Create Joomla! extensions - Manage the Back End- Part 1

Creating extensions for Joomla! 1.5 - Managing the Back - Part 1 view records

Writing extensions with **Marco's Component Maker for Joomla!** is quite easy since it is the program that takes care to write the skeleton for models, views and templates for editing. You have only to define the business logic for data managing, *but this is your own task.*

You need to install the example component created in previous article.

Back End - Part 1 view records

Each component of Joomla! has, whether we speak of front end or back end, an entry point. This is a PHP file, named as the component, which receives the requests that must be managed by the component. This file will then define which **view** and which **controller** to load.

Component Entry point

Since version 1.5.0 you don't need to edit entry point to make component work because a sub menu entry is created aumatically for any controller (/model/table) for record listing. Anyway you may need to change the menu order or remove items for auxiliary controllers

The file generated by marco's component maker includes code for menu generating starting from line 22.

```
... [administrator/components/com_mcm/mcm.php]
foreach($controllers as $controller){
 $link = JRoute::_("index.php?option=com_mcm&controller={$controller}");
 $selected = ($controller == JRequest::getWord('controller'));
 JSubMenuHelper::addEntry(JText::_($controller), "index.php?option=com_mcm&
controller={$controller}", ($controller == JRequest::getWord('controller')));
}
```

$controllers is defined in line 18, modify to change item's order in sub menu:

```
$controllers = explode(',', 'contentlist,categorieslist,sectionslist');
```

File **mcm.php**, as shown by the code, loads the specified controller, which contains the code needed to perform all the tasks ($ task) needed for data managing.

With the its inclusion, the controller calls the **constructor** and the **method display** () of base the class **JController**; and the Joomla! framework then loads the following files:

- /administrator/components/com_mcm/views/XXXlist/ view.html.php
- /administrator/components/com_mcm/views/XXXlist/tmpl/default.php
- /administrator/components/com_mcm/ models/XXXlist.php
- /administrator/components/com_mcm/tables/XXXlist.php

A step back:

1. when statement $controller->execute(JRequest::getVar('task')) is executed, in the file /components/com_mcm/mcm.php,
2. is invoked the method **display()** in file **/administrator/components/com_mcmt/controllers /XXXlist.php**
3. The controller call the method display() of the class McmViewXXXlist in /components/com_mcm /views/XXXlist/view.html.php

Within the last file we have, among others, the following lines of code:

```
function display ($tpl = null) (
 ...
JToolBarHelper ::[...]
 ...
$data = $this->get( 'Data');
$this->assignRef('rows', $data);
...
parent::display ($tpl);
)
```

The header of the method display() require the optional parameter $tpl, Attention! *is not the name of the template to load, but the name of subtemplate!*

1. Instructions **JToolBarHelper::[...]** are used to generate the buttonhole for the functions of sorting, editing and creating new records.
2. The code $data = $this->get('Data') invokes the method getData() in the file /components /com_mcm/models/XXXlist.php (class McmModelXXXlist), this method makes the query on the db and returns a recordset as an array of objects, in the Joomla! standard.
3. the code $this->assignRef('data', $data) creates the property $this->;data within the class McmViewXXXlist so that such information is easily accessible from the template.
4. the code parent::display($tpl) invokes and executes the file /components/com_mcm/views /XXXlist/tmpl/default.php that displays the contents of the recordset.

Listing records

Within the file **/components/com_mcm/views/XXXlist/tmpl/default.php** yuo will find two areas marked by the delimiters *"<! - Joomla!Component Builder - begin code ->* "and" *<! - Joomla!Component Builder - end code ->* ";these are placed in the header of the table and within the table itself. In these areas, Joomla Component Builder writes the fields retrieved from the table in the db so they are viewed:/edited. Fields in this View are selected during component creation in fields table editing ('Show in List'& 'Show in Edit').

In the file /components/com_mcm/views/XXXlist/tmpl/default.php you will find a code like this:

```
<?php
 $k = 0;
 for ($i = 0, $n = count ($this->rows); $i <$n, $i++) (
 $row = &$this->rows[$i];
 $checked = JHTML ::_('grid.id', $i, $row->id);
 $published = JHTML ::_(' grid.published ', $row, $i);
 $link = JRoute:: _ ( 'index.php?option=com_mcm&DC=XXX&task=edit&cid[]='.
$row->id);
?>
[... html code for displaying the adminList, or all records in the table]
<?php
 $k = 1 - $k;
 )
?>
```

The **for** loop iterates through all the records retrieved by the method getData(), present in the model, and assigned to $this->rows by the view with assignRef(); the previous lines of code used to create objects standard adminList for Joomla (checkbox, icon publication, et cetera)

The code **$link = JRoute::_("...")** creates the URL to edit the individual records.

Searching records

Marco's Component Maker for Joomla! writes the code for searching into the fields selected during component creation (in fields table editing: 'Search' checkbox) .
see: */administrator/components/com_mcm/models/contentlist.php*

The 'where condition' is built in _buildQueryWhere() methods.

Anyway you get a javascript message inviting you to check the code.
see: */administrator/components/com_mcm/views/contentlist/tmpl/default.php*

on line 11 remove the *alert('remenber to update _buildQueryWhere funcion in model!!');*

```
<button onclick="this.form.submit();"><?php echo JText::_( 'GO' ); ?></button>
```

Next Step

Since version 1.5.0 Marco's Component Maker for Joomla! writes codes fully functional out of the box, anywhere, here and there, adjustments are needed (ie: to manage select list and relation): in the next article we will see how to do this.

Aggiungi commento

E-Mail

Sito web

1000 caratteri rimasti

Aggiorna

Invia

JComments

Copyright ©Marco Maria Leoni Web Consulting P.IVA 13089190154. - All Rights Reserved.