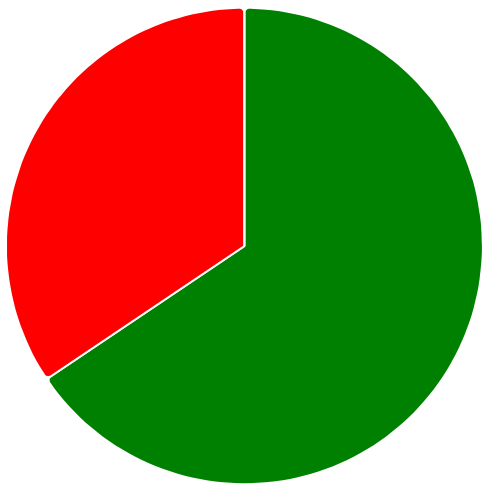
Return To Review



Attempt: 61 PASS!

66% correct (40/61)

Knowledge Area

Q.1. A company recently deployed a Visualforce page with a custom controller that has a data grid of information about Opportunities in the org. Users report that they receive a "Maximum view state size limit" error message under certain conditions.
According to Visualforce best practice, which three actions should the developer take to reduce the view state?
Choose 3 answers.

Answered Right

Ask Instructor

1. Use filters and pagination to reduce the amount of data. (Correct Answer)(Selected Answer)
2. Refine any SOQL queries to return only data relevant to the page. (Correct Answer)(Selected Answer)
3. Use the private keyword in the controller for variables.
4. Use the final keyword in the controller for variables that will not change.
5. Use the transient keyword in the Apex controller for variables that do not maintain state. (Correct Answer)(Selected Answer)

Create Notes

This question is related to the performance of the visualforce page in Salesforce. The scenario given in this question is:

- A developer created and tested a Visualforce page in their developer sandbox, but now users report that they receive a "Maximum view state size limit" error message under certain conditions.

Visualforce is designed to provide developers with the ability to match the functionality, behavior, and performance of standard Salesforce pages.
There are different Visualforce performance issues and one of them is View State Size.

- The view state size of your Visualforce pages must be under 170KB. By reducing your view state size, your pages can load quicker and stall less often.
- You can monitor view state performance through the View State tab in the development mode footer.
- As per documentation given by Salesforce, there are different ways to reduce view state size in Salesforce as below:
  - Use **filters and pagination** to reduce data requiring state.
  - **Use the transient keyword** in your Apex controllers for variables that aren't essential for maintaining state and aren't necessary during page refreshes.
  - If you notice that a large percentage of your view state comes from objects used in controllers or controller extensions, consider **refining your SOQL calls** to return only data that are relevant to the Visualforce page.

With this explanation, the correct answers are:
"Use filters and pagination to reduce the amount of data."

and
"Refine any SOQL queries to return only data relevant to the page."

and
"Use the transient keyword in the Apex controller for variables that do not maintain state."

Reference:
https://developer.salesforce.com/docs/atlas.en-us.salesforce_visualforce_best_practices.meta/salesforce_visualforce_best_practices/pages_best_practices_perf_code_view_st

Create Notes

Q.2. Refer to the markup below:

```
<template>
    <!-- ... other code ... -->
    <lightning-record-form
        record-id={recordId}
        object-api-name="Account"
        layout-type="Full">
    </lightning-record-form>
</template>
```

A Lightning web component displays the Account name and two custom fields out of 275 that exist on the object. The developer receives complaints that the component performs slowly.
What can the developer do to improve the performance?

Answered Right

Ask Instructor

1. Add cache="true" to the component.
2. Replace layout-type="Full" with layout-type="Partial".
3. Replace layout-type="Full" with fields={fields}. (Correct Answer) (Selected Answer)
4. Add density="compact" to the component.

This is a scenario-based question and is related to Lightning Web Component in Salesforce. The scenario given in this question is:

- There is a lightning web component which should display 3 fields like Account Name, Account Industry, Account Rating.
- There are total 275 fields on Account object.

The ask is:

- The developer receives complaints that the component performs slowly.
- What can the developer do to improve the performance?

Let's learn about lightning-record-form

- Use the lightning-record-form component to quickly create forms to add, view, or update a record.
- If the object has so many fields (say 100+) and if we are not interested in manually displaying them one after the other then we will be going ahead with the tag lightning-record-form.

As per the scenario given in this question, we are not interested in displaying all the fields of the page layout and only interested in displaying only 3 fields from the list of all fields.
In that case, we can use a new attribute called fields. To this attribute, we need to pass an array of fields that we are interested in fetching and displaying. This will also improve the performance as we are displaying only limited fields.
Let's re-create the same example given in the question by replacing layout-type attribute with fields attribute.

Screenshot - .html file

form.html

```
force-app > main > default > lwc > form > form.html > ...
1   <template>
2
3   <!-- using lightning-record-form-->
4   <lightning-card  title="Account Information">
5       <lightning-record-form
6           record-id={recordId}
7           object-api-name="Account"
8           fields={fields}
9           mode="view">
10      </lightning-record-form>
11  </lightning-card>
12  </template>
13
```

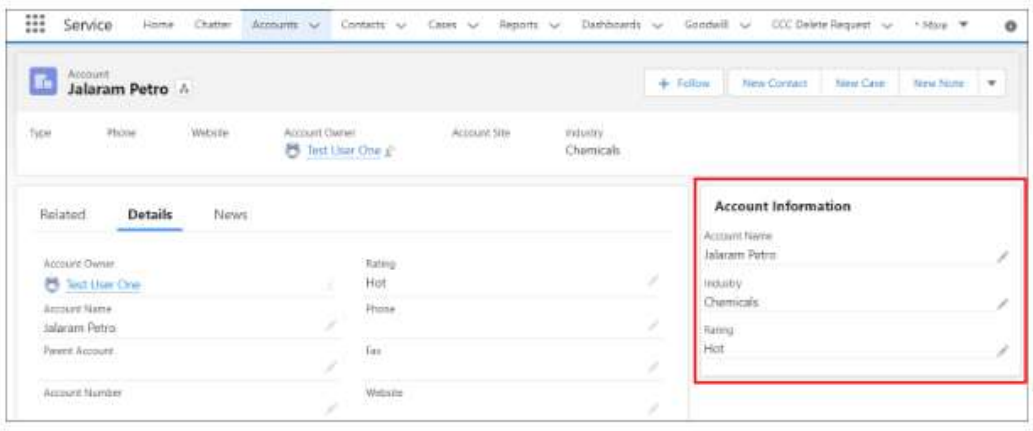
Using fields attribute

Screenshot - .js file

form.js

```
force-app > main > default > lwc > form > form.js > ...
1   import { LightningElement, api } from "lwc";
2
3   //import the reference to fields
4   import NAME_FIELD from "@salesforce/schema/Account.Name";
5   import INDUSTRY_FIELD from "@salesforce/schema/Account.Industry";
6   import RATING_FIELD from "@salesforce/schema/Account.Rating";
7
8   export default class Form extends LightningElement {
9     @api recordId;
10
11    //create an array type property
12    fields = [NAME_FIELD, INDUSTRY_FIELD, RATING_FIELD];
13  }
```

Screenshot – See Account Information



With this explanation,

Create Notes

the correct answer is:
“Replace layout-type="Full” with fields={fields}” because the requirement is to display 3 fields and to improve performance which can be achieved by replacing layout-type attribute by fields attribute.

Reference:
https://developer.salesforce.com/docs/component-library/bundle/lightning-record-form/documentation

---

Q.3. Refer to the code snippet below:

```
import fetchOpps from '@salesforce/apex/OpportunitySearch.fetchOpportunities';
@wire(fetchOpps)
opportunities;
```

When a Lightning web component is rendered, a list of opportunities that match certain criteria should be retrieved from the database and displayed to the end-user.
Which three considerations must the developer implement to make the method available within the Lightning web component?
Choose 3 answers.

Answered Right

Ask Instructor

1. The fetchOpps method cannot mutate the result set retrieved from the database. (Correct Answer)(Selected Answer)
2. The fetchOpps method must be annotated with the @AuraEnabled annotation. (Correct Answer)(Selected Answer)
3. The fecchOpps method must specify the (continuation=true) attribute.
4. The fecchOpps method must specify the (cacheable=true) attribute. (Correct Answer)(Selected Answer)
5. The fetchOpps method must be annotated with the @InvocableMethod annotation.

This question is related to the concept of calling apex methods from lightning web components. The scenario given in this question is:

- There is an apex class named **“OpportunitySearch”** and a method **“fetchOpportunities()”**
- The sample code given in this question uses wire service to read salesforce data. It only reads the data.
- **“fetchOpps”** is a symbol that identifies the Apex method in the lightning web component.

The ask is:

- Which three considerations must the developer implement to make the **“fetchOpps”** method available within the Lightning web component

As per the Salesforce documentation:

- The **AuraEnabled** annotation enables Lightning components to access Apex methods and properties.
- To use @wire to call an Apex method, annotate the Apex method with **@AuraEnabled(cacheable=true)**
- To improve runtime performance, set @AuraEnabled(cacheable=true) to cache the method results on the client. To set cacheable=true, a method must only get data. It can’t mutate data.

With this explanation, the correct answers to this question are:
“The fetchOpps method must be annotated with the @AuraEnabled annotation.”

and
“The fecchOpps method must specify the (cacheable=true) attribute.”

and
“The fetchOpps method cannot mutate the result set retrieved from the database.”

For Reference & more detail:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/lwc.apex_wire_method
https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/controllers_server_apex_auraenabled_annotation.htm

---

Create Notes

Q.4. A developer is building a Lightning web component that retrieves data from Salesforce and assigns it to the record property.

```
import { LightningElement, api, wire } from 'lwc';
import { getRecord } from 'lightning/uiRecordApi';

export default class Record extends LightningElement {
    @api fields;
    @api recordId;
    record;
}
```

What must be done in the component to get the data from Salesforce?

Answered Wrong

Ask Instructor

1. Add the following code above `record;`
```
@api(getRecord, { recordId: '$recordId', fields: '$fields' })
```

2. Add the following code above `record;`
```
@wire(getRecord, { recordId: '$recordId', fields: '$fields' })
```
(Selected Answer)

3. Add the following code above `record;`
```
@api(getRecord, { recordId: '$recordId' })
```
Get the fields in `renderedCallback()` and assign them to `record`.

4. Add the following code above `record;`
```
@wire(getRecord, { recordId: '$recordId' })
```
Get the fields in `renderedCallback()` and assign them to `record`.
(Correct Answer)

A newly added scenario and an explanation will be given earlier.

Q.5. When developing a lightning web component, which setting displays lightning-layout-items in one column on small devices, such as mobile phones, and in two columns on table-size and desktop-size screens?

Answered Wrong

Ask Instructor

1. Set size="12" table-device-size="6"
2. Set size="12" medium-device-size="6" (Correct Answer)
3. Set size="6" mobile-device-size="12"
4. Set size="6" small-device-size="12" (Selected Answer)

Create Notes

This question is related to the concept of the lightning-layout-item in the lightning web component.
The requirement given in this question is:

- Which setting displays lightning-layout-items in one column on small devices, such as mobile phones, and in two columns on table-size and desktop-size screens.
- For example, we have some buttons, so on desktop size screen, these buttons should be displayed as TWO COLUMNS and in small devices like mobile phones, they should display as ONE COLUMN.

Layout Item
The basic element in a lightning-layout component. A layout item groups information together to define visual grids, spacing, and sections.
A lightning-layout-item defines the content to display within lightning-layout. You can arrange one or more lightning-layout-item components inside lightning-layout.
Use the attributes of the lightning-layout-item to configure the size of the layout item, and change how the layout is configured on different device sizes. If you specify the small-device-size, medium-device-size, or large-device-size attributes, you must also specify the size attribute.
Let's understand some attributes:

- **size:**
  - o If the viewport is divided into 12 parts, size indicates the relative space the container occupies. Size is expressed as an integer from 1 through 12. **This applies to all device-types.**
- **medium-device-size:**

o If the viewport is divided into 12 parts, this attribute indicates the relative space the container occupies on device-types larger than tablets. It is expressed as an integer from 1 through 12.
Now, coming back to the question, it says that for small devices like mobile phones, it should be displayed as ONE COLUMN. So, it we set size=12, it will display as ONE COLUMN for all device types including mobile.
Additionally, we have a requirement that, on the desktop size screen, it should be displayed as TWO COLUMNS. So here we can set **the medium-device-size** attribute to 6 so it will display as TWO COLUMNS On the desktop.

