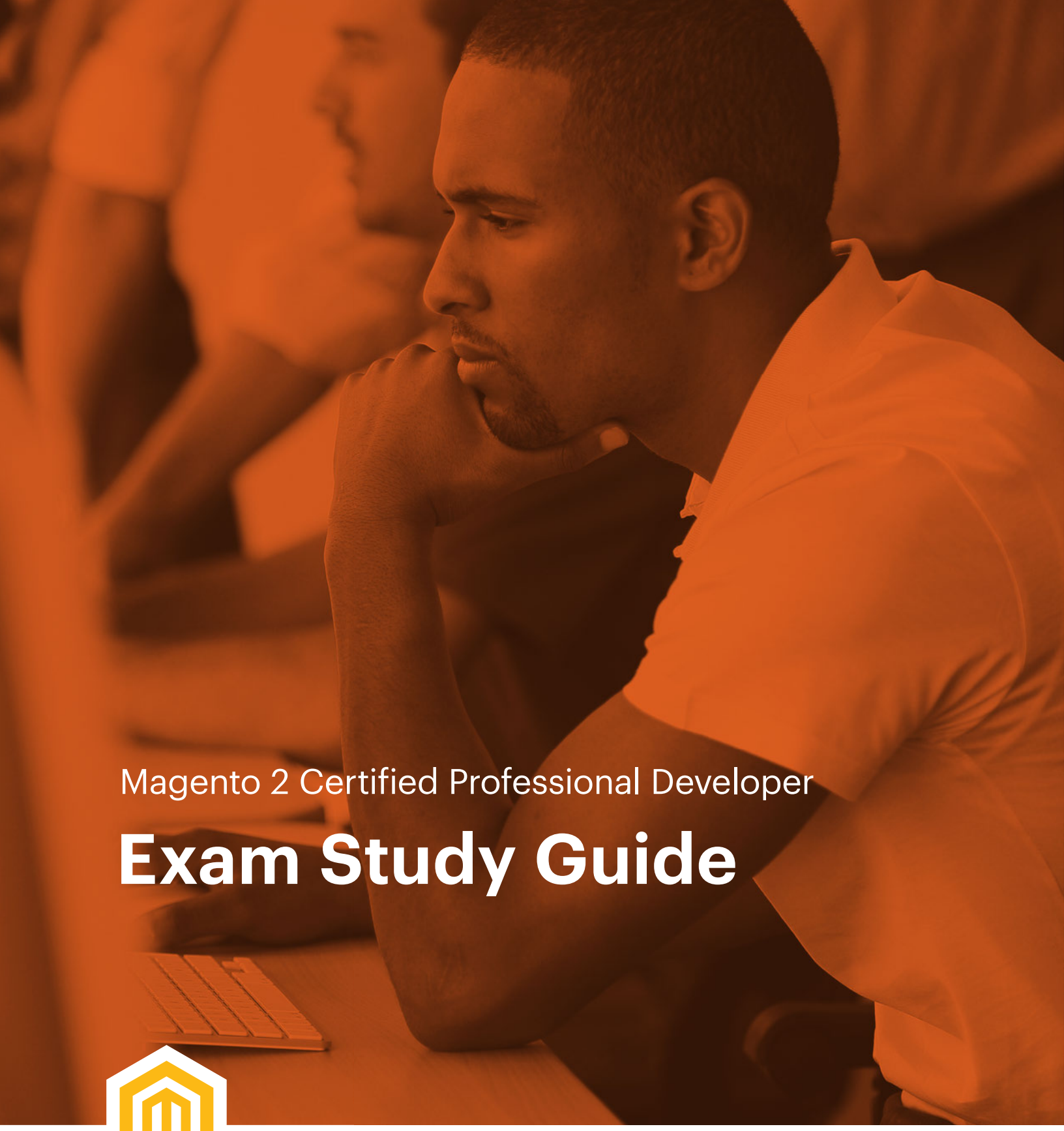


Magento 2 Certified Professional Developer

Exam Study Guide

MAGENTO 2 CERTIFIED Professional Developer

Magento® An Adobe Company | U

Magento® U

training@magento.com

u.magento.com

u.magento.com/certification

Contents

Exam Support Sheet – Online Proctored Tests

PREPARE 24 Hours in Advance

- Download the software, ensure your PC / Mac is compliant
- Minimum 1mbps upload / download speed, ping 200ms or less
- **Standard English keyboards only**
- **Questions? Visit Kryterion's knowledge base at**

  https://kryterion.force.com/support/s/topic/0TO1W000000I5h3WAC/online-proctoring?language=en_US

