Exercise Solutions

Section 4 Lesson 3. Design Layout

Exercise 1: Register a Module Layout XML & Add a Block to the Page's Content Area

Use your IDE to do this exercise.

Step 1. Register a layout XML file for your custom module

a) Add the following XML structure to the `Training_Practice` *config.xml* file.

```
<?xml version="1.0"?>
<config>
<frontend>
<layout>
<updates>
<practice module="Training_Practice">
<file>practice/layout.xml</file>
</practice>
                    </updates>
                </layout>
            </frontend>
</config>
```

Note: The `<practice>` node need only be a unique string, but the `<file>` node must be named "file". Also note that the layout file is declared without any reference to package or themes.

b) Create the file *app/design/frontend/base/default/layout/practice/layout.xml*.

Step 2. Verify that it is loaded by creating invalid XML in the layout file and reloading

- Add the following XML structure to the layout file and reload to see a parse error.

Note: Developer mode must be enabled to see the XML parse error in the browser.

```
<?xml version="1.0"?>
<layout>
<default>
<broken>
            </broken___>
      </default>
</layout>
```

Step 3. Add a `core/template` block as a child to the "content" block on every page, usingthe `default` handle

a) Amend the previous content to the following.

```
<?xml version="1.0"?>
<layout>
  <default>
    <reference name="content">
      <block type="core/template" name="new.block" before="-">
        <action method="setTemplate">
          <path>practice/one.phtml</path>
        </action>
      </block>
    </reference>
  </default>
</layout>
```

b) Create the file *app/design/frontend/base/default/templatepractice/one.phtml* and add some markup for display.

```
<div style="background:green; color:white;">
  <p>
    This block should display everywhere!
  </p>
</div>
```


c) Refresh the layout cache, if enabled, and visit numerous pages in the system. In a "vanilla" Enterprise Edition installation the `new.block` block should be displayed at the top of the content area.

Exercise 2: Specify a Page-Dependent Block Value Via Layout & Display It in the Template

Use your IDE to do this exercise.

Step 1. Use an action directive to set the attribute `background_color` on the block from the previous exercise to "blue"

- Change the layout update file from the previous exercise, adding the content in bold.

  Blocks extend from `Varien_Object`, which allows the `background_color` attribute to be set using magic setters.


```
<?xml version="1.0"?>
<layout>
  <default>
    <reference name="content">
      <block type="core/template" name="new.block" before="-">
        <action method="setTemplate">
          <path>practice/one.phtml</path>
        </action>
        <action method="setBackgroundColor">
          <value>blue</value>
        </action>
      </block>
    </reference>
  </default>
</layout>
```

Step 2. In the template, create a `<div>` with some content and set the background color style to the value set on the block

a) Edit the template from the previous exercise, replacing its content with the following content.

```
<?php
$_bgColor = $this->hasBackgroundColor() ? "background-color: {$this-
>getBackgroundColor()};" : '';
?>
<div style="<?php echo $_bgColor ?> color:white;">
  <p>
    This block should be displayed everywhere!
  </p>
</div>
```

b) Refresh any page and verify that the block now has a blue background.

Step 3. On the product detail page, set the block's `background_color` attribute to "red" using layout XML

a) Change the layout update file from the previous exercise by adding a layout update handle with the following content.


```
<?xml version="1.0"?>
<layout>
      <default>
            <!-- snip... -->
      </default>
      <catalog_product_view>
            <reference name="new.block">
                  <action method="setBackgroundColor">
                        <value>red</value>
                  </action>
            </reference>
  </catalog_product_view>
</layout>
```

b) Browse both product and non-product pages to ensure that the background color is changing accordingly.

Exercise 3: Set Up Your Own Layout Update Handle

Use your IDE to do this exercise.

Step 1. Create your own unique handle in the layout XML file

Note: This handle must not match any existing pattern. For example: `[module]_[controller]_[action]` or `STORE_[storecode]`

- Change the layout update file from the previous exercise by adding the custom layout handle.

```
<?xml version="1.0"?>
<layout>
      <!-- snip... -->
      <custom_handle>

      </custom_handle>
</layout>
```


Step 2. Add an output block of the type `core/text` to the handle

- Change the layout update file from the previous exercise by adding the content below in bold.

```
<?xml version="1.0"?>
<layout>
      <!-- snip... -->
      <custom_handle>
      <block name="customBlock" type="core/text" output="toHtml">

      </block>
</custom_handle>
</layout>
```

Step 3. Set some text on the block using an `action` directive

- Change the layout update file from the previous exercise by adding the `action` node calling the `setText()` method to populate the `core/text` block.

```
<?xml version="1.0"?>
<layout>
      <!-- snip... -->
      <custom_handle>
      <block name="customBlock" type="core/text" output="toHtml">
            <action method="setText">
                  <text>This is some text.</text>
            </action>
      </block>
</custom_handle>
</layout>
```

Step 4. In your controller, add a new action and call `$this->loadLayout(YOUR_HANDLE)->renderLayout()`

- Add the following method to the index controller class from the `Training_Recap` module.

```
public function handleAction()
{
      $this->loadLayout('custom_handle')->renderLayout();
}
```


Step 5. Verify that your block is displayed and explain the result

- Browse to the appropriate URL `/recap/index/handle` and verify that your block is displayed.

Exercise 4: Move an Existing Block

Use your IDE to do this exercise.

Step 1. Choose a category page with 3 columns and enabled layered navigation (for example, the Furniture category from the sample data)

- Visit any category page with a 3-column layout, where the layered navigation is active. The “Is Anchor” setting in the category management page specifies the layered navigation setting.