Therefore, to fulfill this requirement, we can use: **size="12" medium-device-size="6"**
With this explanation,the correct answer is:

Set size="12" medium-device-size="6"Reference:

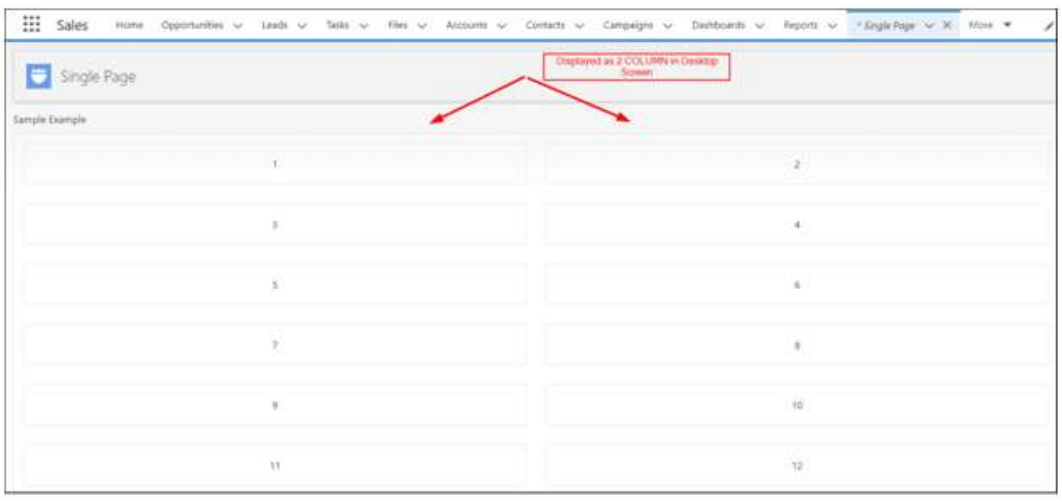
https://developer.salesforce.com/docs/component-library/bundle/lightning-layout-item/example

Screenshot – Sample Code using Lightning Layout Item

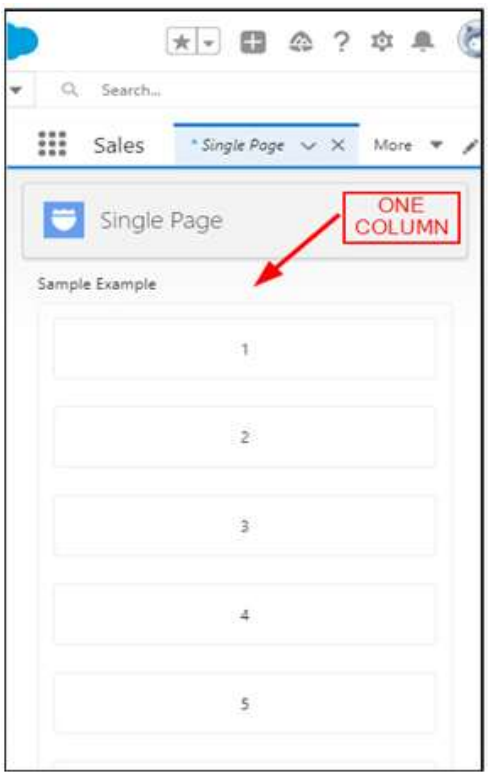
```
<template>
    <p>Sample Example</p>

<div class="slds-box slds-p-around_none slds-m-top_x-small slds-m-bottom_medium slds-m-horizontal_none">
    <lightning-layout multiple-rows>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">1</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">2</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">3</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">4</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">5</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">6</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">7</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">8</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">9</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">10</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">11</div>
        </lightning-layout-item>
        <lightning-layout-item size="12" medium-device-size="6" padding="around-small">
            <div class="custom-box slds-box slds-p-around_medium slds-text-align_center">12</div>
        </lightning-layout-item>
    </lightning-layout>
</div>
</template>
```

Create Notes

Screenshot – Displayed as 2 COLUMNS on desktop screen



Screenshot – Displayed as 1 COLUMN in a small device like a mobile



Q.6. A company manages information about their product offering in custom objects named Catalog and catalog item. Catalog Item has a master-detail field to Catalog, and each Catalog may have as many as 1,00,000 Catalog Items.
Both custom objects have a CurrencyIsoCode Text field that contains the currency code they should use. If a Catalog's CurrencyIsoCode changes, all of its Catalogue items' CurrencyIsoCodes should be changed as well.
What should a developer use to update the CurrencyIsoCodes on the Catalog Items when the Catalog's CurrencyIsoCode changes?

Answered Right

Ask Instructor

1. An *after insert* Trigger on Catalog that updates the Catalog Items if the Catalog's CurrencyIsoCode is different.

2. A *Database.Schedulable* and *Database.Batchable* class that queries the Catalog object and updates the Catalog Items if the Catalog CurrencyIsoCode is different.

3. A *Database.Schedulable* and *Database.Batchable* class that queries the Catalog Item object and updates the Catalogue Items if the Catalog CurrencyIsoCode is different. (Correct Answer)(Selected Answer)

4. An *after insert* Trigger on Catalog Item that updates the Catalog Items if the Catalog's CurrencyIsoCode is different.

Create Notes

This is a scenario-based question. The scenario given in this question is:

- There are two custom objects:
  - o Catalog
  - o Catalog Item
- Catalog Item has Master-Detail relationship to Catalog object
- Each Catalog record may have 1,00,000 (1 Lakh) catalog items.
- Both custom objects have a CurrencyIsoCode Text field. If a Catalog's CurrencyIsoCode changes, all of its Catalogue items' CurrencyIsoCodes should be changed as well.

Observation:

- Each Catalog record may have 1,00,000 (1 Lakh) catalog items which is a huge amount of records to update synchronously. Therefore, some asynchronous mechanisms should be used to solve this requirement.
- We need to use Batch Apex because it can be used to Run large jobs that would exceed normal processing limits.

With this explanation, the correct answer is:

"A Database.Schedulable and Database.Batchable class that queries the Catalog Item object and updates the Catalogue Items if the Catalog CurrencyIsoCode is different." because in this approach it is querying the child records (Catalog Items) which can be lakhs of records.
"A Database.Schedulable and Database.Batchable class that queries the Catalog object and updates the Catalog Items if the Catalog CurrencyIsoCode is different." is **NOT CORRECT** because querying parent can have lakhs of child data which you cannot process in execute in any single batch chunks.

References:
https://trailhead.salesforce.com/content/learn/modules/asynchronous_apex/async_apex_introduction
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_batch_interface.htm

---

Q.7.

```
global with sharing class MyRemoter {
    public String accountName { get; set; }
    public static Account account { get; set; }
    public MyRemoter() {}

    @RemoteAction
    global static Account getAccount(String accountName) {
        account = [SELECT Id, Name, NumberOfEmployees
                   FROM Account WHERE Name = :accountName];
        return account;
    }
}
```

Consider the Apex class above that defines a RemoteAction used on a Visualforce search page.
Which code snippet will assert that the remote action returned the correct Account?

Answered Wrong

Ask Instructor

1.
```
MyRemoter remote = new MyRemoter();
Account a = remote.getAccount('TestAccount');
System.assertEquals( 'TestAccount', a.Name );
```

2.
```
MyRemoter remote = new MyRemoter('TestAccount');
Account a = remote.getAccount();
System.assertEquals( 'TestAccount', a.Name );
```

3.
```
Account a = MyRemoter.getAccount('TestAccount');
System.assertEquals( 'TestAccount', a.Name );
```

(Correct Answer)

4.
```
Account a = controller.getAccount('TestAccount');
System.assertEquals( 'TestAccount', a.Name );
```

(Selected Answer)

Create Notes

This question is related to RemoteAction Annotation and the writing test class related to it. The scenario given in this question is:

- A sample apex class code that defines a RemoteAction is given.

The ask is:

- Which code snippet will assert that the remote action returned the correct Account?

Let's see some basics about RemoteAction

- The RemoteAction annotation provides support for Apex methods used in Visualforce to be called via JavaScript. This process is often referred to as JavaScript remoting.
- Methods with the RemoteAction annotation must be static and either global or public.
- Apex method declaration is preceded with the @RemoteAction annotation
- Since the method with the RemoteAction annotation must be static, the method can be accessed by using ClassName.methodName syntax.

Screenshot – Sample Apex Class is given in the question

MyRemoter.apxc

Code Coverage: None API Version: 54

```
1  global with sharing class MyRemoter {
2      public String accountName {get;set;}
3      public static Account account {get;set;}
4
5      //constructor
6      public MyRemoter(){
7
8      }
9
10     //This method has @RemoteAction annotation
11     @RemoteAction
12     global static Account getAccount(String accountName){
13         account = [SELECT Id, Name FROM Account
14                    WHERE Name= :accountName];
15         return account;
16     }
17 }
```

Screenshot – Apex Test Class for a class given in this question

MyRemoterTest.apxc

Code Coverage: None API Version: 54

```
1  /* Test Class for Apex Class - MyRemoter */
2  @isTest
3  private class MyRemoterTest {
4
5      @testSetup
6      static void setupTestData(){
7          //create the test data for Account record
8          List<Account> testAccts = new List<Account>();
9          for(Integer i=0;i<1;i++) {
10             testAccts.add(new Account(Name = 'TestAcct'+i));
11         }
12         insert testAccts;
13     }
14
15      @isTest
16     static void getAccountTest() {
17         Account acct = [SELECT Id, Name FROM Account LIMIT 1];
18         Account a = MyRemoter.getAccount(acct.Name);
19
20         //asserting
21         System.assertEquals(a.Name, acct.Name);
22     }
23 }
```

Testing Apex Method annoted by @RemoteActio n

With this explanation and code sample, below is the correct answer

```
Account a = MyRemoter.getAccount('TestAccount');
System.assertEquals( 'TestAccount', a.Name );
```

because it uses ClassName.methodName format to invoke the apex class

Reference:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_classes_annotation_RemoteAction.htm

---

Q.8. When calling a RESTful web service, the developer must implement two-way SSL authentication to enhance security. The Salesforce admin has generated a self-sign certificate within Salesforce with a unique name of "ERPSecCertificate"'
Consider the following code snippet:
HttpRequest req = new HttpRequest ();
... // rest of request

Create Notes

Which method must the developer implement in order to sign the HTTP request with the certificate?

Answered Wrong

Ask Instructor

1. *req.setHeader ('certificate', 'ERPSecCertificate');* (Selected Answer)
2. *req.setSecure ('ERPSecCertificate)';*
3. *req.setClientCertificateName ('ERPSecCertificate');* (Correct Answer)
4. *req.setSecureCertificate ('ERPSecCertificate');*

This is a scenario-based question related to using certificates with HTTP requests.
The scenario given in this question is:

- The developer is using two-way SSL authentication to enhance security.
- The Salesforce admin has generated a self-sign certificate within Salesforce with a **unique name of "ERPSecCertificate".**

The ask is:

- Which method must the developer implement in order to sign the HTTP request with the certificate.

Let's learn about Using Certificate with HTTP Requests
You can use two-way SSL authentication by sending a certificate generated in Salesforce or signed by a certificate authority (CA) with your callout. This enhances security as the target of the callout receives the certificate and can use it to authenticate the request against its keystore.

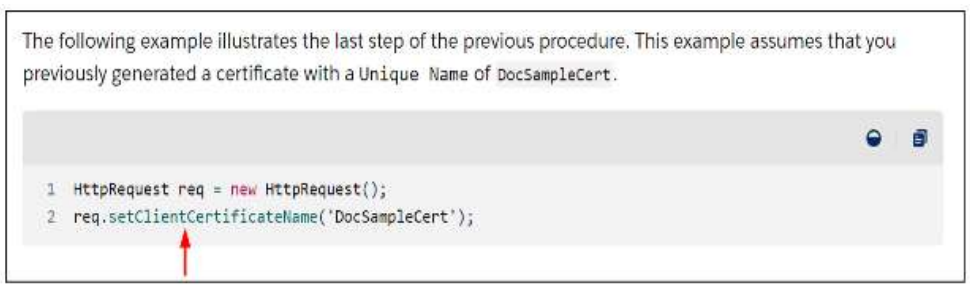
To enable two-way SSL authentication for a callout:

1. Generate a certificate.
2. Integrate the certificate with your code. (Either SOAP or REST services)
3. If you are connecting to a third-party and you are using a self-signed certificate, share the Salesforce certificate with them so that they can add the certificate to their keystore.

Please Note:

- After you have generated a certificate in Salesforce, you can use it to support two-way authentication for a callout to an HTTP request.
- To integrate the certificate with your Apex:
  - **Generate a certificate.** Note the Unique Name of the certificate.
  - In your Apex, use the **setClientCertificateName method** of the HttpRequest class. The value used for the argument for this method must match the Unique Name of the certificate that you generated in the previous step.

Screenshot – Using method setClientCertificateName


The following example illustrates the last step of the previous procedure. This example assumes that you previously generated a certificate with a `Unique Name` of `DocSampleCert`.

```
HttpRequest req = new HttpRequest();
req.setClientCertificateName('DocSampleCert');
```

With this explanation,

the correct answer is:
"req.setClientCertificateName ('ERPSecCertificate')" because this method is used to set the certificate name for authentication.
References:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_callouts_client_certs_http.htm
https://developer.salesforce.com/docs/atlas.en-us.204.0.apexcode.meta/apexcode/apex_callouts_client_certs.htm

---

**Q.9. A developer created a JavaScript library that simplifies the development of repetitive tasks and features and uploaded the library as a static resource called jsUtils in Salesforce. Another developer is coding a new Lightning web component (LWC) and wants to leverage the library.
Which statement properly loads the static resource within the LWC?**

Answered Wrong

Create Notes

Ask Instructor

1. *import {jsUtilities} from '@salesforce/resourceUrl/jsUtils';* (Selected Answer)
2. *import jsUtilities from '@salesforce/resourceUrl/jsUtils';* (Correct Answer)
3. *const jsUtility = $A.get('$Resource.jsUtils');*
4. *<lightning-require scripts="{!$Resource.jsUtils}"/>*

This question is related to using the JavaScript library in lightning web components. The scenario given in this question is:

- A developer created a JavaScript library and uploaded the library as a static resource called jsUtilsin Salesforce.
- Another developer is coding a new Lightning web component (LWC) and wants to leverage the library.

Please Note:
You can use third-party JavaScript libraries with Lightning web components. For example, use a library with interactive charts and graphs or a library that reduces code complexity.

Screenshot – Steps to use JavaScript library in Salesforce

With this explanation, the correct answer is:
import jsUtilities from '@salesforce/resourceUrl/jsUtils';

Reference:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/lwc.js_third_party_library

---

Q.10. A developer has a test class that creates test data before making a mock callout but now receives a 'You have uncommitted work pending. Please commit or rollback before calling out' error.
Which step should be taken to resolve the error?

Answered Right

Ask Instructor

1. Ensure the records are inserted before the *Test.startTest()* statement and the mock callout occurs within a method annotated with *@testSetup.*
2. Ensure the records are inserted before the *Test.startTest()* statement and the mock callout occurs after the *Test.startTest().* (Correct Answer)(Selected Answer)
3. Ensure both the insertion and mock callout occur after the *Test.stopTest().*
4. Ensure both the insertion and mock callout occur after the *Test.startTest().*

This question is related to performing DML operations and Mock Callouts in Salesforce. As per salesforce documentation, we need to keep a few points in mind while writing test classes for HTTP callouts.
By default, callouts aren't allowed after DML operations in the same transaction because DML operations result in pending uncommitted work that prevents calloutsfrom executing. Sometimes, you might want to insert test data in your test method using DML before making a callout. To enable this, enclose the portion of your code thatperforms the callout within Test.startTest and Test.stopTest statements.