External Web Cam REQUIRED – NO Integrated Cameras

- **External web camera required** for online proctored exam
- **You cannot use your computer's integrated camera**
- You must be clearly visible looking into the camera

Phone Support
1-877-313-2008 (U.S.)

+001-602-659-4679 (International)

Email Support
OLPsupport@KryterionOnline.com

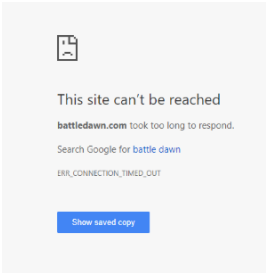
Live Chat Support
https://www.kryteriononline.com/test-taker/online-proctoring-support

SAVE Money – AVOID a Rescheduling Fee

- Can't make your scheduled time? Reschedule / cancel up to 72 hours before Testing Center exams and up to 24 hours before Online Proctored exams at **no charge**
- **Change fees after these deadlines** are $95 to reschedule / cancel a Testing Center exam and $50 to reschedule / cancel an Online Proctored exam

Scheduled Date / Time

- Check that the Time is in the correct time zone, select your location or it defaults to US times: 1600H USA / Phoenix
- Ensure that all address fields are correctly filled out in your Kryterion account including state/city and country



Browser Closed During Test?

- Simply restart the test – your time will automatically stop and all your answers will be saved
- Launch the test again and continue
- If you are having problems, change your browser – try Mozilla Firefox, Internet Explorer, or Chrome

Introduction

This exam is for a Magento 2 developer with a recommended experience level of 1.5 years in customizing different areas of the Magento platform. By passing this exam the developer earns a Magento 2 Certified Professional Developer certification.

This exam validates the skills and knowledge needed to customize Magento 2 in the areas of UI modifications; database changes; admin modifications; checkout process customizations; order management integrations and customizations; and catalog structure and functionality changes.

The test is built for version 2.3.*x* of Magento Commerce and Magento Open Source.

This exam consists primarily of scenario-based questions in a multiple-choice format.

This guide contains several sample questions at the end of the guide.

Exam topics and the approximate percentage covered in the test:

- Magento Architecture & Customization Techniques 18%
- Request Flow Processing 12%
- Customizing the Magento UI 10%
- Working with Databases in Magento 7%
- Using the Entity-Attribute-Value (EAV) Model 8%
- Developing with Adminhtml 10%
- Customizing the Catalog 12%
- Customizing the Checkout Process 13%
- Sales Operations 5%
- Customer Management 5%

Topics and Objectives

1 Magento Architecture and Customization Techniques

1.1 Describe Magento's module-based architecture

Describe module limitations. How do different modules interact with each other? What side effects can come from this interaction?

1.2 Describe Magento's directory structure

Determine how to locate different types of files in Magento. Where are the files containing JavaScript, HTML, and PHP located? How do you find the files responsible for certain functionality?

1.3 Utilize configuration XML and variables scope

Determine how to use configuration files in Magento. Which configuration files correspond to different features and functionality?

1.4 Demonstrate how to use dependency injection

Describe Magento's dependency injection approach and architecture. How are objects realized in Magento? Why is it important to have a centralized process creating object instances?

Identify how to use DI configuration files for customizing Magento. How can you override a native class, inject your class into another object, and use other techniques available in `di.xml` (such as `virtualTypes`)?

1.5 Demonstrate ability to use plugins

Demonstrate how to design complex solutions using the plugin's life cycle. How do multiple plugins interact, and how can their execution order be controlled? How do you debug a plugin if it doesn't work?

