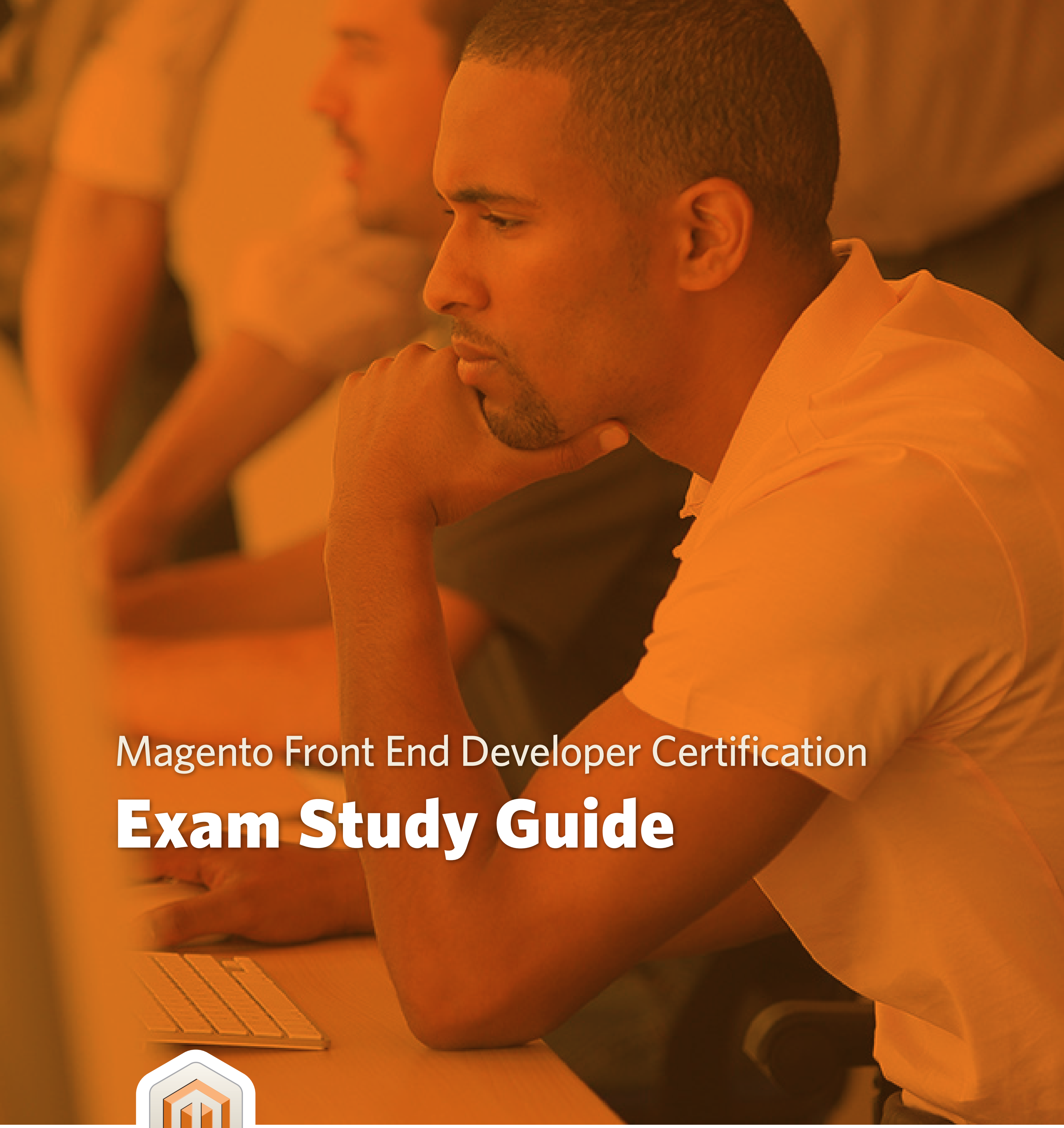

Magento Front End Developer Certification

Exam Study Guide

Magento Certified
FRONT END DEVELOPER

Magento® U

Table of Contents

Introduction.....1
Prerequisites.....2
Objectives (Skills).....3
1. Use the Magento Design Fallback System.....3
2. Use Layout XML to Customize a Theme.....4
3. Create and Customize Template Files.....5
4. Effectively Use the Magento Block-Template Design System.....5
5. Identify Where to Locate Files and Create New Files in the Theming-related Directory Structure.....6
6. Customize and Create JavaScript Within the Magento Framework.....7
7. Use CSS Effectively to Customize Magento Look and Feel.....7
8. Customize the Look and Feel of Specific Magento Pages.....8
9. Correctly Use the Admin Configuration Scopes (Default/Website/Store View Fallback) 10
10. Implement Internationalization of Frontend Pages (CE + EE).....10
Sample Questions.....12
Answers to Sample Questions.....14

