Exercise Solutions

Section 4 Lesson 1. Templates

Exercise 1: Create the Directory for a New Theme

Use your IDE to do this exercise.

Step 1. Create your own theme named `custom` in the `enterprise` package

a) Create the directory *app/design/frontend/enterprise/custom*

b) Create the directory *skin/frontend/enterprise/custom*

Step 2. Create the *template*, *layout*, *css*, *images* and *js* folders in the appropriate directory branches

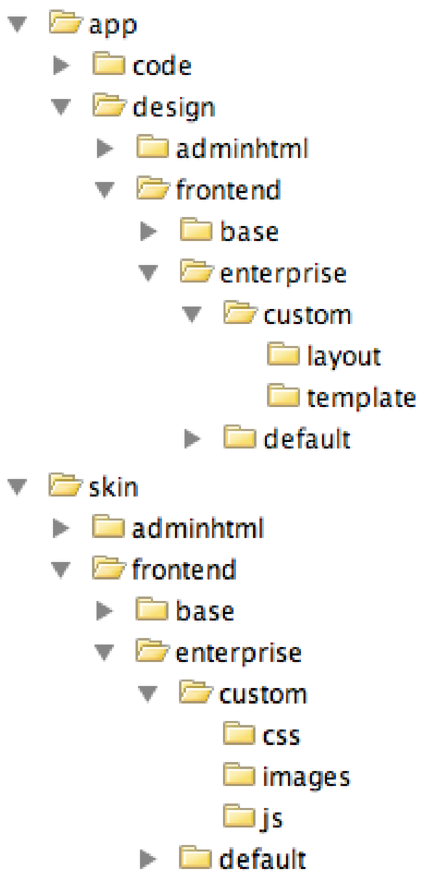
a) Create the directory *app/design/frontend/enterprise/custom/layout.*

b) Create the directory *app/design/frontend/enterprise/custom/template.*

c) Create the directory *skin/frontend/enterprise/custom/css.*

d) Create the directory *skin/frontend/enterprise/custom/images.*

e) Create the directory *skin/frontend/enterprise/custom/js.*



Step 3. (Optional) Create the local XML and CSS theme files

a) Create the file *app/design/frontend/enterprise/custom/layout/local.xml.*

b) Create the file *skin/frontend/enterprise/custom/css/local.css.*

Exercise 2: Register Your Custom Theme

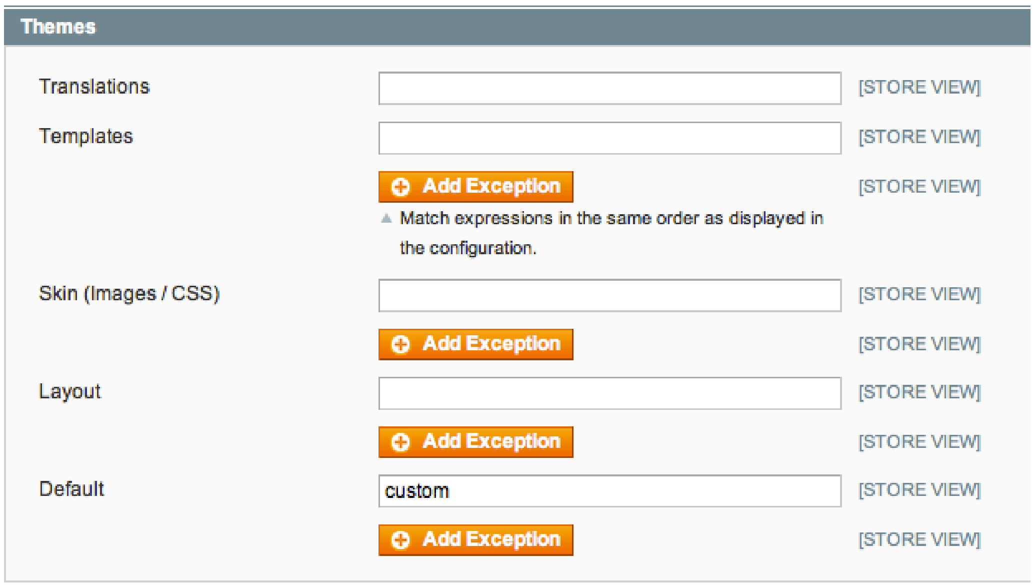
Step 1. Register your custom theme in System > Configuration > Design > Themes > Default

a) Log in to the Admin interface.

b) Navigate to System > Configuration in the top menu.

c) Click the Design tab on the left, and then click the Themes label bar to expand it.

d) In the Default box, enter `custom`, the name of your theme.



e) Save the new setting.