Identify strengths and weaknesses of plugins. What are the limitations of using plugins for customization? In which cases should plugins be avoided?

1.6 Configure event observers and scheduled jobs

Demonstrate how to configure observers. How do you make your observer only be active on the frontend or backend?

Demonstrate how to configure a scheduled job. Which parameters are used in configuration, and how can configuration interact with server configuration?

Identify the function and proper use of automatically available events, for example *_load_after, etc.

1.7 Utilize the CLI

Describe the usage of `bin/magento` commands in the development cycle. Which commands are available? How are commands used in the development cycle?

Demonstrate an ability to create a deployment process. How does the application behave in different deployment modes, and how do these behaviors impact the deployment approach for PHP code, frontend assets, etc.?

1.8 Demonstrate the ability to manage the cache

Describe cache types and the tools used to manage caches. How do you add dynamic content to pages served from the full page cache?

Describe how to operate with cache clearing. How would you clean the cache? In which case would you refresh cache/flash cache storage?

Describe how to clear the cache programmatically. What mechanisms are available for clearing all or part of the cache?

2 Request Flow Processing

2.1 Utilize modes and application initialization

Identify the steps for application initialization. How would you design a customization that should act on every request and capture output data regardless of the controller?

Describe how to use Magento modes. What are pros and cons of using developer mode/production mode? When do you use default mode? How do you enable/disable maintenance mode?

Describe front controller responsibilities. In which situations will the front controller be involved in execution, and how can it be used in the scope of customizations?

2.2 Demonstrate ability to process URLs in Magento

Describe how Magento processes a given URL. How do you identify which module and controller corresponds to a given URL? What is necessary to create a custom URL structure?

Describe the URL rewrite process and its role in creating user-friendly URLs. How are user-friendly URLs established, and how are they customized?

Describe how action controllers and results function. How do controllers interact with another? How are different response types generated?

2.3 Demonstrate ability to customize request routing

Describe request routing and flow in Magento. When is it necessary to create a new router or to customize existing routers? How do you handle custom 404 pages?

2.4 Determine the layout initialization process

Determine how layout is compiled. How would you debug your `layout.xml` files and verify that the right layout instructions are used?

Determine how HTML output is rendered. How does Magento flush output, and what mechanisms exist to access and customize output?

Determine module layout XML schema. How do you add new elements to the pages introduced by a given module?

Demonstrate the ability to use layout fallback for customizations and debugging. How do you identify which exact `layout.xml` file is processed in a given scope? How does Magento treat layout XML files with the same names in different modules?

Identify the differences between admin and frontend scopes. What differences exist for layout initialization for the admin scope?

2.5 Determine the structure of block templates

Identify and understand root templates, `empty.xml`, and `page_layout`. How are page structures defined, including number of columns, which basic containers are present, etc.?

Describe the role of blocks and templates in the request flow. In which situations would you create a new block or a new template?

3 Customizing the Magento UI

3.1 Demonstrate ability to utilize themes and the template structure

Demonstrate the ability to customize the Magento UI using themes. When would you create a new theme? How do you define theme hierarchy for your project?

Demonstrate the ability to customize/debug templates using the template fallback process. How do you identify which exact theme file is used in different situations? How can you override native files?

3.2 Determine how to use blocks

Demonstrate an understanding of block architecture and its use in development. Which objects are accessible from the block? What is the typical block's role?

Identify the stages in the lifecycle of a block. In what cases would you put your code in the `_prepareLayout()`, `_beforeToHtml()`, and `_toHtml()` methods? How would you use events fired in the abstract block?

Describe how blocks are rendered and cached.

Identify the uses of different types of blocks. When would you use non-template block types? In what situation should you use a template block or other block types?

3.3 Demonstrate ability to use layout and XML schema

Describe the elements of the Magento layout XML schema, including the major XML directives. How do you use layout XML directives in your customizations?

Describe how to create a new layout XML file.

Describe how to pass variables from layout to block.

3.4 Utilize JavaScript in Magento

Describe different types and uses of JavaScript modules. Which JavaScript modules are suited for which tasks?