Introduction

This Study Guide is intended to help you prepare for the Magento Certified Frontend Developer Exam, a multiple-choice exam that tests the skills needed by Magento Frontend Developers. The exam is based on Magento Community Edition v.1.7 and Magento Enterprise Edition v.1.12. Each of the 65 questions ("items") on the exam tests a skill that applies to *both* CE v.1.7 *and* EE v1.12. (Items are not edition-specific.)

This Study Guide details the objectives (skills) that are tested. For each objective, it provides questions you should ask yourself as part of your study, along with references to resources that will help you answer the questions.

Most of those references are to the Magento codebase and Admin interface, but some also point to external resources on the Web. In addition to the listed references, feel free to consult other Magento community resources that might provide additional insights.

Magento usually offers several ways to accomplish any given customization, each one with advantages and disadvantages. Often there is no single "right" way to do something: Specific circumstances determine which approach makes the most sense. Therefore, it is important to understand the pros and cons of each technique, in order to make educated choices.

Considerations to keep in mind include:

- Implementation time
- Upgradability
- Required depth of understanding

While studying, when having to choose one approach over another, two goals to keep in mind are optimal upgradability and maintainability.

This Study Guide is continually being revised and improved. When preparing for the exam, remember to check the website for the latest version of this Study Guide.

Prerequisites

The exam questions assume that you have mastered generic frontend developer skills such as HTML, markup validation, CSS2, and plain JavaScript. Your JavaScript knowledge should include variable scope, scope binding, closures, monkey patching, and OOP.

Other related concepts you should know include compressing and merging CSS and JavaScript, image spriting, and other performance-related frontend development techniques.

To check if you have mastered these skills, ask yourself the following questions:

- Which new features are included in HTML5?
- How can I introduce CSS into an HTML page?
- What is the scope of variables in JavaScript?
- How do closures work in JavaScript?
- What methods of inheritance implementation can I name in JavaScript? How do they work?
- Which methods of page speed optimization do I know?

References:

- http://www.w3schools.com/HTML/default.asp
- http://www.w3schools.com/js/default.asp
- http://www.seomoz.org/blog/15-tips-to-speed-up-your-website

Objectives (Skills)

1. Use the Magento Design Fallback System

This section tests how well you can utilize the Magento fallback system to implement different use cases, such as single- and multi-store Magento instance theming, temporary design changes, etc.

This topic comprises approximately 7% of the exam. Questions are drawn randomly from the following topics and objectives:

- 1.1 Create custom themes
- 1.2 Configure the design fallback using the options found in the Admin panel under System > Configuration > Design
  - 1.2.1 Design Package
  - 1.2.2 Default Theme
  - 1.2.3 Type-specific Themes
  - 1.2.4 Design exceptions
- 1.3 Apply a temporary theme configuration to a store view using the options found in the Admin panel under System > Design
- 1.4 Understand the differences and similarities between System > Configuration > Design and System > Design to configure the design fallback

Questions to ask yourself while studying:

- How can I display templates from all available fallback steps within one page?
- When implementing a custom theme based on the default/blank theme, how can I minimize work during upgrades of Magento and the blank theme?
- How many fallback steps are maximally available?
- How does choosing a theme on a category, product, or CMS page influence the fallback configuration?

References:

- The Magento Admin interface
  - System > Design
  - System > Configuration > Design
  - Catalog > Manage Products > [choose product] > Design tab
  - Catalog > Manage Categories > [choose category] > Custom Design tab
  - CMS > Pages > [choose page] > Design tab
- The Magento Designers Guide: http://www.magentocommerce.com/design_guide
- http://www.magentocommerce.com/knowledge-base/entry/magentos-theme-hierarchy

2. Use Layout XML to Customize a Theme

Layout XML is a central part of Magento theming. This objective tests your knowledge of Layout XML syntax and how to apply it to solve specific problems. Even though they are few in number, layout XML tags offer an amazing amount of flexibility and functionality. Layout XML often provides the biggest learning opportunity for more graphics-oriented frontend developers.