The Test.startTest statement must appear before the Test.setMock statement. Also, the calls to DML operations must not be part of the Test.startTest/Test.stopTest block.

Create Notes

With this explanation, the correct answer is:

"Ensure the records are inserted before the Test.startTest() statement and the mock callout occurs after the Test.startTest()"

For Sample code and Reference:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_classes_restful_http_testing_dml.htm

---

Q.11. There is an Apex controller and a Visualforce page in an org that displays records with a custom filter consisting of a combination of picklist values selected by the user.
The page takes too long to display results for some of the input combinations, while for other input choices it throws the exception, "Maximum view state size limit exceeded".
What step should the developer take to resolve this issue?

Answered Right

Ask Instructor

1. Use a StandardSetController or SOQL LIMIT in the Apex controller to limit the number of records displayed at a time.
(Correct Answer)(Selected Answer)
2. Remove instances of the transient keyword from the Apex controller to avoid the view state error.
3. Split the layout to filter records in one Visualforce page and display the list of records in a second page using the same Apex controller.
4. Adjust any code that filters by picklist values since they are not indexed.

This is a scenario-based question related to performance of Visualforce page. This question is specifically related to Controlling Data Size.
Please Note:
When designing Visualforce pages, don't overload pages with functionality and data. Visualforce pages with unbounded data or a large number of components, rows, and fields have poor usability and performance, and they risk hitting governor limits for view state, heap size, record limits, and total page size.

Controlling Data Size
Visualforce pages can't exceed the 15 MB standard response limit, but even smaller page sizes affect load time. To minimize load times, use the following techniques to further limit the amount of data each page displays:

1. **Use Filters**

Use filters to limit the data that SOQL calls in and Apex controllers return. For example, use AND statements in your WHERE clause or remove null results.

2. **Use keywords**

When creating Apex controllers, use the with sharing keyword to retrieve only the records the user can access.

3. **StandardSetController Built-In Pagination**

Use the built-in pagination functionality in list controllers to prevent list views from displaying unbounded data. Unbounded data might cause longer load times, hit governor limits, and become unusable as the data set grows. By default, a list controller returns 20 records on the page, but developers often configure list views to display up to 100 records at a time.

4. **Use SOQL OFFSET Pagination**

Use the SOQL OFFSET clause to write logic that paginates to a specific subset of results within SOQL.

With this explanation, the correct answer is:
"Use a StandardSetController or SOQL LIMIT in the Apex controller to limit the number of records displayed at a time".

Reference:
https://blog.bessereau.eu/assets/pdfs/salesforce_visualforce_best_practices.pdf

---

Q.12. Northern Trail Outfitters (NTO) has a custom object, ServiceJob__c, with an optional Lookup field to Account called Partner_Service_Provider__c. The

Create Notes

TotalJobs__c field on Account tracks the total number of ServiceJob__c records to which a partner service provider Account is related.
What should be done to ensure that the TotalJobs__c field is kept up to date?

Answered Right

Ask Instructor

1. Build a Process Builder with an invocable action.
2. Create an Apex Trigger on ServiceJob__c. (Correct Answer)(Selected Answer)
3. Implement a workflow cross-object field update.
4. Change TotalJobs__c to a roll-up summary field.

This is a scenario-based question. The scenario given in this question is:

- There are two objects involved:
  - ServiceJob__c
  - Account
- There is a lookup relationship field "Partner_Service_Provider__c" on the ServiceJob__c object to the Account object.
- There is a field on the Account object (TotalJobs__c) that tracks the total number of ServiceJob__c records related to that Account

The requirement is:

- What should be done to ensure that the TotalJobs__c field is kept up to date.

In Salesforce, we have a concept of "Roll-Up Summary Field" that calculates values from related records, such as those in a related list. You can create a roll-up summary field to display a value in a master record based on the values of fields in a detail record. But, the important point is, that the "Roll-Up Summary Field" works with **Master-Detail Relationships** and the scenario is given in this question has a lookup relation.

Therefore, we need some custom solution here that updates the TotalJobs__c on the Account based on the ServiceJob__c records.
This can be achieved by writing Apex Trigger on the ServiceJob__c object for all operations like insert, update, and delete.

With this explanation, the correct answer is:

"Create an Apex Trigger on ServiceJob__c."

For Reference & Similar example:
http://varasi.com/salesforce/implementing-roll-up-summary-when-you-are-dealt-with-a-lookup-relationship/

Q.13. A business process requires sending new Account records to an external system. The Account Name Id, CreatedDate, and CreatedById must be passed to the external system in near real-time when an Account is inserted without error.
How should a developer achieve this?

Answered Right

Ask Instructor

1. Use a Process Builder that calls an *@InvocableMethod* method. (Correct Answer)(Selected Answer)
2. Use a *before insert* trigger and an *@future* method.
3. Use a *before insert* trigger and a *Queueable* class.
4. Use a Workflow rule that calls in *@InvocableMethod* method.

This is a scenario-based question which is related to integrating Salesforce with external system. The scenario given in the question is:

- There is a requirement to send new Account records to an external system.
- The fields like Id, Name, CreatedDate, CreatedById must be passed to the external system must be passed to the external system.
- This should happen when an Account is inserted.
- This should happen in real-time.

There can be different ways to achieve this.

Create Notes

Let's go through each option given in this question and decide.
Option **"Use a before insert trigger and an @future method"** is **not correct** because as per the requirement, we need to send Id, CreatedDate fields which we cannot get in before triggers.
Option **"Use a before insert trigger and a Queueable class"** is **not correct** because as per the requirement, we need to send Id, CreatedDate fields which we cannot get in before triggers.
Option **"Use a Workflow rule that calls in @InvocableMethod method"** is **not correct** because we cannot use @InvocableMethod from workflow rule.

Option **"Use a Process Builder that calls an @InvocableMethod method"** is **correct** because:

- We can have an invokable apex method which can make callout to an external system.
- The process builder can call this apex method once the account record is created.

References:
https://developer.salesforce.com/docs/atlas.en-us.236.0.apexcode.meta/apexcode/apex_classes_annotation_InvocableMethod.htm
https://help.salesforce.com/s/articleView?id=sf.process_action_apex.htm&type=5

Q.14. A developer created a Lightning web component that uses a lightning-record-edit-form to collect information about Leads. Users complain that they only see one error message at a time about their input when trying to save a Lead record.
Which best practice should the developer use to perform the validations on more than one field, thus allowing more than one error message to be displayed simultaneously?

Answered Wrong

Ask Instructor

1. Client-side validation. (Selected Answer)
2. Next Best Action.
3. Custom validation rules. (Correct Answer)
4. Apex REST.

This is a scenario-based question related to the lightning web component and client-side validation. The scenario given in this question is:

- Lightning web component uses lightning-record-edit-form to collect information about Leads.
- Users complain that they only see one error message at a time about their input when trying to save a Lead record.

That means the requirement here is to show different messages / specific messages depending on the validation.
To achieve this, there is a need to add multiple validation rules so that different error messages can be displayed.
As per Salesforce documentation for lightning-record-edit-form

> **lightning-record-edit-form** also verifies data input based on your validation rules. The form submits and saves data input only if all data in the fields are valid. The form clears validation rule errors when an **onchange** event is fired on the overall form, and also when you update a field with a validation rule error.
>
> If a single field has multiple validation errors, the form shows only the first error on the field. Similarly, if a submitted form has multiple errors, the form displays only the first error encountered. When you correct the displayed error, the next error is displayed.
>
> We recommend using custom validation rules to verify data input instead of implementing client-side validation errors. A validation rule can contain a formula or expression that evaluates the data in one or more fields. You can include an error message to display on an invalid field. See Validation Rules in Salesforce Help for more information.

With this explanation, the correct answer is

Create Notes

"Custom validation rules" is correct as per recommendation from Salesforce while using lightning-record-edit-form.

Reference:
https://developer.salesforce.com/docs/component-library/bundle/lightning-record-edit-form/documentation

---

Q.15. An org has a requirement that addresses on Contacts and Account should be normalized to a company standard by Apex code anytime that they are saved.
What is the optimal way to implement this?

Answered Right

Ask Instructor

1. Apex trigger on Contact that calls the Account trigger to normalize the address.
2. Apex triggers on Contact and Account that normalize the address.
3. Apex trigger on Account that calls the Contact trigger to normalize the address.
4. Apex triggers on Contact and Account that call a helper class to normalize the address. (Correct Answer)(Selected Answer)

This is a scenario-based question related to Apex Triggers. The scenario given in this question is:

- Whenever Contact and Account are saved, the addresses (Street, Zip code, City, State, Country) should be normalized according to company standards. For example, the address should be stored in capital letters.

The ask is:

- What is the optimal way to implement this?

The solution approach to implement this is by using Apex Triggers and as per Salesforce best practice, we should write reusable code in apex class methods which can be reused from multiple places.
Considering this requirement,

- As the requirement is for both Contacts and Accounts. We need two triggers, one on the Account object and the other on the Contact object.
- We should not write core logic in Triggers itself instead we should use helper classes to write common logic which can be reused.

With this explanation, the correct answer is:

"Apex triggers on Contact and Account that call a helper class to normalize the address."

Let's assume, before insert of Account and Contact, the address needs to be in Capital letters.

Screenshot – Account Trigger utilizing helper method

AccountTrigger.apxt

Code Coverage: None API Version: 53

```
1  trigger AccountTrigger on Account (before insert) {
2
3      if(Trigger.isBefore){
4          if(Trigger.isInsert){
5
6              /**
7               * call helper method to perform logic
8               * logic here is to CAPITALIZE the address fields on Account records
9               */
10             TriggerHelper helper = new TriggerHelper();
11             helper.capitalizeAddress(Trigger.New);
12         }
13     }
14
15 }
```

Using helper method from Account Trigger

Screenshot – Contact Trigger utilizing helper method

Create Notes

ContactTrigger.apxt

Code Coverage: None API Version: 53

```
trigger ContactTrigger on Contact (before insert) {

    if(Trigger.isBefore){
        if(Trigger.isInsert){

            /**
             * call helper method to perform logic
             * logic here is to CAPITALIZE the address fields on Contact records
             */
            TriggerHelper helper = new TriggerHelper();
            helper.capitalizeAddress(Trigger.New);
        }
    }
}
```

Using helper methods from Contact Trigger

Screenshot – helper method

TriggerHelper.apxc

Code Coverage: None API Version: 54

```
/**
 * Trigger helper having methods
 * which can be reused from multiple triggers or
 * classes
 */

public class TriggerHelper {

    public void capitalizeAddress(List<sObject> listRecords){

      // logic here is to CAPITALIZE the address fields on Contact records

    }
}
```

Reusable Method

Q.16. A company has many different unit test methods that create Account records as part of their data setup. A new required field was added to the Account and now all of the unit tests fail.
What is the optimal way for a developer to fix the issue?

Answered Right

Ask Instructor

1. Add a before insert trigger on Account to set the value of the required field.
2. Create a TestDataFactory class that serves as a single place to create Accounts for unit tests and set the required field there.
(Correct Answer)(Selected Answer)
3. Add the required field to the data setup for all of the unit tests.
4. Change the required field to be a validation rule that excludes the System administrator profile.

This is a scenario-based question related to Apex Test Class.

The scenario given in this question is:

- In the current situation, a company has many different unit test methods that create Account records as part of their data setup.
- A new required field was added to the Account and now all of the unit tests fail.
    - The reason behind the failure of all the test methods is Account creation without that new required field.

That means, from now, whenever the test Account records need to be created, the required field needs to be populated.

There are different ways to fix it like “Add the required field to the data setup for all of the unit tests” but this is not the appropriate way to do it.
The best way to do this is to have a single TestDataFactory class that will have a single method to create an Account unit test. This method can be called from any unit test method to create a test account. The benefit of doing this is it has very less maintenance. For example, if any other field is marked as required in the future, we need to just change this method in TestDataFactory class and it will work fine.

With this explanation, the correct answer is:

Create Notes

"Create a TestDataFactory class that serves as a single place to create Accounts for unit tests and set the required field there."

For Reference & Sample Code:
https://trailhead.salesforce.com/content/learn/modules/apex_testing/apex_testing_data?trailmix_creator_id=weina&trailmix_slug=sf-for-beginner-intermediate

---

Q.17. Refer to the code below:
Apex Class:

```
public class OppController {
    public List<Opportunity> getTopOpps() {
        return [SELECT Name FROM Opportunity ORDER BY Amount DESC LIMIT 10];
    }
}
```

Component markup:

```
<aura:component>
    <aura:handler name="init" value="{!this}" action="{!c.doInit}"/>
</aura:component>
```

Component controller:

```
({
    doInit : function(cmp, event, helper) {
        var action = cmp.get("c.getTopOpps");
        action.setCallback(this, function(response) {
            var state = response.getState();
            if (state === "SUCCESS") {
                console.log("From server: " + response.getReturnValue());
            }
        });
        $A.enqueueAction(action);
    }
})
```

A developer is building this Aura component to display information about top Opportunities in the org.
Which three code changes must be made for the component to work?
Choose 3 answers.

Answered Right

Ask Instructor

1. Add the *static* keyword to the Apex method. (Correct Answer)(Selected Answer)
2. Set the controller in the component markup. (Correct Answer)(Selected Answer)
3. Add the *RemoteAction* annotation to the Apex method.
4. Add the *AuraEnabled* annotation to the Apex method. (Correct Answer)(Selected Answer)
5. Get the controller action with *cmp.get ("OppController.getTopOpps")*.

This question is simple and its related to access apex methods from lightning aura components.