Describe UI components. In which situation would you use UiComponent versus a regular JavaScript module?

Describe the use of `requirejs-config.js`, `x-magento-init`, and `data-mage-init`.

4 Working with Databases in Magento

4.1 Demonstrate ability to use data-related classes

Describe repositories and data API classes. How do you obtain an object or set of objects from the database using a repository? How do you configure and create a SearchCriteria instance using the builder? How do you use Data/Api classes?

Describe how to create and register new entities. How do you add a new table to the database?

Describe the entity load and save process.

Describe how to extend existing entities. What mechanisms are available to extend existing classes, for example by adding a new attribute, a new field in the database, or a new related entity?

Describe how to filter, sort, and specify the selected values for collections and repositories. How do you select a subset of records from the database?

Describe the database abstraction layer for Magento. What type of exceptions does the database layer throw? What additional functionality does Magento provide over `Zend_Adapter`?

4.2 Demonstrate an ability to use declarative schema

Demonstrate use of schema. How to manipulate columns and keys using declarative schema? What is the purpose of whitelisting? How to use Data and Schema patches? How to manage dependencies between patch files?

5 Using the Entity-Attribute-Value (EAV) Model

5.1 Demonstrate ability to use EAV model concepts

Describe the EAV hierarchy structure. What happens when a new attribute is added to the system? What is the role of attribute sets and attribute groups? How are attributes presented in the admin?

Describe how EAV data storage works in Magento. Which additional options do you have when saving EAV entities? How do you create customizations based on changes to attribute values?

Describe the key differences between EAV and flat table collections. In which situations would you use EAV for a new entity? What are the pros and cons of EAV architecture?

5.2 Demonstrate ability to use EAV entity load and save

Describe the EAV load and save process and differences from the flat table load and save process. What happens when an EAV entity has too many attributes? How does the number of websites/stores affect the EAV load/save process? How would you customize the load and save process for an EAV entity in the situations described here?

5.3 Demonstrate ability to manage attributes

Describe EAV attributes, including the frontend/source/backend structure. How would you add dropdown/multiselect attributes? What other possibilities do you have when adding an attribute (to a product, for example)?

Describe how to implement the interface for attribute frontend models. What is the purpose of this interface? How can you render your attribute value on the frontend?

Identify the purpose and describe how to implement the interface for attribute source models. For a given dropdown/multiselect attribute, how can you specify and manipulate its list of options?

Identify the purpose and describe how to implement the interface for attribute backend models. How (and why) would you create a backend model for an attribute?

Describe how to create and customize attributes. How would you add a new attribute to the product, category, or customer entities? What is the difference between adding a new attribute and modifying an existing one?

6 Developing with Adminhtml

6.1 Describe common structure/architecture

Describe the difference between Adminhtml and frontend. What additional tools and requirements exist in the admin?

6.2 Define form and grid widgets

Define form structure, form templates, grids, grid containers, and elements. What steps are needed to display a grid or form?

Describe the grid and form workflow. How is data provided to the grid or form? How can this be process be customized or extended?

Describe how to create a simple form and grid for a custom entity. Given a specific entity with different types of fields (text, dropdown, image, file, date, and so on) how would you create a grid and a form?

6.3 Define system configuration XML and configuration scope

Define basic terms and elements of system configuration XML, including scopes. How would you add a new system configuration option? What is the difference in this process for different option types (secret, file)?

Describe system configuration data retrieval. How do you access system configuration options programmatically?

6.4 Utilize ACL to set menu items and permissions

Describe how to set up a menu item and permissions. How would you add a new menu item in a given tab? How would you add a new tab in the Admin menu? How do menu items relate to ACL permissions?

Describe how to check for permissions in the permissions management tree structures. How would you add a new user with a given set of permissions? How can you do that programmatically?

7 Customizing the Catalog

7.1 Demonstrate ability to use products and product types

Identify/describe standard product types (simple, configurable, bundled, etc.). How would you obtain a product of a specific type? What tools (in general) does a product type model provide? What additional functionality is available for each of the different product types?

