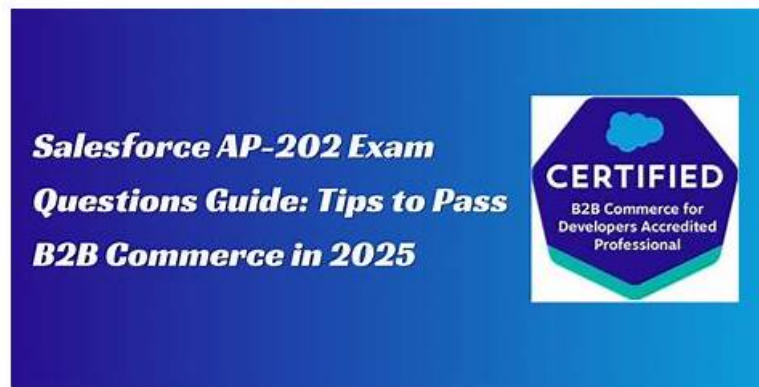


Salesforce certification AP-202 exam targeted exercises



BONUS!!! Download part of PassReview AP-202 dumps for free: https://drive.google.com/open?id=1Z_NqaMtKXspGT6TUkCeDFW1C2hMDbTxk

It is universally accepted that the competition in the labor market has become more and more competitive in the past years. In order to gain some competitive advantages, a growing number of people have tried their best to pass the AP-202 exam. Because a lot of people hope to get the certification by the related exam, now many leaders of companies prefer to the candidates who have the AP-202 certification. In their opinions, the certification is a best reflection of the candidates' work ability, so more and more leaders of companies start to pay more attention to the AP-202 certification of these candidates. If you also want to come out ahead, it is necessary for you to prepare for the exam and get the related certification.

PassReview is a leading platform in this area by offering the most accurate AP-202 exam questions to help our customers to pass the exam. And we are grimly determined and confident in helping you. With professional experts and brilliant teamwork, our AP-202 practice materials have helped exam candidates succeed since the beginning. To make our AP-202 simulating exam more precise, we do not mind splurge heavy money and effort to invite the most professional teams into our group.

>> **Dump AP-202 Collection** <<

AP-202 Latest Test Report & Reliable AP-202 Test Experience

What happens when you are happiest? It must be the original question! The hit rate of AP-202 study materials has been very high for several reasons. Our company has collected the most comprehensive data and hired the most professional experts to organize. They are the most authoritative in this career. At the same time, we are very concerned about social information and will often update the content of our AP-202 Exam Questions.

Salesforce B2B Commerce for Developers Accredited Professional Sample Questions (Q89-Q94):

NEW QUESTION # 89

Which two methods from the platformResourceLoader module are relevant for including third party JavaScript and CSS in a Lightning web component?

- A. loadClientScript
- B. loadCss
- C. loadScript
- D. loadStyle

Answer: C,D

Explanation:

Two methods from the platformResourceLoader module that are relevant for including third party JavaScript and CSS in a Lightning web component are loadScript and loadStyle. The platformResourceLoader module is a module that provides methods for loading JavaScript or CSS files from static resources or external URLs into a Lightning web component. The loadScript method is used to load a JavaScript file and execute it in the component. The loadStyle method is used to load a CSS file and apply it to the component. The loadClientScript method does not exist or is not relevant for including third party JavaScript in a Lightning web

component. The loadCss method does not exist or is not relevant for including third party CSS in a Lightning web component. Salesforce Lightning Web Components Developer Guide: Load Scripts and Style Sheets, [Lightning Web Components Developer Guide: platformResourceLoader Module]

NEW QUESTION # 90

Which three statements are true about Global API versioning? (3 answers)

- A. The API version is scoped at the Class API level and NOT at the method level.
- B. Minimum API_VERSION is 1 and the Maximum API version follows the releases. E.g. The maximum was 4 as of Salesforce B2B Commerce Release-4.5, 5 as of Salesforce B2B Commerce Release 4.6, etc.
- C. Calling in with an API version set to lower than 1 will result in an exceptional case where the exception class `ccrz.BelowMinAPIVersionException` will be returned to callers.
- D. There is no need to pass API_VERSION to the Global APIs, and based on the Salesforce B2B Commerce Managed Package version, Global APIs are able to figure out what version of the API to use.
- E. Calling in with an API version set to more than current maximum will result in exception case where the exception class `ccrz.ExceedsMaxAPIVersionException` will be returned to callers.

Answer: B,C,E

Explanation:

Three statements that are true about Global API versioning are:

Calling in with an API version set to lower than 1 will result in an exceptional case where the exception class `ccrz.BelowMinAPIVersionException` will be returned to callers. This exception indicates that the API version is not supported by the framework and the caller should use a higher API version.