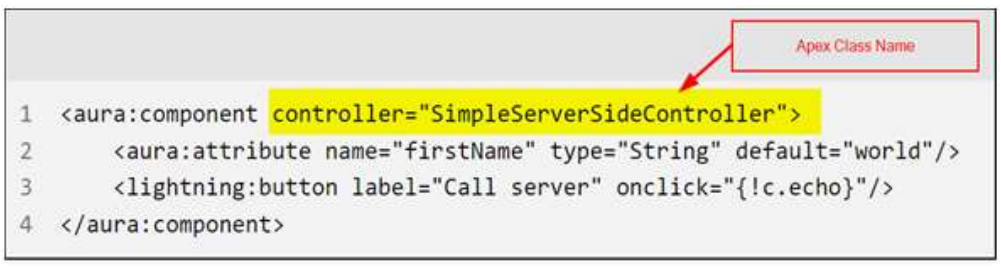
Let's Learn about calling apex methods from Lightning Aura Components:
1. AuraEnabled Annotation

- The AuraEnabled annotation enables Lightning components to access Apex methods and properties.
- Use @AuraEnabled on Apex class **static methods** to make them accessible as remote controller actions in your Lightning components.
- Use @AuraEnabled on Apex instance methods and properties to make them serializable when an instance of the class is returned as data from a server-side action.

Screenshot – AuraEnabled Annotation

2. To access an apex class from Aura Component, we need to set controller attribute to ApexClassName.

Create Notes



With this explanation, the correct answers are:

"Add the static keyword to the Apex method."

and
"Add the AuraEnabled annotation to the Apex method."

and
"Set the controller in the component markup."

References:
https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/controllers_server_apex_auraenabled_annotation.htm
https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/controllers_server_actions_call.htm

---

Q.18. An Apex trigger creates an Order__c record every time an Opportunity is won by a Sales Rep. Recently the trigger is creating two orders.
What is the optimal method for a developer to troubleshoot this?

Answered Wrong

Ask Instructor

1. Add system.debug() statements to the code and use the Developer Console logs to trace the code. (Correct Answer)
2. Run the Apex Test Classes for the Apex trigger to ensure the code still has sufficient code coverage.
3. Set up debug logging for every Sales Rep, then monitor the log for errors and exceptions.
4. Turn off all Workflow Rules, then turn them on one at at time to see which one causes the error. (Selected Answer)

This is scenario-based question and is related to debugging the code in Salesforce. The scenario given in this question is:

- An Apex trigger creates an Order__c record every time an Opportunity is won by a Sales Rep.

The issue is, the Apex Trigger is creating two orders so how a developer can troubleshoot and figure out the root-cause.
Let's go by each option given in this question.
Option **"Turn off all Workflow Rules, then turn them on one at time to see which one causes the error."** is **not correct** because this is not feasible way to debug as there can be many workflow rules.
Option **"Set up debug logging for every Sales Rep, then monitor the log for errors and exceptions."** is **not correct** because this issue not for specific Sales Rep, so no need to setup debug log for every Sales Rep.
Option **"Run the Apex Test Classes for the Apex trigger to ensure the code still has sufficient code coverage."** is **not correct** because Test Class coverage will not ensure to find the exact root-cause of this issue.

The **correct** answer is:
"Add system.debug() statements to the code and use the Developer Console logs to trace the code." because adding system.debug() in the code will help to understand which methods are getting called and we can use developer console logs to trace the code.
References:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_debugging_debug_log.htm
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_debugging_system_log_console.htm

---

Create Notes

Q.19. A company has a web page that needs to get Account record information, such as name, website, and employee number. The Salesforce record ID is known to the web page and it uses JavaScript to retrieve the account information.
Which method of integration is optimal?

Answered Wrong

Ask Instructor

1. Apex REST web service (Selected Answer)
2. SOAP API
3. Apex SOAP web service.
4. REST API (Correct Answer)

This is a scenario-based question related to access Salesforce data from JavaScript. The scenario given in this question is:

- A company has a web page that needs to get Account record information like Name, website, employee number etc.
- The Salesforce recordId is known to the web page.
- The web page uses JavaScript to retrieve the account information.

The ask is:

- Which method of Integration is optimal?

For this requirement, the optimal way of integration would be: **REST API**

> **REST API**
>
> REST API is a simple and powerful web service based on RESTful principles. It exposes all sorts of Salesforce functionality via REST resources and HTTP methods. For example, you can create, read, update, and delete (CRUD) records, search or query your data, retrieve object metadata, and access information about limits in your org. REST API supports both XML and JSON.
>
> Because REST API has a lightweight request and response framework and is easy to use, it's great for writing mobile and web apps.

This will be a simple REST API that just accepts the recordId and provide relevant record information.
With this explanation, the correct answer is:

"REST API"

References:
https://trailhead.salesforce.com/en/content/learn/modules/api_basics/api_basics_overview
https://trailhead.salesforce.com/content/learn/modules/mobile_sdk_hybrid/mobilesdk_hybrid_rest_apis

---

Q.20. A company has reference data stored in multiple Custom Metadata records that represent default information and delete behavior for certain geographic regions. When a contact is inserted, the default information should be set on the contact from the custom metadata records based on the contact's address information. Additionally, if a user attempts to delete a contact that belongs to a flagged region, the user must get an error message. What is the optimal way to automate this?

Answered Right

Ask Instructor

1. Apex invocable method.
2. Apex Trigger. (Correct Answer)(Selected Answer)
3. Flow Builder.
4. Remote action.

Create Notes

This is a scenario-based question related to implementing custom logic.

The scenario given in this question is:

- Reference data stored in multiple Custom Metadata records that represent default information and delete behavior for certain geographic regions.
- When a contact is inserted, the default information should be set on the contact from the custom metadata records based on the contact's address information.
- if a user attempts to delete a contact that belongs to a flagged region, the user must get an error message.

The ask is:

- What is the optimal way to automate this?

With the above-mentioned scenarios, below are the important points / requirements:

- On Contact creation, setting the default information.
- On Contact deletion, show error and stop deletion in certain scenarios.

For this requirement, the best approach is to use **Apex Trigger** as

- We can write logic to fetch information from Custom Metadata and can set default information while contact creation.
- Also, we can use **addError()** method to show the error and to prevent DML operations.

With this explanation, the correct answer is:
"Apex Trigger."
Reference:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_triggers_exceptions.htm

---

**Q.21. A company has a custom object, Order__c, that has a required, unique external ID field called Order_Number__c.
Which statement should be used to perform the DML necessary to insert new records and update existing records in a list of Order__c records using the external ID field?**

Answered Right

Ask Instructor

1. *merge orders Order_Number__c;*
2. *upsert orders;*
3. *upsert orders Order_Number__c;*
(Correct Answer)(Selected Answer)
4. *merge orders;*

This is a scenario-based question which is related to updating or inserting records using external id. The scenario given in this question is:

- There is a custom object named "Order__c" that has a required, unique external ID field called Order_Number__c.
- There is a list of Order__c records called as "orders".**The ask is:**
- How to perform DML to insert new records and update existing records in a list of Order__c records using the external ID field.

Let's learn about upsert DML statement.
Using the upsert operation, you can either **insert** or **update** an existing record in one call. To determine whether a record already exists, the upsert statement uses the record's ID as the key to match records, a custom external ID field, or a standard field with the idLookup attribute set to true.

- If the key is not matched, then a new object record is created.
- If the key is matched once, then the existing object record is updated.
- If the key is matched multiple times, then an error is generated and the object record is neither inserted or updated.

Create Notes

Upsert with externalId

- Use of upsert with an external ID can reduce the number of DML statements in your code, and help you to avoid hitting governor limits.

- The syntax to use upsert statement with external id is:
  - Upsert <list of sobject> <API Name of external id field>

With this explanation,
the correct answer of this question is:
"upsert orders Order_Number__c;" because we need to use upsert statement with the name of external Id field "Order_Number__c".
Reference:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/langCon_apex_dml_examples_upsert.htm

---

Q.22. Salesforce users consistently receive a "Maximum trigger depth exceeded" error when saving an Account.
How can a developer fix this error?

Answered Right

Ask Instructor

1. Use a helper class to set a Boolean to TRUE the first time a trigger is fired, and then modify the trigger to only fire when the Boolean is FALSE. (Correct Answer)(Selected Answer)
2. Convert the trigger to use the *@future* annotation, and chain any subsequent trigger invocations to the Account object.
3. Modify the trigger to use the *isMultiThread=true* annotation.
4. Split the trigger logic into two separate triggers.

This question is related to the Error - 'Maximum Trigger Depth Exceeded'. Let's understand about the root cause of this error and how to resolve it.
As per the name 'Maximum Trigger Depth Exceeded', it means the Apex Trigger code is executing multiple times recursively. Normally 'Maximum Trigger depth Exceeded' error occurs when a trigger recursively invokes triggers more than 16 times (Maximum stack depth).
To avoid this, we have to break the recursive call. One way is to use static Boolean flag wherever your business logic permits.
Resolution - **"Maximum trigger depth exceeded"** or **"Recursive Trigger".**

1. Create a class with static Boolean variable and set its value true to avoid the exception.

TriggerController.apxc

Code Coverage: None API Version: 53

```
1  public class TriggerController {
2
3      // Boolean variable to stop recursion
4      public static Boolean isFirstCall = true;
5  }
```

2. Check the Boolean flag value before process your code condition and set the flag value to false at the beginning of your process. After setting flag value (Boolean variable) false, the recursion will stop and your code will process without any exception.

AccountTrigger.apxt

Code Coverage: None API Version: 53

```
1  trigger AccountTrigger on Account (after update) {
2
3      if(Trigger.isAfter){
4          if(Trigger.isUpdate){
5
6              if(TriggerController.isFirstCall){
7                  TriggerController.isFirstCall = false;
8
9                  // code
10             }
11         }
12     }
13 }
```

Create Notes

With this explanation,

the correct answer of this question is:
"Use a helper class to set a Boolean to TRUE the first time a trigger is fired, and then modify the trigger to only fire when the Boolean is FALSE" because this is the resolution to stop firing trigger recursively.
Reference:
https://help.salesforce.com/s/articleView?id=000332407&type=1

Q.23. What is the optimal technique a developer should use to programmatically retrieve Global Picklist options in a Test Method?

Answered Right

Ask Instructor

1. Perform a SOQL Query.
2. Use the *Schema* namespace. (Correct Answer)(Selected Answer)
3. Use a static resource.
4. Perform a *callout* to the Metadata API.

This question is related to using Global Picklist options and accessing it in apex test methods.

Global Picklist Value Set
Use global picklist value sets to share values across objects and custom picklist fields, and to restrict the picklists to only the values that you specify.
A custom picklist is tied to a particular object as a field on the object. Unlike a custom picklist field, a global picklist exists independently as a global picklist value set. Its values are shared with any picklist that's based on it.

Access Picklist in Apex
Dynamic Apex is used commonly for accessing picklist values from a server controller or a trigger. Along with its use in dynamic SOQL, it lets users analyze objects and fields with methods from the Schema namespace. Using this technique, we can know the fields to an object, type of field, or values of the picklist field.
Few of the instance methods related to PicklistEntry to get this information are:

Methods	Function performed	Signature	Return Value (9 Types)
getValue()	It returns the values in the picklist.	public String getValue()	String
getLabel()	It gives the display name in the picklist.	public String getLabel()	String
isActive()	Return "true" if the item is displayed in the drop-down list for the picklist. Return "false" if not.	public Boolean isActive()	Boolean
isDefaultValue()	Return "true" if the item is the default value of the picklist. Return "false" if not.	public Boolean isDefaultValue()	Boolean

Using the above methods, we can retrieve picklist values with Apex in Salesforce. It is combined with getDescribe() and getPicklistvalues() methods to check the values of picklist.

Screenshot – Sample Code using Schema methods

Create Notes

Screenshot – Output of above code

With this explanation and code example,the correct answer is

“Use the Schema namespace”
Reference:
https://help.salesforce.com/s/articleView?id=sf.fields_creating_global_picklists.htm&type=5

Q.24. A developer wants to call an Apex server-side controller from an Aura component.
What are two limitations to the data being returned by the controller?
Choose 2 answers.

Answered Right

Ask Instructor

1. A custom Apex class can be returned, but only the values of public instance properties and methods annotated with *@AuraEnabled* are serialized and returned.
(Correct Answer)(Selected Answer)

2. Basic data types are supported, but defaults, such as maximum size for a number, are defined by the objects that they map to. (Correct Answer)
(Selected Answer)

3. Lists of custom Apex classes cannot be returned by Apex controllers called by Aura components.

4. Only basic data types and sObjects are supported as return types for Apex controllers called by Aura components.

This question is related to Returning Data from an Apex Server-Side Controller.
The apex methods can return the data which can be used by Aura Component.
The ask is:

- What are two limitations to the data being returned by the controller?

Returning Data from an Apex Server-Side Controller
Return results from a server-side controller to a client-side controller using the return statement. Results data must be serializable into JSON format. Return data types can be any of the following:

- Simple data – Like String or Integer
- sObject – Standard and Custom objects
- Apex – an instance of Apex Class
- Collection – a collection of any of the other types

Please Note:
When an instance of an Apex class is returned from a server-side action, the instance is serialized to JSON by the framework. Only the values of public instance properties and methods annotated with **@AuraEnabled** are serialized and returned.
Basic types are supported but since the framework is written in Java, defaults, such as maximum size for a number, for these basic types are defined by the Java objects that they map to.

With this explanation,

the correct answers are:
“A custom Apex class can be returned, but only the values of public instance properties and methods annotated with @AuraEnabled are serialized and returned.”

and
“Basic data types are supported, but defaults, such as maximum size for a number, are defined by the objects that they map to.”

Create Notes

References:
https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/controllers_server_apex_returning_data.htm

https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/ref_attr_types_basic.htm#ref_attr_types_basic

---

Q.25. An org has a requirement that an Account must always have one and only one Contact listed as Primary. So selecting one Contact will de-select any others. The client wants a checkbox on the Contact called 'Is Primary' to control this feature. The client also wants to ensure that the last name of every Contact is stored entirely in uppercase characters.
What is the optimal way to implement these requirements?

Answered Right

Ask Instructor

1. Write a single trigger on Contact for both after update and before update and callout to helper classes to handle each set of logic.
(Correct Answer)(Selected Answer)

2. Write an after update trigger on Contact for the Is Primary logic and a separate before update trigger on Contact for the last name logic.

3. Write an after update trigger on Account for the Is Primary logic and a before update trigger on Contact for the last name logic.