Step 2. In the `custom` theme, override the `page/1column.phtml` template

a) Create the directory *app/design/frontend/enterprise/custom/template/page.*

b) Copy the file *app/design/frontend/enterprise/default/page/1column.phtml* to *app/design/frontend/enterprise/custom/template/page/1column.phtml.*

Step 3. Insert this code into the content area of the copied template so that it looks different from the original

- <h2><?php echo $this->getTemplateFile() ?></h2>

a) Open the file *app/design/frontend/enterprise/custom/template/page/1column.phtml.*

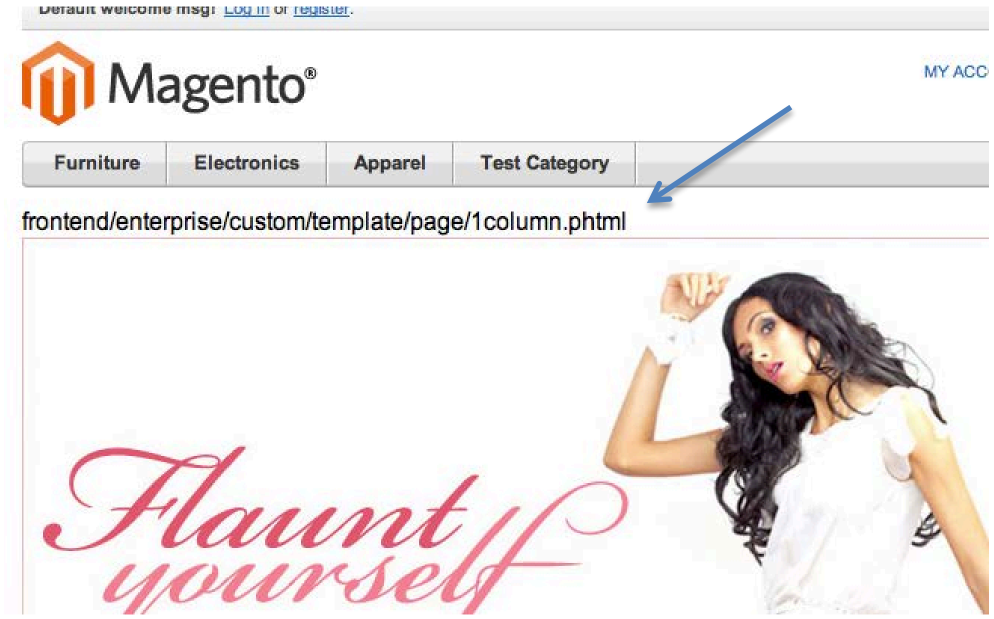
b) Update the content of the main column so that it looks like this.

```
<div class="main col1-layout">
  <div class="col-main">
    <h2><?php echo $this->getTemplateFile() ?></h2>
    <?php echo $this->getChildHtml('content') ?>
  </div>
</div>
```

Step 4. Check that your `custom` theme is being used

a) Reload the home page.

b) Look for the template name that should be displayed above the content and check that the template in the custom theme is being used.



frontend/enterprise/custom/template/page/1column.phtml

Exercise 3: Create a New Package

Step 1. Create and register a new package and name it `training`

a) Create the directory *app/design/frontend/training.*

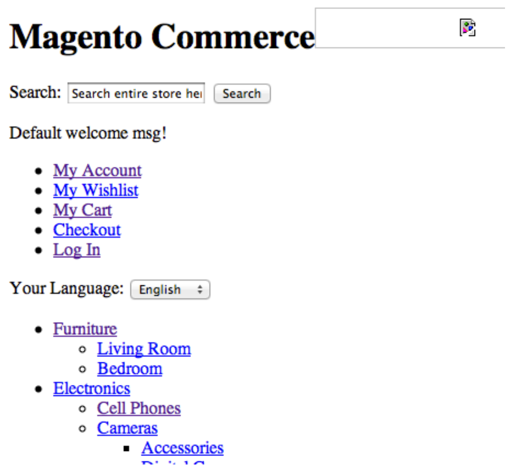
b) Create the directory *skin/frontend/training.*

c) In the Admin interface, navigate to System > Configuration and again click the Design tab, and then open the Package options.

d) In the input labeled “Current Package Name” enter `training`, the name of your theme.

Step 2. Verify that the `base/default` is used

- Reload the home page.



Magento Commerce

Search: Search entire store her Search

Default welcome msg!

- My Account
- My Wishlist
- My Cart
- Checkout
- Log In

Your Language: English

- Furniture
  - Living Room
  - Bedroom
- Electronics
  - Cell Phones
  - Cameras
    - Accessories

The `base/default` theme is used because the fallback process can find no theme in the training package. It contains no CSS declarations, so the page looks broken.