This topic comprises approximately 19% of the exam. Questions are drawn randomly from the following topics and objectives:

2.1 Demonstrate knowledge of all layout XML directives (layout handles, block, reference, action, remove, update and label) and their arguments
2.2 Define an output block
2.3 Use a local.xml file
2.4 Understand layout merging
2.5 Understand processing order of layout handles and other directives
2.6 Set values on block instances using layout XML
2.7 Move a child block from one parent to another using layout XML
2.8 Specify the root template for all pages or for specific pages

Questions to ask yourself while studying:

- What is the difference between the `name` and `as` attributes for blocks?
- What is the processing order of layout XML instructions?
- How can I include a custom CSS file on a product detail page for a specific product?
- How can I include a custom CSS file on a product detail page for a specific product type (e.g. grouped, downloadable, configurable, etc.)?
- How can I include a custom CSS file on a CMS page?
  How can I include a custom CSS file on all Magento pages?
- How can you avoid copying core layout XML files into a custom theme by using the local.xml layout file?

References:

- The Magento Designers Guide:
  http://www.magentocommerce.com/design_guide
- http://www.magentocommerce.com/design_guide/articles/intro-to-layouts
- app/design/frontend/base/layout/page.xml
- app/design/frontend/base/layout/catalog.xml
- The Magento Admin interface
  - Catalog > Manage Products > [choose product] > Design tab
  - Similar options for categories and CMS pages

3. Create and Customize Template Files

Template files are responsible for the visual representation of page content. In order to work with Magento templates efficiently, you should not only understand how they relate to layout XML and the fallback, but also know how to use basic PHP. A frontend developer must also be familiar with Magento's own set of conventions to be used in templates.

This topic comprises approximately 16% of the exam. Questions are drawn randomly from the following topics and objectives:

3.1 Assign a customized template file using layout XML
3.2 Override a native template file with a customized template file, using the design fallback
3.3 Describe conventions used in template files:
3.3.1 Variable naming (_ prefix for variables declared in a template)
3.3.2 PHP loop and block constructs (for example: `if (…): endif;`)
3.3.3 Opening and closing PHP tags (`<?php` vs `<?`, `<%` and others)

Questions to ask yourself while studying:
- What are the conventions for variable naming?
- What does `$this` refer to in template files?
- How can I include another template inside the current one?

References:
- The Magento Designers Guide: http://www.magentocommerce.com/design_guide

4. Effectively Use the Magento Block-Template Design System

To fully utilize Magento's design system, it is necessary to grasp three major concepts—layout XML, templates, and blocks—and to understand how they relate to each other. The page content hierarchy is represented in all three of them. The parent-child associations always need to be considered. Blocks provide the functionality and content displayed by templates. This section asks you to demonstrate your knowledge of Magento blocks and how to use them to customize a Magento theme.

This topic comprises approximately 11% of the exam. Questions are drawn randomly from the following topics and objectives:

4.1 Declare a custom template using a core/template block via layout XML with a custom template, and output it as a child of a structural block or of a template block
4.2 Understand the differences between structural and content blocks
4.3 Assign a default root template on all pages

4.4 Assign a root template on a specific page
4.5 Demonstrate the use of core/text_list blocks

Questions to ask yourself while studying:

- What is the difference between a template and a block?
- How can I see which block methods are available in a template?
- How can I assign a child block to another block?
- How do I display a child block?
- How can I access data from a parent block?
- How can I set data on a block via layout XML, and how do I display that data in a template?

References:

- The Magento Designers Guide: http://www.magentocommerce.com/design_guide
- http://www.sitepoint.com/an-introduction-to-magento-themes/

5. Identify Where to Locate Files and Create New Files in the Theming-related Directory Structure

Especially for developers new to Magento, the deep folder hierarchies and hundreds of files in the directory structure may be confusing. This section of the exam asks you to demonstrate your understanding of the underlying logic of the structure and your ability to navigate efficiently through the directories.