Step 2. Move the poll block from the right column to the left block above the layered navigation using your module’s layout XML

a) Create a `catalog_category_layered` layout update handle in the `Training_Practice` layout update file.

```
<?xml version="1.0"?>
<layout>
      <!-- snip... -->
      <catalog_category_layered>

</catalog_category_layered>
</layout>
```

b) Unset the parent-child relationship between the poll block and the right column.

```
<?xml version="1.0"?>
<layout>
      <!-- snip... -->
      <catalog_category_layered>
      <reference name="right">
            <action method="unsetChild">
                  <block>right.poll</block>
            </action>
      </reference>
</catalog_category_layered>
</layout>
```


c) Create a parent-child relationship between the poll block and the left column.

```
<catalog_category_layered>
<reference name="right">
      <action method="unsetChild"><block>right.poll</block></action>
</reference>
<reference name="left">
      <action method="insert"><block>right.poll</block></action>
</reference>
</catalog_category_layered>
```

Step 3. Be sure to use the existing block instance to assure maximum upgradability

- It would be possible to `<remove>` the block from the right and re-create it in the left using a different name. However, using the approach from the previous step ensures that any other references to this block by name will still continue to work.

Step 4. Describe any difficulties you encountered and explain how you solved them

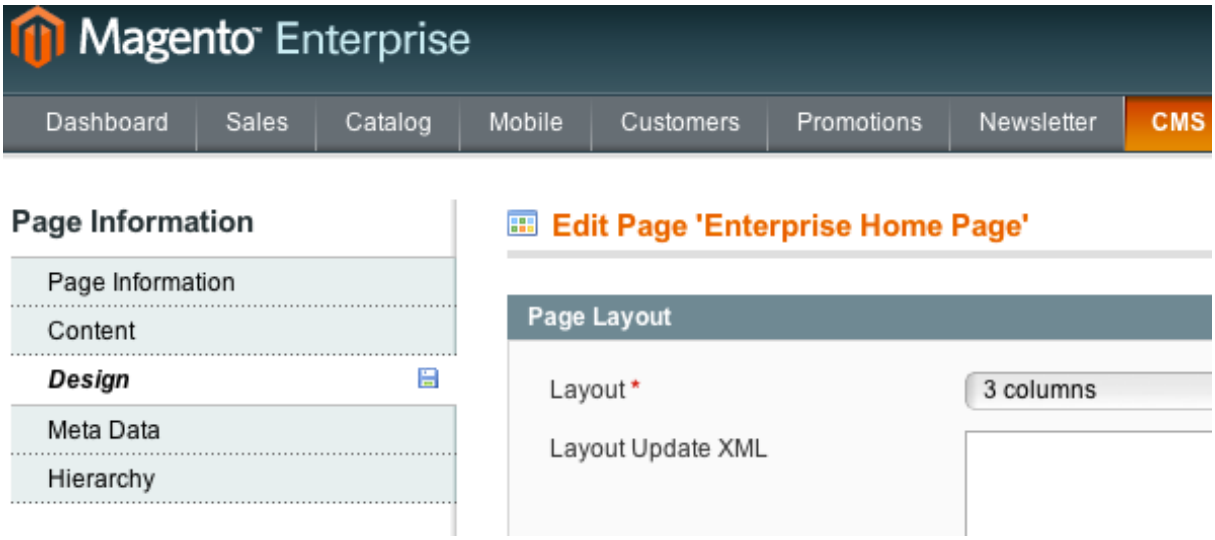
- Discuss the things you discovered or learned during this exercise with your fellow students.

Exercise 5: Make Layout Updates Work on Different Pages

Use your IDE to do this exercise. This exercise builds on the previous exercise.

Step 1. Make the home page use a 3-column layout

- In the Admin interface, navigate to CMS > Pages > Manage Content. Select the "enterprise-home" page, and change its layout in the Design tab.



Step 2. Make the poll appear at the top of the left column

- Change the layout update file from the previous exercise, adding the content in bold to a `cms_index_index` update handle.

```
<?xml version="1.0"?>
<layout>
      <!-- snip... ->
      <cms_index_index>
            <reference name="left">
                  <action method="insert">
                        <block>right.poll</block>
                  </action>
      </reference>
</cms_index_index>
</layout>
```

Step 3. Once that works, return to the category page with an enabled layered navigation

- Notice that there are now two poll blocks in the left column: one at the top and one under the layered navigation.

Step 4. Ensure that you don't have two poll blocks in the left column

a) Update the layout XML file so it matches the following code.

```
<?xml version="1.0" encoding="UTF-8"?>
<config>
      <default>
            <update handle="move_poll_to_left"/>
      </default>
      <catalog_category_layered>
            <update handle="move_poll_to_left"/>
      </catalog_category_layered>
      <catalog_category_default>
            <update handle="move_poll_to_left"/>
      </catalog_category_default>

      <move_poll_to_left>
            <reference name="right">
                  <action method="unsetChild">
                        <alias>right.poll</alias>
                  </action>
            </reference>
            <reference name="left">
                  <action method="unsetChild">
                        <alias>right.poll</alias>
                  </action>
            </reference>
            <reference name="left">
                  <action method="insert">
                        <name>right.poll</name>
                        <sibling>catalog.leftnav</sibling>
                        <after>0</after>
                  </action>
            </reference>
      </move_poll_to_left>
</config>
```

b) Reload the category page and confirm that now there is only one poll block above the layered navigation.