Step 3. Copy the `enterprise/default` theme to `training/default`

- Copy the directory *app/design/frontend/enterprise/default* to *app/design/frontend/training/default* with all subdirectories.

Step 4. Verify that the `training/default` theme is used

- Reload the home page. Notice that it looks just like the original enterprise theme.

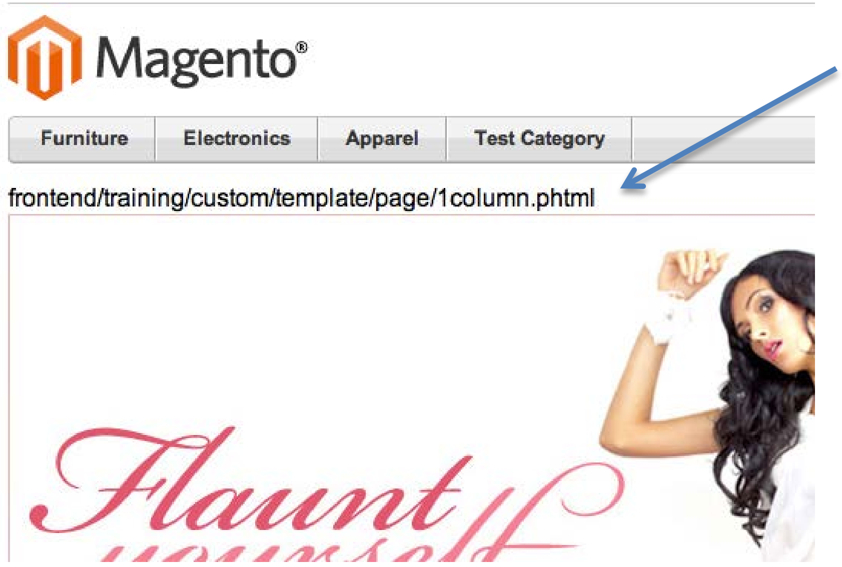
Step 5. Copy the `enterprise/custom` theme to `training/custom`

- Copy the directory *app/design/frontend/enterprise/custom* to *app/design/frontend/training/custom* with all subdirectories.

Step 6. Verify that your `custom` theme is used again

a) Reload the home page.

b) Notice that the template path is shown again, and this time it confirms that the *page1.phtml* template from the *training/custom* theme folder is being rendered.



frontend/training/custom/template/page/1column.phtml

Exercise 4: Create a Template-Specific Fallback Theme

Step 1. Create a new theme named `custom2` in the training package

a) Create the directory *app/design/frontend/training/custom2.*

b) Create the directory *skin/frontend/training/custom2.*

Step 2. Copy *page/1column.phtml* from the `custom` theme to `customer2`

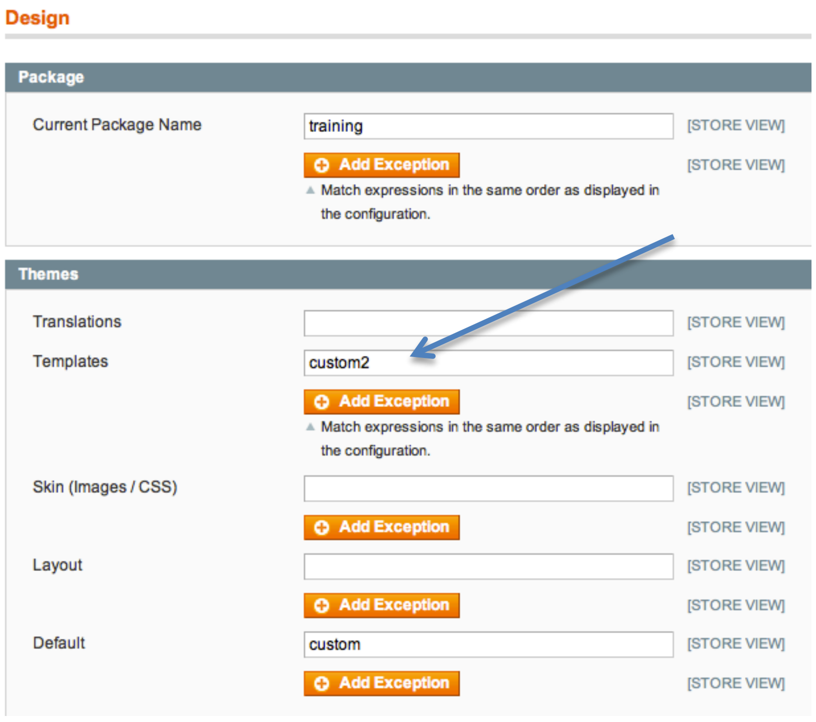
a) Create the directory *app/design/frontend/training/custom2/template* and *app/design/frontend/training/custom2/template/page.*

b) Copy *app/design/frontend/training/custom/template/page/1column.phtml* to *app/design/frontend/training/custom2/template/page/1column.phtml.*

Step 3. Register your `custom2` theme for templates only

a) In the Admin interface, navigate to System > Configuration and again click the Design tab, and then open the Themes options.

b) In the Templates box, enter `custom2`.