This topic comprises approximately 7% of the exam. Questions are drawn randomly from the following topics and objectives:

5.1 Describe the contents of the app/design/ and skin/directory branches and why they are separated
5.2 Describe the general directory hierarchy format under app/design/ and skin/ [area]/[package]/[theme]/[filetype]/[module-or-namespace]/

Questions to ask yourself while studying:

- What is the purpose of packages?
- What is the purpose of the theme fallback?
- What is the purpose of separating the skin and the design files?

References:

- The Magento Designers Guide: http://www.magentocommerce.com/design_guide

6. Customize and Create JavaScript Within the Magento Framework

Many of Magento's functionalities work with JavaScript. This starts with form validation and ends with more complex pages like the one-page checkout. Prototype provides a framework to cleanly extend and modify core functionality without having to modify core files or revert to theme fallback overrides. To demonstrate mastery of this topic, you also need to understand how to include custom JavaScript files using layout XML.

This topic comprises approximately 8% of the exam. Questions are drawn randomly from the following topics and objectives:

6.1 Include custom JavaScript on all pages
6.2 Include custom JavaScript on specific pages
6.3 Demonstrate understanding of using prototype.js
6.4 Override core JavaScript classes (without overriding core JavaScript files)
6.5 Configure JavaScript merging in the Admin panel

Questions to ask yourself while studying:

- How can I extend an existing JavaScript class instance with new or customized functionality?
- How many upgrade-safe ways can I find to execute custom JavaScript functions when a one-page checkout step is submitted?
- How can I add custom JavaScript form validation using the framework provided by Magento?
- How can I change the label text of the configurable product dropdowns in an upgrade-safe way?
- How do I use the JavaScript translation facility for custom text?

References:

- http://prototypejs.org/
- js/prototype/validation.js
- js/varien/product.js
- skin/frontend/base/default/js/opcheckout.js
- skin/frontend/base/default/js/bundle.js

7. Use CSS Effectively to Customize Magento Look and Feel

CSS is arguably the most important tool in the frontend developers' toolbox. Of course, Magento, being a web application, is no exception. This section tests your knowledge of how to apply custom CSS within Magento (of course with the help of Layout XML).

This topic comprises approximately 9% of the exam. Questions are drawn randomly from the following topics and objectives:

7.1 CSS2
7.2 CSS Merging
7.3 CSS Compression
7.4 Other CSS skills

Questions to ask yourself while studying:
- Which types of selectors and rules of priority are available in CSS2?
- Which CSS related methods of Page Speed Optimization do you know?

References:
- http://www.w3schools.com/css/default.asp
- http://www.seomoz.org/blog/15-tips-to-speed-up-your-website

8. Customize the Look and Feel of Specific Magento Pages

Besides the generic content framework that Magento's design provides, each content page has individual hooks, wrappers, and elements that are unique to that specific page. A good frontend developer needs to be familiar with all of them in order to work efficiently and using best practices.

This topic comprises approximately 12% of the exam. Questions are drawn randomly from the following topics and objectives:

8.1 Generic Page Elements
- Top Links
- Page Header and Footer
- Quick Search
- Store View (Language) and Store Switcher
- Mini Cart
- Breadcrumbs
- Sidebar menu

8.2 Product Detail Pages (including Simple, Configurable, Bundle, Grouped, Virtual, Downloadable, Gift Card products)
- Configure design changes (page layout)
- Use custom layout updates (XML)
- Use container blocks provided by native Magento to display additional information on category pages

8.3 Category Pages
- Configure design changes (page layout)
- Use custom layout updates (XML)
- Configure the layered navigation

- Configure design changes (page layout)
- Configure design inheritance
- Configure a CMS block as a category landing page

8.4 CMS Pages
- Configure design changes (page layout)
- Use custom layout updates (XML)
- Use static variables
- Understand the use of CMS template directives (var, store, block, ...)

8.5 Widgets
- Insert widgets
- Understand the differences among widget types
- Configure a widget instance

8.6 CMS Blocks
- Create and insert CMS blocks
- Understand the use of CMS template directives (var, store, block, ...)

8.7 Customer Account Pages
- Remove an item from the customer account navigation using layout XML
- Add a custom page to the customer account navigation using layout XML

8.8 One-page Checkout
- Use container blocks provided in the native Magento checkout to display additional information

8.9 Multi-address Checkout
- Use container blocks provided in the native Magento checkout to display additional information

8.10 Understand Customization of Transactional Email Templates
- Create and assign custom transactional email templates
- Use of template variables available in each email {store, var, ...) and how to access properties of variable objects (for example `var order.getCustomer.getName`)
- Link to custom images from transactional email templates
- Create links to store pages in transactional email templates

Questions to ask yourself while studying:
- Which structural blocks are available as part of the content on all pages?
- To which menus can I add custom entries using layout XML in an upgrade-safe way?
- How can I create links to store pages in transactional email templates?
- How can I include a custom template in transactional email templates?
- How can I access properties of the template variables (for example the order in the new order email, or the customer in the registration confirmation email)?

