Exercise Solutions

Section 2 Lesson 9. Event Observers

Exercise 1: Register and Create an Empty Module

Use your IDE to do this exercise.

Step 1. Register an observer on product after load

- Add the event observer configuration XML to the file
  app/code/local/Training/Practice/etc/config.xml. This code must be inside the root
  `<config>` node.

```xml
<frontend>
  <events>
    <catalog_product_load_after>
      <observers>
        <training_practice>
          <type>model</type>
          <class>training_practice/observer</class>
          <method>catalogProductLoadAfter</method>
        </training_practice>
      </observers>
    </catalog_product_load_after>
  </events>
</frontend>
```

Step 2. Create the observer class and method

a) Add the model class group declaration, so Magento will be able to resolve the class ID `training_practice/observer`. This code must be inside the root `<config>` node.

```
<global>
  <models>
    <training_practice>
      <class>Training_Practice_Model</class>
    </training_practice>
  </models>
</global>
```

b) Create the `Model` directory for the `Training_Practice` module *app/code/local/Training/Practice/Model/*.

c) Create the observer class file *app/code/local/Training/Practice/Model/Observer.php.*
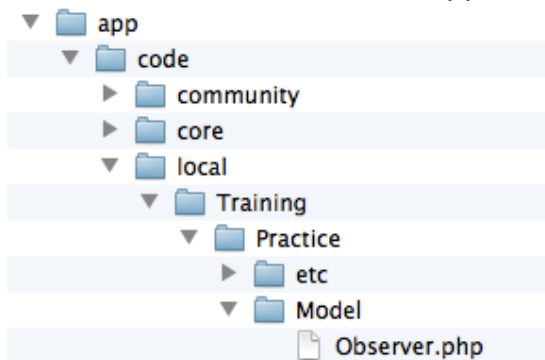
d) Add the class to the observer file.

```
class Training_Practice_Model_Observer
{
  public function catalogProductLoadAfter(Varien_Event_Observer $observer)
  {
  }
}
```

Step 3. In the observer method, set the product's special price

Note: If there is no product in your store, log in to the Admin interface now and create a test product, without specifying a special price. This is not covered by this exercise.

a) Fetch the product model from the `$observer` argument and set the special price attribute.

```
public function catalogProductLoadAfter(Varien_Event_Observer $observer)
{
  $product = $observer->getProduct();
  $product->setSpecialPrice($product->getPrice() -1);
}
```

b) Visit a product detail page on the frontend and check that the product indeed displays the special price.

Step 4. Make it work on the frontend/Admin only

a) First check that the current observer declaration works on the frontend only. Log in to the Admin interface to do so.

b) Navigate to Catalog > Manage Products and select a product from the list.

c) Check that there is no special price defined for the product.

d) Change the observer configuration so that it is contained in an `<adminhtml>` node instead of in the `<frontend>` node.

```
<frontend>
<adminhtml>
  <events>
    <catalog_product_load_after>
      <observers>
        <training_practice>
          <type>model</type>
          <class>training_practice/observer</class>
          <method>catalogProductLoadAfter</method>
        </training_practice>
      </observers>
    </catalog_product_load_after>
  </events>
</adminhtml>
</frontend>
```

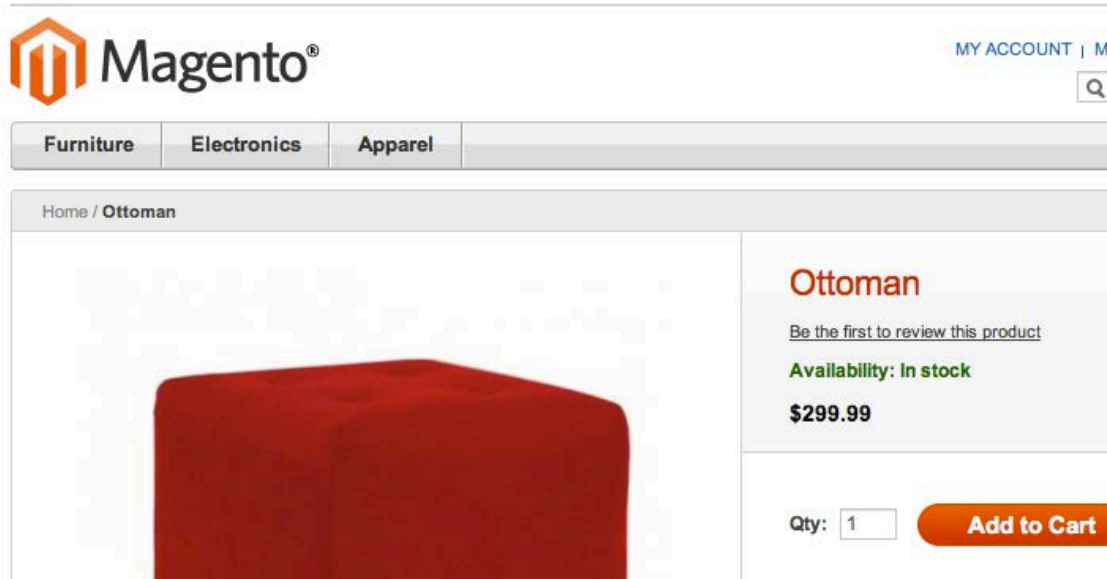e) Reload the Admin edit product page to check that the observer now is setting a special price.

f) Reload a frontend product detail page and confirm that the observer is no longer being executed, so that only the standard price is displayed.

Step 5. Make it work in both areas

a) There are two possible ways to do this: either specify the observer twice, once in `<frontend>` and once in `<adminhtml>`, or change the area node to `<global>`. For now, implement the latter version, because it avoids code duplication.

```
<adminhtml>
<global>
  <events>
    <catalog_product_load_after>
      <observers>
        <training_practice>
          <type>model</type>
          <class>training_practice/observer</class>
          <method>catalogProductLoadAfter</method>
        </training_practice>
      </observers>
    </catalog_product_load_after>
  </events>
</global>
</adminhtml>
```

b) Reload the Admin product edit page and check that the special price is still set.

c) Reload a frontend product detail page and check that the observer triggered again.

Step 6. Deactivate your observer by setting the `type` to `disabled`

a) Change the observer configuration so that the type node contains the value `disabled` instead of `model`.

b) Reload the frontend product detail page to confirm that the observer is no longer active in that area.

c) Reload the Admin edit product page to confirm that the observer is no longer active in the adminhtml area.