The API version is scoped at the Class API level and NOT at the method level. This means that all the methods in a class will use the same API version that is specified by the caller. For example, if the caller passes an API version of 4 to `ccrz.ccServiceProduct.getProducts()`, then all the other methods in `ccrz.ccServiceProduct` will also use API version 4.

Calling in with an API version set to more than current maximum will result in exception case where the exception class `ccrz.ExceedsMaxAPIVersionException` will be returned to callers. This exception indicates that the API version is not supported by the framework and the caller should use a lower API version. Salesforce B2B Commerce and D2C Commerce Developer Guide, API Versioning

NEW QUESTION # 91

A developer needs to implement a custom Lightning web component (LWC) for the storefront.

The LWC contains language-specific text values.

How should the developer translate the text values?

- A. Create custom labels for the text values and import them in the LW
- B. Use a CustomLabel.xml file in the LWC to add the text values there.
- C. Import static resources for the text values and add them into the LWC.
- D. Create a custom Metadata object for the text values and query it in the LWC.

Answer: A

Explanation:

Custom labels are text values that can be translated into any language that Salesforce supports. They are useful for displaying language-specific text in Lightning web components. To use custom labels in a LWC, the developer needs to create them in the Setup menu and assign them to a language and a value. Then, the developer can import them in the LWC using the `@salesforce/labelscoped` module. For example, if the developer has a custom label named `welcomeHeader`, they can import it as follows:

```
import welcomeHeader from '@salesforce/label/c.welcomeHeader';
```

Then, they can use it in the HTML template or the JavaScript file of the LWC. For example, in the HTML template, they can use it as follows:

HTMLAI-generated code. Review and use carefully. More info on FAQ.

```
<template>  
<h1>{welcomeHeader}</h1>  
</template>
```

The custom label will automatically display the translated value based on the user's language preference. The developer can also use the `lightning-formatted-text` component to format the custom label value with HTML tags.

The other options are not correct because:

A) Importing static resources for the text values is not a recommended way to translate text values in a LWC. Static resources are files that are stored in Salesforce and can be referenced by applications. They are not designed for storing language-specific text values and they do not support automatic translation based on the user's language preference.

B) Using a CustomLabel.xml file in the LWC to add the text values there is not a valid option. Custom labels are not stored in.xml files, but in the Setup menu. They cannot be added directly to the LWC, but they need to be imported using the @salesforce/labels module.

D) Creating a custom Metadata object for the text values and querying it in the LWC is not a feasible option. Custom Metadata objects are records that store configuration data that can be deployed and packaged. They are not intended for storing language-specific text values and they do not support automatic translation based on the user's language preference. Querying them in the LWC would also require an Apex class and a wire service, which would add unnecessary complexity to the solution.

Use Custom Labels in Lightning Web Components

Custom Labels

Internationalizing Your Lightning Web Component (LWC)

NEW QUESTION # 92

Which two steps are necessary to enable Salesforce B2B Commerce logging in the managed package?

- A. Turn On the Checkbox "Cloudcraze Logging" in CC Admin.
- B. Ensure the value saved in the Logging token field is appended to the ccLog query parameter.
- C. Set a cookie with the Id of the user accessing the storefront in CC Admin
- D. Ensure you save a value in the Logging Token input field in the Global Settings section of CC Admin.

Answer: B,D

Explanation:

To enable Salesforce B2B Commerce logging in the managed package, you need to do two steps. First, you need to save a value in the Logging Token input field in the Global Settings section of CC Admin. This value can be any string that you choose, such as "debug". Second, you need to ensure that the value saved in the Logging token field is appended to the ccLog query parameter in the URL of the storefront page that you want to debug. For example, if your logging token is "debug", then your URL should look like this: <https://my-storefront.com/?ccLog=debug>. This will enable logging for that page only. You do not need to turn on the checkbox "Cloudcraze Logging" in CC Admin, as this is an old setting that is no longer used. You also do not need to set a cookie with the Id of the user accessing the storefront in CC Admin, as this is not required for logging. Salesforce [B2B Commerce Developer Guide: Logging]

NEW QUESTION # 93

What two kinds of queries do the methods in Salesforce B2B Commerce services perform by default? (2 answers)

- A. SOSL
- B. SOQL
- C. SQL
- D. Schema-less queries

Answer: B,D

Explanation:

Two kinds of queries that the methods in Salesforce B2B Commerce services perform by default are SOQL and schema-less queries. SOQL is the query language that is used to retrieve data from Salesforce objects and fields. Schema-less queries are queries that do not specify the object or field names explicitly, but use placeholders instead. For example, `ccrz.ccServiceDao.getQuery('SELECT Id FROM Account WHERE Name = :name')` is a schema-less query that uses `:name` as a placeholder for the field name. The framework will transform this query to use the actual field name based on the query transformation rules. Salesforce B2B Commerce and D2C Commerce Developer Guide, Query Transformation

NEW QUESTION # 94

.....