Current Package Name	training	[STORE VIEW]
	Add Exception Match expressions in the same order as displayed in the configuration.	[STORE VIEW]
Translations		[STORE VIEW]
Templates	custom2	[STORE VIEW]
	Add Exception Match expressions in the same order as displayed in the configuration.	[STORE VIEW]
Skin (Images / CSS)		[STORE VIEW]
	Add Exception	[STORE VIEW]
Layout		[STORE VIEW]
	Add Exception	[STORE VIEW]
Default	custom	[STORE VIEW]
	Add Exception	[STORE VIEW]

c) Save the settings.

Step 4. Determine the loading order of templates

- Reload the home page and check which *1column.phtml* is being used.

Exercise 5: Configure a Different Theme for Different Stores

Step 1. In the system configuration, switch to a store view scope

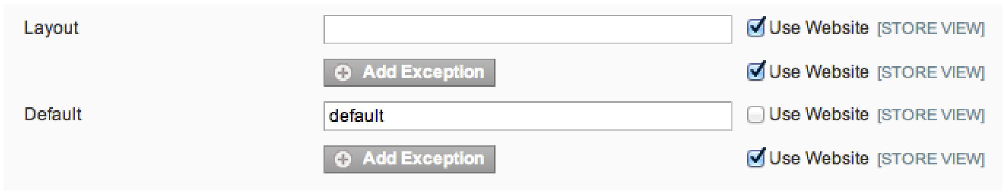
a) In the Admin interface, navigate to System > Configuration and, once again, click the Design tab.

b) In the dropdown list at the top left, choose the store view “German”.

Step 2. Set the value for the default theme to `default`

a) Clear the checkbox labeled “Use Website” beside the field “Default”.

b) In the field, change the value from `custom` to `default`.

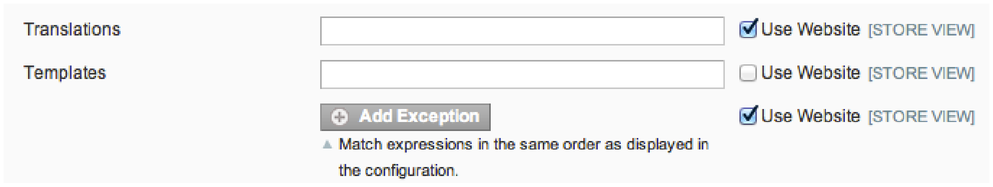


Layout		✓ Use Website [STORE VIEW]
	Add Exception	✓ Use Website [STORE VIEW]
Default	default	☐ Use Website [STORE VIEW]
	Add Exception	✓ Use Website [STORE VIEW]

Step 3. Clear the template-specific theme setting

a) Clear the checkbox labeled “Use Website” beside the field “Templates”.

b) Clear the field “Templates”.



Translations		✓ Use Website [STORE VIEW]
Templates		☐ Use Website [STORE VIEW]
	Add Exception ▲ Match expressions in the same order as displayed in the configuration.	✓ Use Website [STORE VIEW]

c) Save the changes to the configuration.

Step 4. To verify on the frontend that it works, change the store in the store switcher and check that different templates are used

a) Reload the home page in the English store view. If the English store view isn’t active, switch to it using the language switcher.

b) Check that the template *training/custom2/template/page/1column.phtml* is used.

c) Switch to the German store view on the home page.

d) Check that the *training/default/template/page/1column.phtml* template is used.

Because no debugging output is included in the template file, you can use template path hints to confirm that correct template file really was used.

Exercise 6. (Optional) Add “Our price” to Price Template to Be Displayed Wherever Price Is Rendered

Step 1. Open the `price.phtml` template

a) Using template path hints, find the price template that is used for rendering.

b) Copy it to the `custom` theme.

c) Open the file in your IDE.

Step 2. Analyze the template structure

- Find out which parameters specify how the price will be rendered.

Step 3. Add the custom label inside the condition that is used

- You can either add the “Our price” label inside each condition, or find out which ones are used for your instance and just add them there.

Step 4. Verify that your message appears in every place where the price is rendered

- View a couple of categories with product listings and a couple of product detail pages of products with different types (configurable products, grouped products, simple products). Verify that your custom label is rendered correctly for each variant.