7.2 Describe price functionality

Identify the basic concepts of price generation in Magento. How would you identify what is composing the final price of the product? How can you customize the price calculation process?

Describe how price is rendered in Magento. How would you render price in a given place on the page, and how would you modify how the price is rendered?

7.3 Demonstrate ability to use and customize categories

Describe category properties and features. How do you create and manage categories?

Describe the category hierarchy tree structure implementation (the internal structure inside the database). What is the meaning of `parent_id` 0? How are paths constructed? Which attribute values are required to display a new category in the store? What kind of strategies can you suggest for organizing products into categories?

7.4 Determine and manage catalog rules

Identify how to implement catalog price rules. When would you use catalog price rules? How do they impact performance? How would you debug problems with catalog price rules?

8 Customizing the Checkout Process

8.1 Demonstrate ability to use quote, quote item, address, and shopping cart rules in checkout

Describe how to modify these models and effectively use them in customizations.

Describe how to customize the process of adding a product to the cart. Which different scenarios should you take into account?

8.2 Demonstrate ability to use totals models

Describe how to modify the price calculation process in the shopping cart. How can you add a custom totals model or modify existing totals models?

8.3 Demonstrate ability to customize the shopping cart

Describe how to implement shopping cart rules. What is the difference between sales rules and catalog rules? How do sales rules affect performance? What are the limitations of the native sales rules engine?

Describe add-to-cart logic in different scenarios. What is the difference in adding a product to the cart from the product page, from the wishlist, by clicking Reorder, and during quotes merge?

Describe the difference in behavior of different product types in the shopping cart. How are configurable and bundle products rendered? How can you create a custom shopping cart renderer?

Describe the available shopping cart operations. Which operations are available to the customer on the cart page? How can you customize cart edit functionality? How would you create an extension that deletes one item if another was deleted? How do you add a field to the shipping address?

8.4 Demonstrate ability to customize shipping and payment methods

Describe shipping methods architecture. How can you create a new shipping method? What is the relationship between carriers and rates?

Describe how to troubleshoot shipping methods and rate results. Where do shipping rates come from? How can you debug the wrong shipping rate being returned?

Describe how to troubleshoot payment methods. What types of payment methods exist? What are the different payment flows?

9 Sales Operations

9.1 Demonstrate ability to customize sales operations

Describe how to modify order processing and integrate it with a third-party ERP system.

Describe how to modify order processing flow. How would you add new states and statuses for an order? How do you change the behavior of existing states and statuses?

Described how to customize invoices. How would you customize invoice generation, capturing, and management?

Describe refund functionality in Magento. Which refund types are available, and how are they used?

10 Customer Management

10.1 Demonstrate ability to customize My Account

Describe how to customize the "My Account" section. How do you add a menu item? How would you customize the "Order History" page?

10.2 Demonstrate ability to customize customer functionality

Describe how to add or modify customer attributes.

Describe how to extend the customer entity. How would you extend the customer entity using the extension attributes mechanism?

Describe how to customize the customer address. How would you add another field into the customer address?

Describe customer groups and their role in different business processes. What is the role of customer groups? What functionality do they affect?

Describe Magento functionality related to VAT. How do you customize VAT functionality?

Magento 2 Certified Professional Developer Exam Example Questions

See the Answer Key following the questions for answers and references.

Question 1

A merchant asks you to create a module that is able to process URLs with a custom structure that can contain any combination of a product type code, a partial name, and a 4-digit year in any order.

The request path will look like this: `/product/:type-code/:name-part/:year`

Which layer in the Magento request processing flow is suited for this kind of customization?

A. Front controller

B. Router

C. Action controller

D. HTTP Response

Question 2

While integrating a merchant's product information management system with Magento, you create a module `MyCompany_MerchantPim` that adds a catalog product EAV attribute `pim_entity_id` programmatically.

In which type of setup script do you create the EAV attribute?

A) Setup/InstallSchema.php

B) Setup/UpgradeSchema.php

C) Setup/InstallEntity.php

D) Setup/UpgradeData.php

Question 3

You are facing a bug, which is supposedly caused by the customization of `\Magento\Catalog\Api\ProductRepositoryInterface::save()`.