4. Write a Validation Rule on Control for the Is Primary logic and a before update trigger on Contact for the last name logic.

This is a scenario-based question which is related to Apex Triggers in Salesforce. The scenario / requirement given in this question is:

- There should be a checkbox on Contact named as IsPrimary.
- An org has a requirement that an Account must always have one and only one Contact listed as Primary. So, selecting one Contact will de-select any others.
- The client also wants to ensure that the last name of every Contact is stored entirely in uppercase characters.

The ask is:

- What can be the optimal way to implement these requirements.

Let's go through each option and decide about the optimal approach.
Option **"Write an after-update trigger on Contact for the Is Primary logic and a separate before update trigger on Contact for the last name logic"** is **not correct** because as per trigger best practice suggested by Salesforce, we should write only one trigger per object.
Option **"Write an after-update trigger on Account for the Is Primary logic and a before update trigger on Contact for the last name logic"** is **not correct** because as per the requirement, we need to write trigger on Contact and not on Account.
Option **"Write a Validation Rule on Control for the Is Primary logic and a before update trigger on Contact for the last name logic"** is **not correct** because for isPrimary requirement, we need business logic which cannot be achieved using validation rules.
Option **"Write a single trigger on Contact for both after update and before update and callout to helper classes to handle each set of logic"** is **correct** because as per Salesforce, this is the best practice to write single trigger and utilize helper class to
perform business logic.

Q.26. A developer wrote a test class that successfully asserts a trigger on Account. It fires and updates data correctly in a sandbox environment.
A Salesforce admin with a custom profile attempts to deploy this trigger via a change set into the production environment, but the test class fails with an insufficient privileges error.
What should a developer do to fix the problem?

Answered Right

Ask Instructor

Create Notes

1. Verify that *Test.startTest()* is not inside a FOR loop in the test class.
2. Configure the production environment to enable "Run All Tests as Admin User."
3. Add *seeAllData=true* to the test class to work within the sharing model for the production environment.
4. Add *System. runAs()* to the test class to execute the trigger as a user with the correct object permissions. (Correct Answer)(Selected Answer)

This is a scenario-based question related to the concept of Apex Trigger and Test Class. The scenario given in this question is:

- There is a Test Class to assert a trigger on Account.
- It fires and updates data correctly in a sandbox environment.
- A Salesforce admin with a custom profile attempts to deploy this trigger via a change set into the production environment, but the test class fails with an insufficient privileges error.

The ask is:

- What should a developer do to fix the problem?

The root-cause of this failure (test class fails with an insufficient privilege error) is that the test class is getting run by user who does not have sufficient privilege.
As per the Salesforce documentation for Testing Best Practice. We need to Use the **runAs** method to test your application in different user contexts.

RunAs Method:
Use the runAs method to test your application in different user contexts. The system method runAs enables you to write test methods that change either the user contexts to
an existing user or a new user. When running as a user, all of that user's record sharing is then enforced. You can only use runAs in a test method. The original system context is started again after all runAs test methods complete.

Please Note:
The runAs method ignores user license limits. You can create new users with runAs even if your organization has no additional user licenses.

Screenshot – Test Class which executes based on Specific User

With this explanation,

the correct answer is:
"Add *System. runAs()* to the test class to execute the trigger as a user with the correct object permissions."

Reference:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_testing_best_practices.htm

---

Q.27. A developer is creating a Lightning web component that can be added to a Lightning app page and displayed when the page is rendered in desktop and mobile phone format. To ensure a great mobile experience, the developer chooses to use the SLDS grid utility.
Which two Lightning web components should the developer implement to

Create Notes

ensure the application is mobile-ready?
Choose 2 answers.

Answered Right

Ask Instructor

1. <lightning-tree-grid></lightning-tree-grid>
2. <lightning-tree></lightning-tree>
3. <lightning-layout></lightning-layout>
(Correct Answer)(Selected Answer)
4. <lightning-layout-item></lightning-layout-item>
(Correct Answer)(Selected Answer)

This question is related to the concept of lightning web component with grid utility. The requirement is to keep the lightning web component mobile-ready. As per Salesforce documentation, we can use Lightning Layout and Layout Item to display the content based on device size.

. **Layout:** Represents a responsive grid system for arranging containers on a page.

. **Layout Item:** The basic element in a lightning-layout component. A layout item groups information together to define visual grids, spacing, and sections.
With default attribute values of size and flexibility, layout items take the size of their content and don't occupy entire width of the container.
We can use below attributes of layout-item to show content based on device size.

- large-device-size
- medium-device-size
- small-device-size With this explanation,the correct answers are:

"<lightning-layout></lightning-layout>" and

"<lightning-layout-item></lightning-layout-item>"

For Code & Reference:
https://developer.salesforce.com/docs/component-library/bundle/lightning-layout-item/specification
https://developer.salesforce.com/docs/component-library/bundle/lightning-layout/example

---

Q.28. A developer wants to write a generic Apex method that will compare the Salesforce Name field between any two object records. For example, to compare the Name field of an Account and an Opportunity; or the Name of an Account and a Contact.
Assuming the Name field exists, how should the developer do this?

Answered Right

Ask Instructor

1. Use a *String.replace()* method to parse the contents of each Name field and then compare the results.
2. Use the Salesforce Metadata API to extract the value of each object and compare the Name fields.
3. Invoke a *Schema.describe()* function to compare the values of each Name field.
4. Cast each object into an sObject and use *sObject.get('Name')* to compare the Name fields. (Correct Answer)(Selected Answer)

This is a scenario-based question related to field value comparison. The scenario given in this question is:

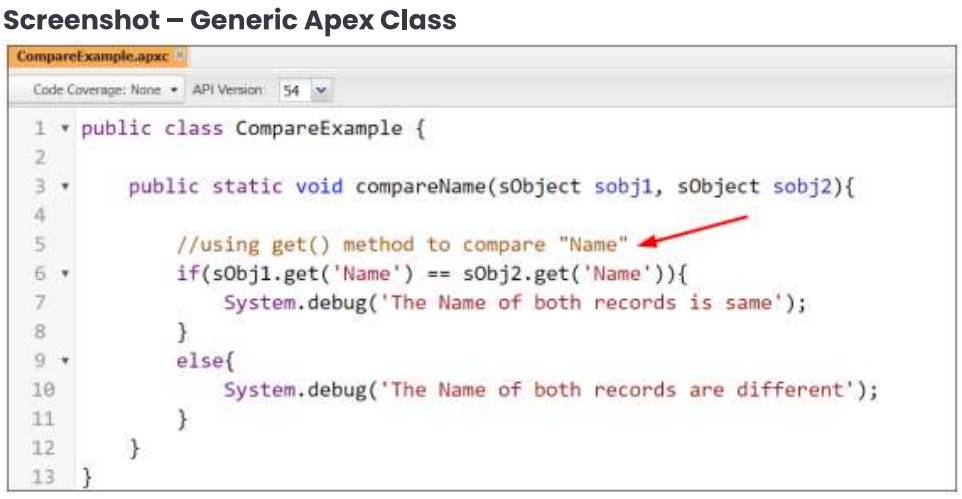
. There is a need to write generic Apex method that will compare the Salesforce Name field between any two object records.

. For example, to compare the Name field of an Account and Contact record.
For this requirement, we need to write a generic apex method therefore, we need to use sObjects.

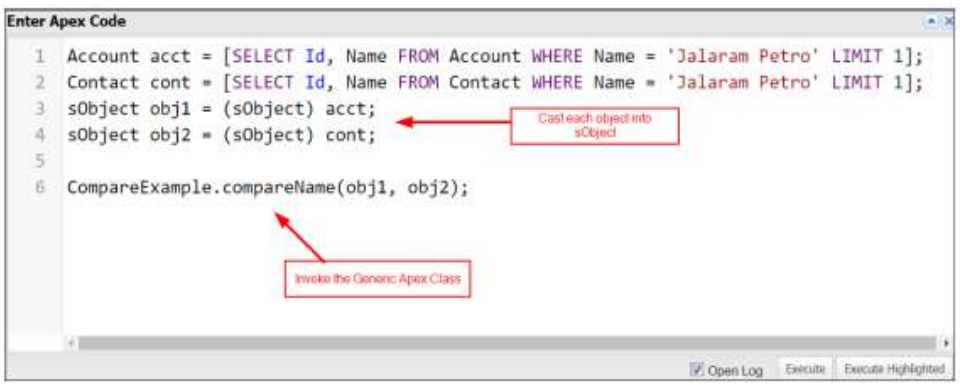
Create Notes

Use sObjects
Because Apex is tightly integrated with the database, you can access Salesforce
records and their fields directly from Apex. Every record in Salesforce is natively represented as an **sObject** in Apex.
We can cast generic sObject to specific sObject and vice versa.
The fields of a generic sObject can be accessed only through the put() and get() methods.
Let's write a Generic Apex Class as asked in this question

Screenshot – Generic Apex Class



Screenshot – Invoking Apex Class



With this explanation,

the correct answer is:
"Cast each object into an sObject and use sObject.get('Name') to compare the Name fields."

Reference:
https://trailhead.salesforce.com/en/content/learn/modules/apex_database/apex_database_sobjects

---

Q.29. Which three approaches should a developer implement to obtain the best performance for data retrieval when building a Lightning web component?
Choose 3 answers.

Answered Wrong

Ask Instructor

1. Use *getRecordUi* to obtain metadata. (Selected Answer)
2. Use the Lightning Data Service. (Correct Answer)
3. Use lazy load for occasionally accessed data. (Correct Answer)(Selected Answer)
4. Use *layoutTypes: ['Full']* to display a set of fields.
5. Use *(Cacheable=true)* whenever possible. (Correct Answer)(Selected Answer)

Create Notes

This question is related to following best practice while writing Lightning Web Components.

Let's learn about Lighting Web Component and how the best performance can be achieved.
Lightning web components run on the client-side, in a single page, where they are created and destroyed as needed, alongside other components that work on the same data.
In this question, the ask is to use best practice while **Data Retrieval** which means while accessing data from Salesforce, what are the best practices we can follow.
With Lightning Web Components, you have various options available to you when retrieving data from the server. To optimize all of your server round-trips, keep these things in mind:

- Use the **Lightning Data Service** or cache data whenever possible
- If you can't use Lightning Data Service, use Apex. A Cacheable Apex method is a server action whose response is stored in the client cache so that subsequent
requests for the same server method with the same set of arguments can be accessed from that cache instead of the server. Creating a Cacheable Apex method is easy. Simply annotate your Apex methods with
@AuraEnabled(cacheable=true)
- When making a call to the server, **limit the fields** and rows of the result set

- **Lazy load** occasionally accessed data. Don't preload data that the user may never need. For example, put least accessed components in a secondary tab
that the user may not click.

the correct answers are:
With this explanation,**"Use the Lightning Data Service."**

and
"Use (Cacheable=true) whenever possible."

and
"Use lazy load for occasionally accessed data."

For Reference & more Details:
https://developer.salesforce.com/blogs/2020/06/lightning-web-components-performance-best-practices#data_retrieval

---

**Q.30. There are user complaints about slow render times of a custom data table within a Visualforce page that loads thousands of Account records at once.
What can a developer do to help alleviate such issues?**

Answered Right

Ask Instructor

1. Use JavaScript remoting to query the accounts.
2. Use the standard Account List controller and implement pagination. (Correct Answer)(Selected Answer)
3. Upload a third-party data table library as a static resource.
4. Use the transient keyword in the Apex code when querying the Account records.

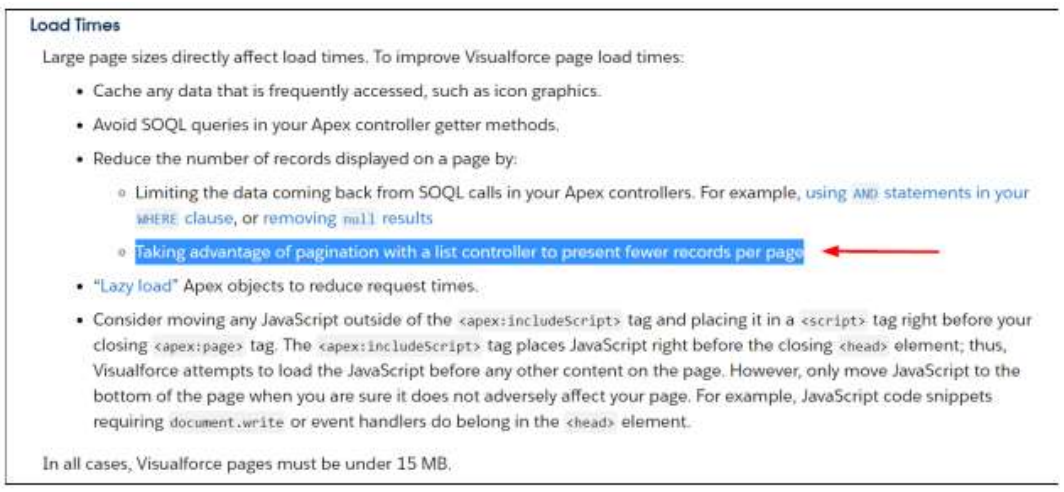
This is a scenario-based question related to performance of Visualforce page.
The scenario given in this question is:

- There is a Visualforce page that uses custom data table to show Account records.
- The Visualforce page loads thousands of Account records at once.

The ask is:

There are user complaints about slow render times of this Visualforce page.
How the performance can be improved.
As per Salesforce documentation, large page size directly affects load times.
PFB:

Create Notes



With this explanation,

the correct answer is:
"Use the standard Account List controller and implement pagination."
because as per the question, the Visualforce page loads thousands of Account records so, pagination with List Controllers will improve performance.
Reference:
https://developer.salesforce.com/docs/atlas.en-us.pages.meta/pages/pages_best_practices_performance.htm

---

Q.31. Which interface needs to be implemented by an Aura component so that it may be displayed in modal dialog by clicking a button on a Lightning record page?