References:
- The Magento User Guide
  http://www.magentocommerce.com/resources/magento-user-guide

9. Correctly Use the Admin Configuration Scopes (Default/Website/Store View Fallback)

Even though it's a simple concept, admin configuration scopes is an essential idea to understand. This section of the exam asks you to demonstrate the ability to apply your knowledge of the configuration scopes to everyday tasks.

This topic comprises approximately 6% of the exam.

Questions to ask yourself while studying:
- Why are some options not configurable on a store view level?
- How can I use the available configuration scopes to the greatest effect?

References:
- The Magento admin interface
    - All settings under System > Configuration

10. Implement Internationalization of Frontend Pages (CE + EE)

Almost every project requires some degree of internationalization. The localization tools also can be used as a tool to customize content.

This topic comprises approximately 4% of the exam. Questions are drawn randomly from the following topics and objectives:

10.1 Create and change translations
 10.1.1 Understand the prioritization of translation methods
  - Theme locale translate.csv file
  - The database table core_translate
 10.1.2 Understand the pros and cons of applying translations via the translate.csv file versus the core_translate table
 10.1.3 Understand how to apply a module scope to a translation
 10.1.4 Understand the inline translation feature
10.2 Configure time zones and locale settings using the Admin panel
10.3 Configure currencies

Questions to ask yourself while studying:
- Which methods are available for translation?
- How can I customize translations in an upgrade-safe manner?
- How can I add translations for custom strings in templates?
- How can I use the translation feature to customize existing text?
- How can translations be moved from one host to another (for example, from a development host to the live server)?
- How can I translate transactional emails?
- How can I specify localized variables in transactional emails?

References:

- app/locale/*
- app/design/frontend/*/*/locale/translate.csv
- The Magento Admin interface
    - System > Configuration > Developer > Inline Translation

Sample Questions

1. You want to set up the default package and theme for an entire Magento instance without bypassing the fallback system. Which configuration area must you use?

A. System > Configuration “Design“ Tab at “Default” ConfigScope
B. System > Configuration “Design” Tab for each Website-Scope
C. System > Design for the Default StoreView
D. System > Design for all active StoreViews

2. Given the following layout XML, what will be the expected result to the My Account left navigation?

```
<customer_account translate="label">
   <reference name="left">
      <block type="customer/account_navigation"
      name="customer_account_navigation"
      template="customer/account/navigation.phtml">
         <action method="addLink"
      module="customer"><name>account</name><path>customer/account/</path><la
      bel>Account Dashboard</label></action>
      </block>
   </reference>
</customer_account>
<customer_account>
   <reference name="customer_account_navigation">
      <action method="addLink" module="wishlist">
         <name>wishlist</name>
         <path>wishlist/</path>
         <label>My Wishlist</label>
      </action>
   </reference>
</customer_account>
```

A. A single link of "My Wishlist" will be shown.
B. A link of "account" and "wishlist" will be shown.
C. A single link of "wishlist" will be shown.
D. A link of "Account Dashboard" and "My Wishlist" will be shown.
E. A single link of "account" will be shown.

3. Which one of the following statements about translations in Magento is correct?

A. An inline translation has priority over the translation.csv file.
B. The translation.csv file has priority over an inline translation.
C. The translation.csv file and inline translations have the same priority; therefore Magento translates what was last changed.
D. The translation.csv file has priority over inline translations, but inline translations have priority over translations from "app/local".

4. What is the correct XML code for configuring a page to use the single column layout?

A. `<action method="getTemplate">page/1column.phtml</action>`

B.
```
<action
method="getTemplate"><template>page/1column.phtml</template>
</action>
```

C.
```
<action
method="setTemplate"><template>page/1column.phtml</template>
</action>
```

D. `<action method="setTemplate">page/1column.phtml</action>`

Answers to Sample Questions

1- A
2- D
3- A
4- C