To resolve the issue, you decide to find all logic which customizes this method.

Which two places do you search for customization declarations? (Choose 2)

A. `*/di.xml`

B. `*/config.xml`

C. `*/events.xml`

D. `*/plugins.xml`

Question 4

You are implementing a customization of the sales management within a module `MyCompany_MySalesProcess`. You have created several event observers to add the custom functionality. Each observer is a separate class, but they require some common functionality.

How do you implement the common functionality in the event observers, keeping maintainability and testability in mind?

A. You create a trait with the common methods and use the trait in the observer classes.

B. You create an abstract class AbstractObserver with the common methods and extend the observer classes from it.

C. You create a regular class implementing the common functionality as public static methods and call those from the observers.

D. You create a regular class implementing the common functionality as public methods and use constructor injection to make them available to the observers.

Question 5

A custom module is performing an optimized custom query for quote items. The class applies the query customizations on the select object of a quote item collection instance.

```
public function __construct(
    \Magento\Quote\Model\ResourceModel\Quote\Item\Collection $collection
) {
    $this->collection = $collection;
}


public function fetchData()
```

```
{
    $select = $this->collection->getSelect();
    ... code modifying $select...
   return $this->collection->getData();
}
```

You are tasked to resolve an issue where the query sometimes does not deliver the expected results. You have debugged the problem and found another class is also using a quote item collection and is loading the collection before the custom module.

How do you resolve the issue, keeping maintainability in mind?

A. You change the argument type to \Magento\Quote\Model\ResourceModel\Quote\Item\CollectionFactory and instantiate the collection using $collectionFactory->create();

B. You remove the constructor argument and use ObjectManager::*getInstance*()->create(\Magento\Quote\Model\ResourceModel\Quote\Item\Collection::class) to instantiate the collection instead.

C. You inject \Magento\Framework\DB\Select instead of the collection and perform the desired query independently of the collection.

D. You inject \Magento\Quote\Api\CartItemRepositoryInterface because low level query customizations are not allowed.

Question 6

In a custom module you implement the interface `\Magento\Framework\App\Config\DataInterface`.

```
/**
 * Configuration data storage
 *
 * @api
 */
interface DataInterface
{
    public function getValue($path);
    public function setValue($path, $value);
}
```

What version constraint for `magento/framework` do you add to your module's `composer.json` file?

A. major

B. minor

C. patch

D. stable

Answer Key

Answer Key

Question 1

Answer: B

Reference:
http://devdocs.magento.com/guides/v2.2/extension-dev-guide/routing.html

Question 2

Answer: D

References:
`\Magento\Eav\Setup\EavSetup::addAttribute()` fails if the reference `entity_type_id` does not exist.

https://github.com/magento/magento2/blob/2.2-develop/app/code/Magento/Eav/Setup/InstallSchema.php#L739-L744

https://github.com/magento/magento2/blob/2.2-develop/app/code/Magento/Catalog/Setup/InstallData.php#L68

Question 3

Answers: A and C

References:

http://devdocs.magento.com/guides/v2.2/extension-dev-guide/plugins.html

http://devdocs.magento.com/guides/v2.2/extension-dev-guide/events-and-observers.html

Question 4

Answer: D

Reference:

Magento Technical Coding Guidelines, Section 2.6 “Inheritance SHOULD NOT be used. Composition SHOULD be used for code reuse.“
http://devdocs.magento.com/guides/v2.2/coding-standards/technical-guidelines.html

Question 5

Answer: A

References:

Magento DevDocs developer guide on dependency injection

http://devdocs.magento.com/guides/v2.2/extension-dev-guide/depend-inj.html

Magento DevDocs developer guide on factories

http://devdocs.magento.com/guides/v2.2/extension-dev-guide/factories.html

Question 6

Answer: B

Reference:

Magento DevDocs developer guide on module version dependencies

http://devdocs.magento.com/guides/v2.2/extension-dev-guide/versioning/dependencies.html