Answered Right

Ask Instructor

1. *lightning:quickAction*
2. *force:lightningEditAction*
3. lightning:editAction
4. *force:lightningQuickAction* (Correct Answer)(Selected Answer)

This question is related to accessing Aura Component as specific places. The scenario given in this question is:

- There is need to create an Aura Component
- This Aura Component should be displayed in modal dialog if we click a button on a Lightning Record Page.

This requirement is related to implementing interfaces in Aura Component. When we try to create an Aura Component we will be given the option to include interfaces, by adding interfaces we are trying to make the components accessible at different places in Lightning Experience.
There are many interfaces we can use in Aura Component depending on the requirement. As per the requirement given in this question, we will need a Quick Action, on click of which, we want to show the Aura Component as modal dialog.

Lightning Quick Action:
Add the force:lightningQuickAction interface to a Lightning component to allow it to be used as a custom action in Lightning Experience or the Salesforce mobile app. You simply add the interface name to the component's implements attribute.

Screenshot – Implementing Interface in Aura Component

```
<!--quickAdd.cmp-->
<aura:component implements="force:lightningQuickAction">

    <!-- Very simple addition -->

    <lightning:input type="number" name="myNumber" aura:id="num1" label="Number 1"/>
    <lightning:input type="number" name="myNumber" aura:id="num2" label="Number 2"/>

    <br/>
    <lightning:button label="Add" onclick="{!c.clickAdd}"/>

</aura:component>
```

Create Notes

the correct answer is:
With this explanation and code example,**"force:lightningQuickAction"**

Reference:
https://developer.salesforce.com/docs/component-library/bundle/force:lightningQuickAction/documentation

---

Q.32. A developer is asked to develop a new AppExchange application. A feature of the program creates Survey records when a Case reaches certain stage and is of a certain Record Type. This feature needs to be configurable, as different Salesforce instances require Surveys different times. Additionally, the out-of-the-box AppExchange app needs to come with a set of best practice settings that apply to more customers.
What should the developer use to store and package the custom configuration settings for the app?

Answered Right

Ask Instructor

1. Custom Settings.
2. Custom Metadata. (Correct Answer)(Selected Answer)
3. Custom Objects.
4. Process Builder.

This is a scenario-based question which is related to configuration in Salesforce Platform. The scenario/requirement given in this question is:

- There is a need to develop a new AppExchange application.
- There is a feature to create Survey records when a Case reaches certain stage and is of certain Record Type.
- The important requirement here is that the feature needs to be configurable, as different Salesforce instances require Surveys different times.

The ask is:

- What should the developer use to store and package the custom configuration settings for the app.

Let's look at few configuration mechanisms used in Salesforce Platform:

1. **Custom Settings**

Custom settings are similar to custom objects. Application developers can create custom sets of data and associate custom data for an organization, profile, or specific user. Only custom settings definitions are included in packages, not data.

2. **Custom Metadata**

Custom metadata is customizable, deployable, packageable, and upgradeable application metadata. First, you create a custom metadata type, which defines the form of the application metadata. Then you build reusable functionality that determines the behavior based on metadata of that type.

Please Note: Main difference between custom metadata and custom setting is that custom metadata records are deployable and packagable. But we cannot deploy custom setting data.
As per the requirement given in the question it says to store and package the custom configuration settings.

Therefore,
"Custom Metadata" is correct because Custom Metadata are deployable and packagable.
References:
https://help.salesforce.com/s/articleView?id=sf.custommetadatatypes_about.htm&type=5
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_customsettings.htm

---

Q.33. What are three reasons that a developer should write Jest-tests for Lightning web components?
Choose 3 answers.

Answered Wrong

Ask Instructor

1. To test basic user interaction. (Correct Answer)

Create Notes

2. To verify that events fire when expected. (Correct Answer)(Selected Answer)

3. To test how multiple components work together. (Selected Answer)

4. To verify the DOM output of a component. (Correct Answer)(Selected Answer)

5. To test a component's non-public properties.

This question is related to Test Lightning Web Components. Jest is a powerful tool with rich features for writing JavaScript tests. Use Jest to write unit tests for all of your Lightning web components.

As per Salesforce Documentation, we can write Jest tests to:

- Test a component in isolation
- Test a component's public API (@api properties and methods, events)
- Test basic user interaction (clicks)
- Verify the DOM output of a component
- Verify that events fire when expected

With this use cases and explanation, the correct answers are:
"To test basic user interaction." because we write Jest Test to test basic user interaction.

and
"To verify that events fire when expected." because we write Jest Test to test to Verify that events fire when expected.

and
"To verify the DOM output of a component." because we write Jest Test to test to verify the DOM output of a component.

Reference:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/lwc.testing

---

Q.34. A developer wants to integrate invoice and invoice line data into Salesforce from a custom billing system. The developer decides to real-time callouts from the building system using the SOAP API. Unfortunately, the developer is getting a lot of errors when inserting invoice line data because the invoice header record does not exist yet.
What will help ensure the transactional integrity of the integration?

Answered Right

Ask Instructor

1. Develop a custom Apex web service to handle a custom JSON data structure with both invoice header and related invoice line.

2. Use an ETL tool and the BULK API running nightly, thus ensuring all of the data is handled at the same time.

3. Set the AllOrNoneHeader to true when calling each of create() for invoice headers and create() for invoice lines.

4. Create the invoice header and the related invoice lines in the same create() call leveraging External Ids. (Correct Answer)(Selected Answer)

A newly added scenario and an explanation will be provided soon.

Q.35. Which tag should a developer use to display different text while an *<apex:commandButton>* is processing an action?

Answered Wrong

Ask Instructor

1. *<apex:actionPoller>*
2. *<apex:actionSupport>* (Selected Answer)
3. *<apex:pageMessages>*
4. *<apex:actionStatus>* (Correct Answer)

Create Notes

This is a scenario-based question related to Visualforce pages in Salesforce.
The scenario given in this question is:

- There is a Visualforce page having command button.
- There is a need to show different text while an is processing an action.

The ask is:
Which tag should a developer use to achieve this.

Let's learn about `apex:actionStatus`
A component that displays the status of an AJAX update request. An AJAX request can either be in progress or complete.
Depending upon the AJAX request status (whether AJAX request is in progress or complete), this component will display different message to user.
In many scenarios AJAX request takes some time. So we should display some message to user that your request is in progress. Once request is complete, we can display some different message to user.

With this explanation,
`<apex:actionStatus>` is correct because it is used to display different message to user.
For Reference & Code:
https://developer.salesforce.com/docs/atlas.en-us.pages.meta/pages/pages_compref_actionStatus.htm

---

Q.36. How should a developer assert that a trigger with an asynchronous process has successfully run?

Answered Wrong

Ask Instructor

1. Create all test data in the test class, invoke *Test.startTest()* and *Test.stopTest()* and then perform assertions. (Correct Answer)
2. Create all test data in the test class, use *System.runAs()* to invoke the trigger, then perform assertions.
3. Insert records into Salesforce, use *seeAllData=true*, then perform assertions.
4. Create all test data, use *@future* in the test class, then perform assertions. (Selected Answer)

This question is related to writing apex test class for Asynchronous Apex. Let's learn some basics about writing Test Class.
Testing is an important part of the development process. Before you can deploy Apex or package it for the Salesforce AppExchange, the following must be true.

- Unit tests must cover at least 75% of your Apex code, and all of those tests must complete successfully.
- Every trigger must have some test coverage.
- All classes and triggers must compile successfully.

Please Note:

The ask in this question is to write Apex Test for Asynchronous Apex. A unit test forms a single transaction, and asynchronous code enqueued within that transaction cannot be executed until the transaction commits successfully.
For this reason, Salesforce has provided a framework to force asynchronous code to execute synchronously for testing: We enclose our test code between **Test.startTest()** and **Test.stopTest()** methods.
The system collects all asynchronous calls made after startTest(). When stopTest() is executed, these collected asynchronous processes are then run synchronously and complete before control returns to our code.
This way developer asserts that a trigger with an asynchronous process has successfully run.

With this explanation, the correct answer is:
"Create all test data in the test class, invoke `Test.startTest()` and `Test.stopTest()` and then perform assertions."

Reference:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_testing_tools_start_stop_test.htm

Create Notes

Q.37. What is a benefit of using a WSDL with Apex?

Answered Right

Ask Instructor

1. Reduces the number of callouts to third-party web services.
2. Allows for classes to be generated from WSDL and imported into Salesforce. (Correct Answer)(Selected Answer)
3. Enables the user to not pass a Session ID where it is not necessary.
4. Allows for web services to be tested and achieve code coverage.

This question is related to SOAP services in Apex. The ask in this question is:

- What is a benefit of using a WSDL with Apex?

As per the Salesforce Document, we can define a Class from WSDL document. Classes can be automatically generated from a WSDL document that is stored on a local hard drive or network. Creating a class by consuming a WSDL document allows developers to make callouts to the external Web service in their Apex code.
The successfully generated Apex classes include stub and type classes for calling the third-party Web service represented by the WSDL document. These classes allow you to call the external Web service from Apex. For each generated class, a second class is created with the same name and with a prefix of Async.

- The first class is for synchronous callouts.
- The second class is for asynchronous callouts.

Steps to generate an Apex class from a WSDL:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_callouts_wsdl2apex.htm

With this explanation, the correct answer of this question is:
"Allows for classes to be generated from WSDL and imported into Salesforce." because we can define a Class from WSDL document.

Q.38. The Account edit button must be overridden in an org where a subset of users still use Salesforce Classic. The org already has a Lightning Component that will do the work necessary for the override, and the client wants to be able to reuse it.
How should a developer implement this?

Answered Right

Ask Instructor

1. Override the edit button for Lightning with a Lightning Page; and for Classic, override the edit button with a Visualforce page that contains the Lightning Component.
2. Override the edit button for both Lightning and Classic with a new Visualforce page.
3. Override the edit button for both Lightning and Classic with a Lightning Component.
4. Override the edit button for Lightning with a Lightning Component; and for Classic, override the edit button with a Visualforce page that contains the Lightning Component. (Correct Answer)(Selected Answer)

Create Notes

This is a scenario-based question which is related to overriding standard button on Account. The scenario given in this question is:

- There is a need to override the Account Edit button.
- There are few users who still use Salesforce Classic.
- The org already has a Lightning Component that will do the work necessary for the override.

The requirement is:

- The client wants to **reuse this Lightning Component.** That means, the Account Edit button should be overridden for both classic and lightning with making sure that the Lightning Component is reused.

Let's check what option we get when we want to override the Account "Edit" button.

Screenshot – Override Account Edit button

Therefore, the solution approach we can use is:

- Override the edit button for Lightning with a Lightning Component
- For Classic, override the edit button with a Visualforce page that contains the Lightning Component. We can use Lightning Component inside Visualforce page.

With this explanation, the correct answer is:

"Override the edit button for Lightning with a Lightning Component; and for Classic, override the edit button with a Visualforce page that contains the Lightning Component."

For Reference: (Using Lightning Component inside Visualforce page)
https://developer.salesforce.com/docs/atlas.en-us.236.0.lightning.meta/lightning/components_visualforce.htm

---

Q.39. Consider the controller code below that is called from an Aura component and returns data wrapped in a class.

```
public class myServerSideController {
    @AuraEnabled
    public static MyDataWrapper getSomeData( String theType ) {
        Some_Object__c someObj = [
                SELECT ID, Name
                FROM Some_Object__c
                WHERE Type__c = :theType
                LIMIT 1
        ];

        Another_Object__c anotherObj = [
                SELECT ID, Option__c
                FROM Another_Object__c
                WHERE Some_Object__c = :someObj.Name
                LIMIT 1
        ];

        MyDataWrapper theData = new MyDataWrapper();

        theData.Name = someObj.Name;
        theData.Option = anotherObj.Option__c;
        return theData;
    }

    public class MyDataWrapper {
        public String Name { get; set; }
        Public String Option {get; set;}
        Public MyDataWrapper() {}
    }
}
```

The developer verified that the queries return a single record each and there is error handling in the Aura component, but the component is not getting anything back when calling the controller getSomeData().

Create Notes

What is wrong?

Answered Right

Ask Instructor

1. The member's Name and Option of the class *MyDataWrapper* should be annotated with *@AuraEnabled* also. (Correct Answer)(Selected Answer)
2. Instances of Apex classes, such as *MyDataWrapper*, cannot be returned to a Lightning component.
3. The member's *Name* and *Option* should not have getter and setter.
4. The member's *Name* and *Option* should not be declared public.

This is scenario-based question which is related to getting Salesforce data in lightning aura component. The scenario given in this question is:

- There is a wrapper class named "MyDataWrapper"
- There is an apex method named "getSomeData ()" which sets the variables "Name" and "Option" of wrapper class and return instance of wrapper class.

The issue faced is:

- The queries return a single record each but the component is not getting anything back when calling the apex method "getSomeData ()".

Let's understand about @AuraEnabled annotation

The AuraEnabled annotation enables Lightning components to access Apex methods and properties. The AuraEnabled annotation is overloaded, and is used for two separate and distinct purposes.

- Use @AuraEnabled on Apex class static methods to make them accessible as remote controller actions in your Lightning components.
- Use @AuraEnabled on Apex instance methods and properties to make them serializable when an instance of the class is returned as data from a server-side action.

