's an example of a XML file that declares a RML report :

```
<?xml version="1.0"?>
<openerp>
    <data>
    <report id="sale_category_print"
            string="Sales Orders By Categories"
            model="sale.order"
            name="sale_category.print"
            rml="sale_category/report/sale_category_report.rml"
            menu="True"
            auto="False"/>
```

```
    </data>
</openerp>
```

A report is declared using a **report tag** inside a “data” block. The different arguments of a report tag are :

- **id** : an identifier which must be unique.
- **string** : the text of the menu that calls the report (if any, see below).
- **model** : the OpenERP object on which the report will be rendered.
- **rml** : the .RML report model. Important Note : Path is relative to addons/ directory.
- **menu** : whether the report will be able to be called directly via the client or not. Setting menu to False is useful in case of reports called by wizards.
- **auto** : determines if the .RML file must be parsed using the default parser or not. Using a custom parser allows you to define additional functions to your report.

Action creation

Linking events to action

The available type of events are:

- **client_print_multi** (print from a list or form)
- **client_action_multi** (action from a list or form)
- **tree_but_open** (double click on the item of a tree, like the menu)
- **tree_but_action** (action on the items of a tree)

To map an events to an action:

```
<record model="ir.values" id="ir_open_journal_period">
    <field name="key2">tree_but_open</field>
    <field name="model">account.journal.period</field>
    <field name="name">Open Journal</field>
    <field name="value" eval="'ir.actions.wizard,%d'%action_move_journal_line_form_select"/>
    <field name="object" eval="True"/>
</record>
```

If you double click on a journal/period (object: account.journal.period), this will open the selected wizard. (id=”action_move_journal_line_form_select”).

You can use a res_id field to allow this action only if the user click on a specific object.

```
<record model="ir.values" id="ir_open_journal_period">
    <field name="key2">tree_but_open</field>
    <field name="model">account.journal.period</field>
    <field name="name">Open Journal</field>
    <field name="value" eval="'ir.actions.wizard,%d'%action_move_journal_line_form_select"/>
    <field name="res_id" eval="3"/>
    <field name="object" eval="True"/>
</record>
```

The action will be triggered if the user clicks on the account.journal.period n°3.

When you declare wizard, report or menus, the ir.values creation is automatically made with these tags:

- <menuitem... />
- <report... />

So you usually do not need to add the mapping by yourself.