Scandiweb

Training tasks : Magento extension

1st Development task

Commenting magento extension

Introduction

The second educational extension development task will let you go deeper in the Magento module development. This extension is called “Commenting extension”. After developing the extension you will learn to :

- Calls to models
- Filtering of collections
- Validation of entered data with default Magento tool
- Appending Magento DB with custom tables, usage of model “resources”
- Transfer of data, error/success message
- Adding elements to Adminhtml

Functionality

Extension flow should be as follows:

① Create Magento extension and name it “Scandi_ProductComments”.

② Add new product attribute in Magento backend.

A) Attribute type - ‘yes/no’, should be added by extension automatically, attribute code - “commenting”, attribute label - “Allow comments”. Attribute should be placed under `General` tab.

③ Products that are allowed for commenting will be displayed in a list.

A) Extension output should be put in a separate block.

B) Block should be added to a separate CMS Page layout update. This way you will allow Magento admin to easily change page URL.

④ When user opens product with enabled commenting, he sees product view with list of approved comments and a form to add new comment: name, e-mail, and text.

A) Each field in comment form is validated - so you can not enter a@a.v as your email, for example, also make check for special characters in each field. (It can be managed easy with Zend Validation class, search for it)

B) If entered data is valid and comment has been successfully added, user should see success message after submitting his comment, otherwise display error message.

C) Submitted comment is being stored in a separate table in the database, which belongs to Your module.

⑤ All comments are shown in the Magento backend, in separate menu tab under Scandi->Product Comments.
(Remember to set correct title for browser window, don't let it be like Magento Admin, it should be Product Comments / Scandi / and also set Scandi menu item as active)

A) Grid title - "Product Comments". Export action is not required. Default sort is on date, to view last added comments.

B) Grid should function as default, e.g. all columns sort-able / filterable, option to select row quantity per page, pager.

C) Grid has following fields:

1. Date - date when comment was added.
2. Product Name - name of product comment was added to.

   (Remember that it should be stored in database like product_id, and then value should be joined to collection. Why not rendered? Fields are sorted by column index (e.g. by product_id) so it will be sort by numbers but we need in alphabetical order, but if we join field product_name then we can set column index to product_name and sorting will work correct. To make filter possible you should add filter_condition_callback for this column).

3. Author Name - Name Surname of customer/user who added comment.
4. Comment - This is comment text (please note, that there might be large comments so take care, that it does not break the grid layout). This field doesn't needs to be filterable/sort-able.
5. Status - This is status of comment. Filter type select, values - Approved, Not Approved.

D) Extension should allow two mass-actions: change status (approve/don't approve) and delete.

E) Grid row click leads to selected comment edit page.

1. This page contains buttons:
   - Back - Leads back to grid.
   - Delete - Deletes comment.
   - Save and Continue - Saves comment and stays at current edit page.
   - Save - Saved comment and leads back to comments grid.
2. Fields for this page
   - Product name - Input type text field, product_id should be entered, please verify that product with this id exists
   - otherwise, display error message.
   - Customer name - Input type text field. Required.
   - Customer email - Input type text field. Required.
   - Comment - Textarea type field. Required.
   - Status - Select type field with values Approved not Approved. Default value Not approved. Required.
3. Remember to validate all fields and add success/error messages.

F) Add new button allows to create new comment for product. Fields are same as Comment edit page. Only difference, when comment is being created from admin, Customer name and Customer email should be automatically filled by admin data and disabled. Remember to make sure that fields were not changed, you can verify it in controller.

Technical requirements

Developed extension must meet Scandiweb.com - Magento extension development guidelines. Extension installation should be done by unpacking extension archive to Magento root folder. Extension should not overwrite any existing files.

Remember

All error/success/notification messages should be added to session.

2nd Development task

Add "NAME" field to Magento newsletter

General task

Add "name" field to Magento newsletter subscription block.

How it works

1. Create Magento extension in community code pool. Name it "Scandi_Newsletter".
2. Create sql setup script to add "name" column to database. Column should be VARCHAR(50) NOT NULL. Also, remember, that You are not allowed to hardcode table name, because on different installations there can be table prefixes. So use getTableName() method.
3. Create new template file for newsletter subscription block (do not use/edit existing newsletter/subscribe.phtml file!). When you install extension, this template will replace the Default newsletter template, and store owner will not need to add anything by him/herself. Add it under app/design/frontend/base/default, in this way Magento will always find your template.
4. Create an event observer which will add subscriber name. (Remember, that you have to put some validations, js validation can be disabled by console, so don't count on it very much. Also remember special chars. For example, restrict names like this - <script>location.reload();</script>)
5. Add column "Name" to newsletter subscriber grid in BE Newsletter->"Newsletter Subscribers" by using Event Observers, it should remain all default functionality e.g. filter/sort.

3rd Development task

Magento extension - Beginner

General task

Display subcategories of a specified a category inside the static block

How it works

1. Create Magento extension which will allow us to display subcategories of a specified category inside of any static block or cms page. Name it "Scandi_CategoryList".
2. Subcategory display will be called in the static block via block code which will look similar to this: {{block type="categorylist/list" category_id="7"}}
3. Inside this code it should be possible to change category ID whose subcategories we need to show.
4. The output of categories should be structured in a list

```
<div class="menu_categories menu_block">
      <h3><a href="http://www.projecturl.com/parent_category.html" title="Parent
      category">Parent category</a></h3>
        <ul>
            <li><a href="http://www.projecturl.com/parent_category/subcategory1.html"
            title="Subcategory 1">Subcategory 1</a></li>
            <li><a href="http://www.projecturl.com/parent_category/subcategory2.html"
            title="Subcategory 1">Subcategory 2</a></li>
            <li><a href="http://www.projecturl.com/parent_category/subcategory3.html"
            title="Subcategory 1">Subcategory 3</a></li>
            <li><a href="http://www.projecturl.com/parent_category/subcategory4.html"
            title="Subcategory 1">Subcategory 4</a></li>
            <li><a href="http://www.projecturl.com/parent_category/subcategory5.html"
            title="Subcategory 1">Subcategory 5</a></li>
            <li><a href="http://www.projecturl.com/parent_category/subcategory6.html"
            title="Subcategory 1">Subcategory 6</a></li>
            <li><a href="http://www.projecturl.com/parent_category/subcategory7.html"
            title="Subcategory 1">Subcategory 7</a></li>
            <li><a href="http://www.projecturl.com/parent_category/subcategory8.html"
            title="Subcategory 1">Subcategory 8</a></li>
            <li><a href="http://www.projecturl.com/parent_category/subcategory9.html"
            title="Subcategory 1">Subcategory 9</a></li>
        </ul>
</div>
```

Remember

Extension does not require any backend configuration or menu tabs and these should not be included.