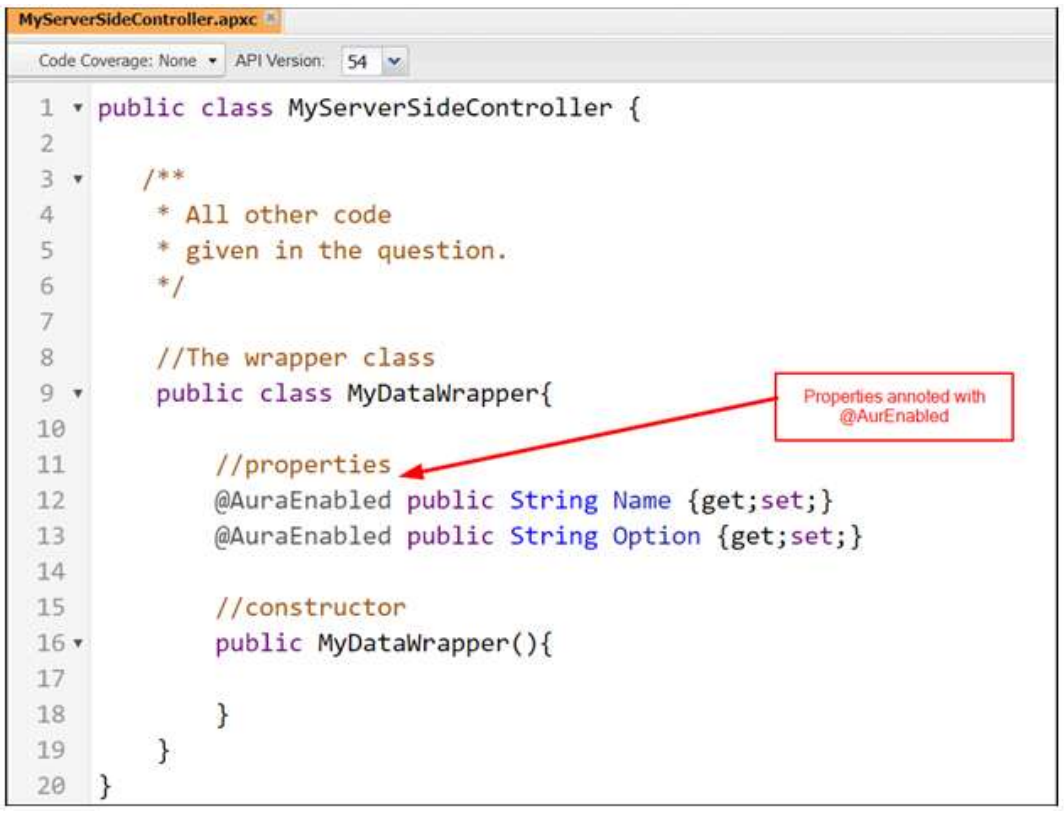
Please Note:
When an instance of an Apex class is returned from a server-side action, the instance is serialized to JSON by the framework. Only the values of public instance properties and methods annotated with @AuraEnabled are serialized and returned.
For example, here's a simple "MyDataWrapper" Apex class that contains Name and Option properties. When returned from a remote Apex controller action, because it doesn't have the @AuraEnabled annotation for Name and Option property, these properties aren't serialized on the server side, and isn't returned as part of the result data.

With this explanation,

the correct answer is:
"The member's Name and Option of the class MyDataWrapper should be annotated with @AuraEnabled also" is correct.

Screenshot – Modified Version of code given in the question

Create Notes



References:

https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/controllers_server_apex_returning_data.htm
https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/controllers_server_apex_auraenabled_annotation.htm

---

Q.40. Given the following containment hierarchy:

```
<!—myParentComponent.html -->
<template>
    <c-my-child-component></c-my-child-component>
</template>
```

What is the correct way to communicate the new rule of a property named "passthrough" to my-parent-component if the property defined within my-child-component?

Answered Right

Ask Instructor

1. `let cEvent = new CustomEvent('passthrough', { detail: this.passthrough });`
`this.dispatchEvent(cEvent);`

(Correct Answer)(Selected Answer)

2. `let cEvent = new customEvent('passthrough', { detail: 'this.passthrough' });`
`this.dispatchEvent(cEvent);`

3. `let cEvent = new CustomEvent($passthrough);`
`this.dispatchEvent(cEvent);`

4. `let cEvent = new CustomEvent('passthrough');`
`this.dispatchEvent(cEvent);`

Create Notes

This is a scenario-based question related to communication between lightning web components. The scenario given in this question is:

- There is a parent lightning web component named "myParentComponent"
- There is a child lightning web component named "myChildComponent"
- There is a property named "passthrough" in child lightning web component which needs to be passed to parent component.

The ask is:
What is the correct way to communicate the new rule of a property named "passthrough" to my-parent-component if the property defined within my-child-component.

Let's learn few concepts:
There are typically **3 approaches** for communication between the components using events.

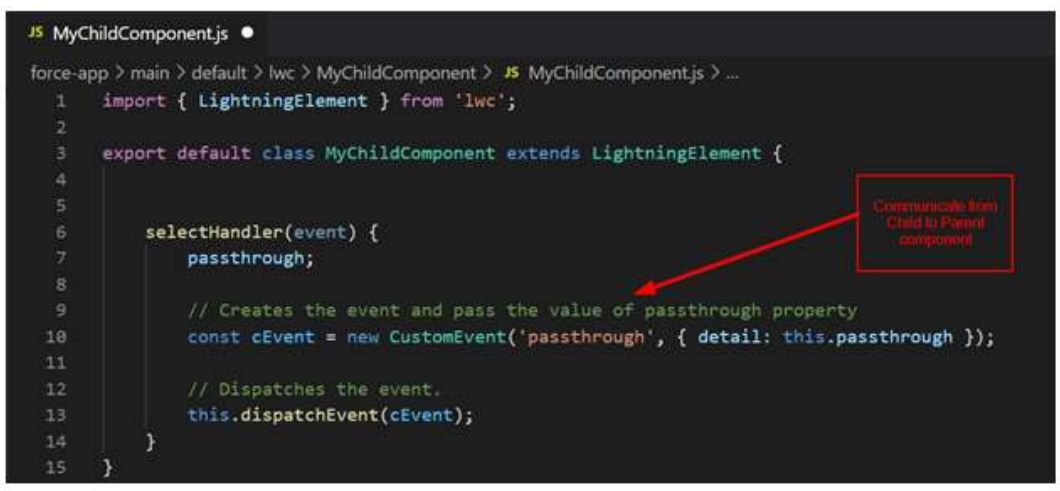
1. Communication using Method in LWC **(Parent to Child).**
2. Custom Event Communication in Lightning Web Component **(Child to Parent).**
3. Publisher- Subscriber model in Lightning Web Component **(Two components which doesn't have a direct relation).**

As per the scenario given in the question, we need to use **"Custom Event Communication in Lightning Web Component"** because we need to pass data from Child to Parent component.

Using CustomEvent:

- To create an event, use the CustomEvent () constructor
- To dispatch an event, call the EventTarget.dispatchEvent() method.
- The CustomEvent () constructor has one required parameter, which is a string indicating the event type.
- Don't prefix your event name with the string on, because inline event handler names must start with the string on.
- To pass data up to a receiving component, set a detail property in the CustomEvent constructor. Receiving components access the data in the **detail property** in the event listener's handler function.

Screenshot – Sample Code to communicate from Child to Parent using CustomEvent



With this explanation, the correct answer is

```
let cEvent = new CustomEvent('passthrough', { detail: this.passthrough });
this.dispatchEvent(cEvent);
```

Reference:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/lwc.events_create_dispatch

Create Notes

Q.41.

```
Line 1: @isTest
Line 2: static void testMyTrigger()
Line 3: {
Line 4:     //Do a bunch of data setup
Line 5:     DataFactory.setupDataForMyTriggerTest();
Line 6:
Line 7:     List<Account> acctsBefore = [SELECT Is_Customer__c FROM Account WHERE Id IN
:DataFactory.accounts];
Line 8:
Line 9:     //Utility to assert all accounts are not customers before the update
Line 10:    AssertUtil.assertNotCustomers(acctsBefore);
Line 11:
Line 12:    //Set accounts to be customers
Line 13:    for(Account a : DataFactory.accounts)
Line 14:    {
Line 15:        a.Is_Customer__c = true;
Line 16:    }
Line 17:
Line 18:    update DataFactory.accounts;
Line 19:
Line 20:    List<Account> acctsAfter = [SELECT Number_Of_Transfers__c FROM Account WHERE Id IN
:DataFactory.accounts];
Line 21:
Line 22:    //Utility to assert Number_Of_Transfers__c is correct based on test data
Line 23:    AssertUtil.assertNumberOfTransfers(acctsAfter);
Line 24: }
```

The test method above tests an Apex trigger that the developer knows will make a lot of queries when a lot of Accounts are simultaneously updated to be customers. The test method fails at the Line 20 because of too many SOQL queries.
What is the correct way to fix this?

Answered Right

Ask Instructor

1. Replace most of the Apex Trigger with Process Builder processes to reduce the number of queries in the trigger.
2. Add Test.startTest() before Line 18 of the code and add Test.stopTest() after line 18 of the code.
(Correct Answer)(Selected Answer)
3. Change the DataFacory class to create fewer Accounts so that the number of queries in the trigger is reduced.
4. Add Test.startTest() before and Test.stopTest() after both Line 7 of the code and Line 20 of the code.

A newly added scenario and an explanation will be given earlier.

Q.42. The Salesforce admin at Cloud Kicks created a custom object called Region_c to store all postal zip codes in the United States and the Cloud Kicks sales region the zip code belongs to.

Object Name:
Region__c
Fields:
Zip_Code__c (Text)
Region_Name__c (Text)

Cloud Kicks wants a trigger on the Lead to populate the Region based on the Lead's zip code.
Which code segment is the most efficient way to fulfill this request?

Answered Wrong

Ask Instructor

Create Notes

1.

```
Set<String> zips = new Set(String)();
for (Lead 1 : Trigger.new) {
        If(1 : PostalCode != Null) {
              zips.add(1.PostalCode);
        }
}

List<Region_c> regions = [SELECT Zip_Code_c, Region_Name_c FROM Region_c WHERE
Zip_Code_c IN :zips];

Map<String, String> zipMap = new Map<String, String>{};
for(Region_c r : regions) {
       zipMap.put(r.Zip_Code_c, r.Region_Name_c);
}

for(Lead 1 : Trigger.new) {
      if(1.PostalCode != Null) {
            1.Region_c = zipMap.get(1.PostalCode);
      }
}
```

(Correct Answer)

2.

```
Set<String> zips = new Set(String)();
for (Lead 1 : Trigger.new) {
        If(1 : PostalCode != Null) {
              zips.add(1.PostalCode);
        }
}

for(Lead 1 : Trigger.new) {
      List<Region_c> regions = [SELECT Zip_Code_c, Region_Name_c FROM Region_c WHERE
      Zip_Code_c IN :zips];
      for(Region_c r :regions) {
      }
}
```

3.

```
Set<String zips> = new Set<String>();
for(Lead 1 : Trigger.new) {
      if(1.PostalCode != Null) {
            zips.add(1.PostalCode);
      }
}

List<Region_c> regions = [SELECT Zip_Code_c, Region_Name_c FROM Region_c WHERE
Zip_Code_c IN : zips];

Map<String, String> zipMap = new Map<String, String>();
for(Region_c r : regions) {
     zipMap.put (r.Zip_Code_c, r.Region_Name_c);
}

for(Lead 1 : Trigger.new) {
      if(1.PostalCode != Null) {
            1.Region_c = zipMap.get(1.PostalCode);
      }
}
```

(Selected Answer)

4.

```
for(Lead 1 : Trigger.new) {
      Region_c reg = [SELECT Region_Name_c FROM Region_c WHERE Zip_Code_c =
Zip_Code_c IN :zips];

for(Lead 1 : Trigger.new) {
      for(Region_c r :regions) {
            if(1.PostalCode == r.Zip_Code_ c) {
                  1.Region_c = r.Region_Name_c;
            }
        }
    }
}
```

Create Notes

This is a scenario-based question. The scenario given in this question is:

- There is a custom object called Region_c to store all postal zip codes in the United States.
- **Object Name:** Region__c
- **Fields:**
  - Zip_Code__c (Text)
  - Region_Name__c (Text)

The requirement is:

- There is a trigger on the Lead to populate the Region based on the Lead's zip code. Which code segment is the most efficient way to fulfill this request?

This question is related to Apex Trigger best practice. As per the Apex Trigger best practices:

- Avoid SOQL Queries or DML statements inside FOR Loops

Let's check each option and find the appropriate one.
Below Option is **not correct** because it used SOQL query inside for loop.

```
Set<String> zips = new Set(String)();
for (Lead 1 : Trigger.new) {
      If(1 : PostalCode != Null) {
            zips.add(1.PostalCode);
      }
}

for(Lead 1 : Trigger.new) {
      List<Region_c> regions = [SELECT Zip_Code_c, Region_Name_c FROM Region_c WHERE
      Zip_Code_c IN :zips];
      for(Region_c r :regions) {
      }
}
```

Create Notes

Below option is **not correct** because it uses SOQL query inside for loop

```
for(Lead 1 : Trigger.new) {
    Region_c reg = [SELECT Region_Name_c FROM Region_c WHERE Zip_Code_c =
Zip_Code_c IN :zips];

for(Lead 1 : Trigger.new) {
    for(Region_c r :regions) {
        if(1.PostalCode == r.Zip_Code_ c) {
            1.Region_c = r.Region_Name_c;
        }
    }
  }
}
```

Below option is **not correct** because it uses Set <String zips> which is invalid.

```
Set<String zips> = new Set<String>();
for(Lead 1 : Trigger.new) {
    if(1.PostalCode != Null) {
        zips.add(1.PostalCode);
    }
}

List<Region_c> regions = [SELECT Zip_Code_c, Region_Name_c FROM Region_c WHERE
Zip_Code_c IN : zips];

Map<String, String> zipMap = new Map<String, String>();
for(Region_c r : regions) {
    zipMap.put (r.Zip_Code_c, r.Region_Name_c);
}

for(Lead 1 : Trigger.new) {
    if(1.PostalCode != Null) {
        1.Region_c = zipMap.get(1.PostalCode);
    }
}
```

Below option is **correct** because it uses correct logic & best practice to query.

```
Set<String> zips = new Set(String)();
for (Lead 1 : Trigger.new) {
    If(1 : PostalCode != Null) {
        zips.add(1.PostalCode);
    }
}

List<Region_c> regions = [SELECT Zip_Code_c, Region_Name_c FROM Region_c WHERE
Zip_Code_c IN :zips];

Map<String, String> zipMap = new Map<String, String>{};
for(Region_c r : regions) {
    zipMap.put(r.Zip_Code_c, r.Region_Name_c);
}

for(Lead 1 : Trigger.new) {
    if(1.PostalCode != Null) {
        1.Region_c = zipMap.get(1.PostalCode);
    }
}
```

Q.43. A developer created the following test method:

```
@isTest (SeeAllData = true)
  public static void testDeleteTrigger(){

    Account testAccount = new Account (name = 'Test1');
    insert testAccount;

    List<Account>testAccounts = [SELECT Id, Name from Account WHERE Name like 'Test%'];
    System.assert(testAccounts.size() > 0);

    delete testAccounts;
    testAccounts = [SELECT Id, Name from Account WHERE Name like 'Test%'];
    System.assert(testAccounts.size() == 0);
}
```

The developer org has five accounts where the name starts with "Test". The developer executes this taste in the Developer Console.
After the test code runs, which statement is true?

