Adobe

AD0-E717
Adobe Commerce Developer Professional

Latest Version: 7.0

Question: 1

How should a developer display a custom attribute on the category edit page in the admin panel when a new module Vendor.Category is created?

A. Create view/adminhtml/layout/catalog_category_edit xml in the module, and then define a block that would display the field for the attribute.
B. The field for the attribute will appear automatically.
C. Create view/adminhtml/ui_component/category_form.xml file in the module, and then define the field for the attribute.

Answer: C

Explanation:
To display a custom attribute on the category edit page in the Magento admin panel, a developer should use the UI component approach. This involves creating a category_form.xml file within the view/adminhtml/ui_component directory of the module. This XML file defines the form fields (including any custom attributes) that should appear on the category edit page. The UI component system in Magento provides a flexible way to manage form elements and their interactions on the admin side, ensuring a consistent user interface and experience .

Question: 2

How should a grid or form be included in an admin page layout using the Ul Component?

A. <referenceContainername='content"> q <uiComponentname="example_listing.xml7>
</referenceContainer>
B. <referenceContainername='contenf> q <uiComponent name="example_listing7>
</referenceContainer>
C. <referenceContainername='content">
<uiComponentname="Vendor_Module::ul_component/example_listing.xml7> </referenceContainer>

Answer: B

Explanation:
To include a grid or form in an admin page layout using the UI Component, the correct approach is to use the <uiComponent name="example_listing"/> within a <referenceContainer name='content'> block of the layout XML file. This method directly references the UI component's configuration file (e.g., example_listing.xml) which defines the structure and functionality of the UI component, such as grids or forms. This configuration file is located under the view/adminhtml/ui_component directory of the corresponding module .

Question: 3

Which action, if any, should be taken to forbid Adobe Commerce Admin from performing specific actions?

A. Create a new user role with custom-defined resources, and assign it to the admin user
B. This action cannot be taken since all admin users must have full access.
C. Enable custom roles in the store configuration, and assign admin user ID(s).

Answer: A

Explanation:
To forbid Adobe Commerce Admin from performing specific actions, a developer should create a new user role with custom-defined resources, and assign it to the admin user. This can be done by going to System > Permissions > Roles and creating a new role. In the Resources section, the developer can select the specific resources that they want to restrict the admin user from accessing.
To restrict specific actions within the Adobe Commerce Admin, the recommended approach is to utilize Magento's Access Control List (ACL). This can be done by creating a new user role with custom-defined resources and assigning this role to the admin user. This approach allows for granular control over what actions an admin user can perform by specifying allowed resources within the role. Magento's ACL system is designed to manage permissions effectively, ensuring that users only have access to the necessary functionalities required for their role .

Question: 4

Which file should a developer use to set the default value when creating configuration fields for admin?

A. etc/adminhtml/config.xml
B. etc/config xml
C. etc/adminhtml/system.xml

Answer: A

Explanation:
When creating configuration fields for the admin panel and setting their default values, a developer should use the etc/config.xml file within their module. This file is used to declare default values for the module's configuration options. When Magento is installed or the module is enabled, these values are automatically loaded into the database, under the core_config_data table, setting the initial configuration for the module. This approach ensures that the module has sensible defaults and operates correctly upon installation

Question: 5

How can a developer prioritize a plugin's execution, if possible?

A. The developer can use sortOrder property by specifying a lower value than the target plugin.
B. The developer can use sortOrder property by specifying a higher value than the target plugin.
C. This cannot be achieved as the plugins are always executed by their module's load order in app/etc/config.php file.

Answer: A

Explanation:
A developer can prioritize the execution of a plugin by using the sortOrder property within the plugin's declaration in the di.xml file. Specifying a lower value for the sortOrder property gives the plugin higher priority, meaning it will be executed before other plugins with a higher sortOrder value. This allows developers to control the order of plugin execution, which can be critical for ensuring the desired outcome when multiple plugins are affecting the same method .