Answered Wrong

Create Notes

Ask Instructor

1. The test will fail.
(Selected Answer)

2. There will be no accounts where the name starts with "Test".

3. There will be five accounts where the name starts with "Test".
(Correct Answer)

4. There will be six accounts where the name starts with "Test".

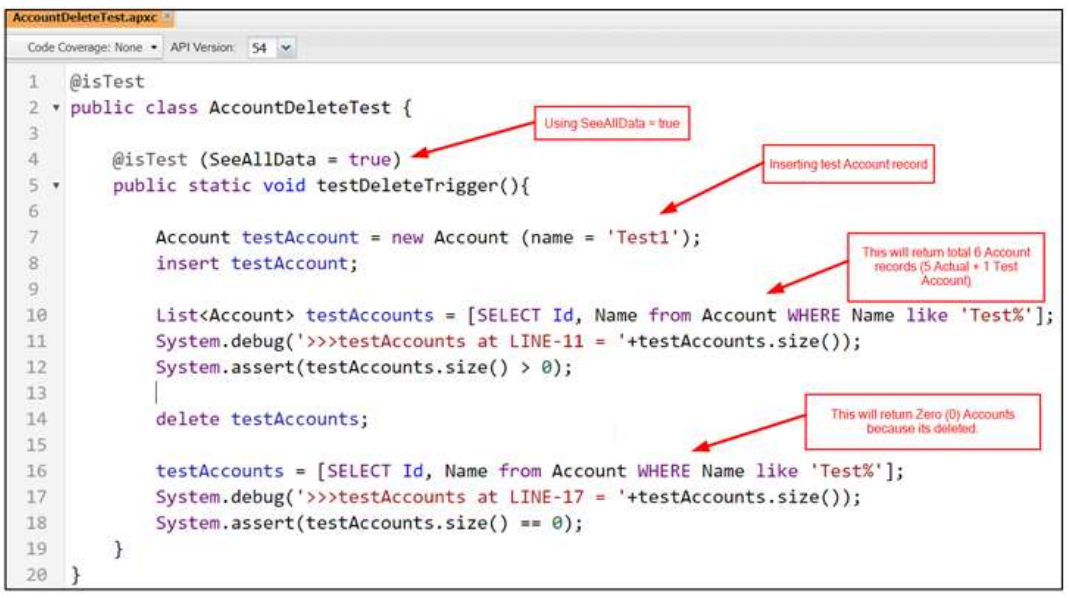
This is scenario-based question related to Test Class in Salesforce. The scenario given in this question is:

- A sample apex test code is given which uses SeeAllData=true.
- The developer org has five accounts where the name starts with "Test".
- The developer executes this test in the Developer Console.

The ask is:

- After the test code runs, which statement is true?

Screenshot – Sample Code given in question:



From the above given code, please find below observations:

1. In System, there are total 5 Account records where the name starts with "Test".
2. This sample code uses SeeAllData=true so it will be able to access those 5 Actual Account Records.
3. The sample code creates 1 Test Account record in LINE-08
4. The query in LINE-10 will return total 6 Account Records. (5 Actual Records + 1 Test Records)
5. The sample code deletes all records
6. The Query at LINE-16 will return zero (0) Account records because its deleted.

Please Note:

Salesforce helps us out by ensuring that any DML operations performed in a unit test are not committed to the database – everything is rolled back at the end of the transaction.

In this scenario, initially we have 5 actual account records. Therefore, at the end of transaction, these 5 account records will remain as is.

With this explanation,

the correct answer is:
"There will be 5 accounts where the name starts with "Test". because once the transaction is completed the deleted records will be rolled back.
Reference:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_testing_seealldata_using.htm

---

Q.44. A developer is tasked with creating a Lightning web component that is responsive on various devices.
Which two components should help accomplish this goal?
Choose 2 answers.

Answered Right

Ask Instructor

1. *lightning-input-location*

Create Notes

2. *lightning-navigation*
3. *lightning-layout*
(Correct Answer)(Selected Answer)
4. *lightning-layout-item* (Correct Answer)(Selected Answer)

This question is related to the concept of lightning web component with grid utility. The requirement is to keep the lightning web component mobile-ready. As per Salesforce documentation, we can use Lightning Layout and Layout Item to display the content based on device size.

- **Layout:** Represents a responsive grid system for arranging containers on a page.
- **Layout Item:** The basic element in a lightning-layout component. A layout item groups information together to define visual grids, spacing, and sections.

With default attribute values of size and flexibility, layout items take the size of their content and don't occupy entire width of the container.

We can **use below attributes** of layout-item to show content based on device size.

- large-device-size
- medium-device-size
- small-device-size

With this explanation, the correct answers are:
"<lightning-layout></lightning-layout>" and

"<lightning-layout-item></lightning-layout-item>"

For Code & Reference:
https://developer.salesforce.com/docs/component-library/bundle/lightning-layout-item/specification
https://developer.salesforce.com/docs/component-library/bundle/lightning-layout/example

---

Q.45. A developer is creating a Lightning web component that contains a child component. The property stage is being passed from the parent to the child. The public property is changing, but the setOppList function is not be invoked.

```
@api stage;
opps;
connectedCallback() {
      this.opps = this.setOppList(this.stage);
   }
}
```

What should the developer change to allow this?

Answered Right

Ask Instructor

1. Move the logic from *connectedCallBack()* to *constructor()*.
2. Create a custom event from the parent component to set the property.
3. Move the logic from *connectedCallBack()* to *renderedCallBack()*.
4. Move the logic to a getter/setter pair. (Correct Answer)(Selected Answer)

Create Notes

This is a scenario-based question which is related to component communication in Lightning Web Component. The scenario given in this question is:

- There is a parent component and a child component.
- The parent component passes the stage name to child component.

The Issue:

- The public property is changing, but the setOppList function is not be invoked.

In short, this question is related to when parent component passes a value to child component and when the value changes, how to perform custom logic in child component.
As per the Salesforce documentation,

- To execute logic each time a public property is set, **write a custom setter.**
- If you write a setter for a public property, you must also write a getter. Annotate either the getter or the setter with @api, but not both. It's a best practice to annotate the getter.
- To hold the property value inside the getter and setter, use a field.

The example given in this question is:

- The property stage is being passed from the parent to the child.
- When the stage changes, the logic need to be executed to get opportunities.

Let's recreate this scenario:

- Created a child component named "showOppChild"
- Created a parent component named "showOppParent"
- The parent component "showOppParent" has the dropdown having stageName of opportunities. When user change the dropdown value, the stageName is passed to child component and it executes a custom logic using setters/ getters to fetch the opportunities.

Screenshot – Child Component **showOppChild.html**

showOppChild.html

force-app > main > default > lwc > showOppChild > showOppChild.html > ...

```
<template>
    <lightning-card title="Show Opportunity" icon-name="custom:custom57" class="slds-p-around_medium">
        <div class="slds-p-around_medium">

            <!--heading-->
            <h2 class="slds-p-bottom_medium message-text">{message}</h2>

            <!--check if there is atleast one opportunity-->
            <template if:true={opps}>

                <!-- data table -->
                <template if:true={oppFound}>
                    <lightning-datatable key-field="Id" data={opps} columns={columns} hide-checkbox-column>
                    </lightning-datatable>
                </template>

            </template>
        </div>
    </lightning-card>
</template>
```

Create Notes

Screenshot – Child Component **– showOppChild.js**

```
import {LightningElement, api} from 'lwc';
import getOpps from '@salesforce/apex/OppControllerApex.getOppByStage';

const columns = [
    { label: 'Id', fieldName: 'Id' },
    { label: 'Name', fieldName: 'Name', type: 'Text' },
    { label: 'Stage', fieldName: 'StageName', type: 'picklist' }
];

export default class ShowOppChild extends LightningElement {
    _providedValue;
    opps;
    columns = columns;
    message;
    oppFound = false;

    @api
    get providedValue(){
        return this._providedValue;
    }

    /**
     * when the drop-down change happen in parent component,
     * the value gets passed in child component,
     * the setter gets executed and we can write logic here
     */
    set providedValue(value){
        this._providedValue = value;

        if(this._providedValue === 'Select'){
            this.message = 'Please Select Opportunity Stage from dropdown to see Opportunities';
        }else{
            this.message = `Please find below Opportunities for ${this._providedValue} Stage`;
        }

        //call the apex method to get list of opportunity based on opportunity Stage
        getOpps({ selectedStage: this._providedValue })
        .then(result => {
            this.opps = result;
            this.error = undefined;
            if(this.opps.length > 0){
                this.oppFound = true;
            }
        })
        .catch(error => {
            this.error = error;
            this.opps = undefined;
        })
    }
}
```

To execute a logic each time a public property is set, write a custom setter.

Screenshot–Parent Component **– showOppParent.html**

```
<template>
    <div class="slds-form-element__control slds-p-bottom_medium">
        <div class="slds-select_container">
            <select class="slds-select"
                    onchange={selectionChangeHandler}>
                <option value="Select">Select</option>
                <template for:each={options} for:item="option">
                    <option key={option.label} value={option.value}>{option.value}</option>
                </template>
            </select>
        </div>
    </div>

    <!--invoking child component by passing selectedOption-->
    <c-show-opp-child provided-value={selectedOption}></c-show-opp-child>
</template>
```

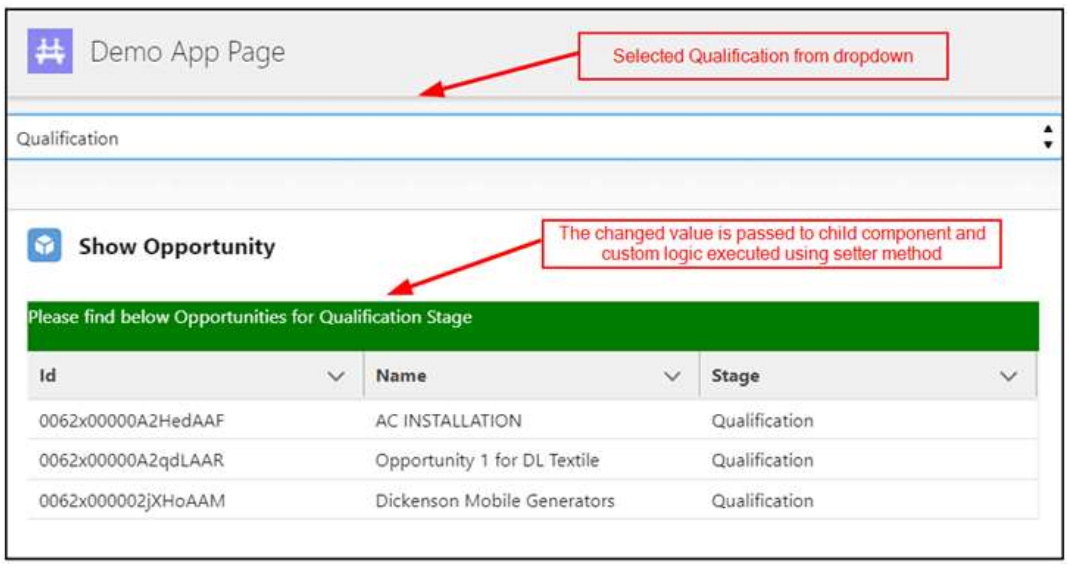
calling child component by passing stageName

Screenshot – Parent Component **– showOppChild.js**

```
import { LightningElement } from 'lwc';

export default class ShowOppParent extends LightningElement {

    selectedOption = 'Select';
    options = [
        {label : "Prospecting", value : "Prospecting"},
        {label : "Qualification", value : "Qualification"},
        {label : "Closed Won", value : "Closed Won"},
        {label : "Closed Lost", value : "Closed Lost"}
    ];

    //when the value change from dropdown, update the value of selectedOption property
    selectionChangeHandler(event){
        this.selectedOption = event.target.value;
    }
}
```

Screenshot – Output when we select “Qualification” as Opportunity Stage from drop down

Create Notes



With this explanation,

the correct answer is:
"Move the logic to a getter/setter pair." because to execute logic each time a public property is set, write a custom setter. If you write a setter for a public property, you must also write a getter.

Reference:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/js_props_getters_setters

---

Q.46. A developer creates a Lightning web component to allow a Contact to be quickly entered. However, error messages are not displayed.

```
<template>
    <lightning-record-edit-form
            object-api-name="Contact">
        <lightning-input-field field-name="FirstName"></lightning-input-field>
        <lightning-input-field field-name="LastName"></lightning-input-field>
        <lightning-input-field field-name="Email"></lightning-input-field>
        <lightning-button type="submit"
                          name="submit"
                          label ="Create Contact">
            </lightning-button>
    </lightning-record-edit-form>
</template>
```

Which component should the developer add to the form to display error messages?

Answered Wrong

Ask Instructor

1. *apex:messages*
2. *aura:messages*
3. *lightning-messages* (Correct Answer)
4. *lightning-error* (Selected Answer)

Create Notes

This question is related to Record Edit Form in Lightning Web Component. The scenario given in this question is:

- There is a lightning web component used to enter a new contact.
- This lightning web component uses lightning-record-edit-form.

The issue:

- The user is not able to see error messages if something is wrong.

The ask is:

- Which component should the developer add to the form to display error messages?

Record Edit Form
Use the lightning-record-edit-form component to create a form that's used to add a Salesforce record or update fields in an existing record on an object. The component displays fields with their labels and the current values, and enables you to edit their values.
Please Note:

When we use record edit form, to display the error message automatically, include **lightning-messages** immediately before or after the lightning-input-field components.

Screenshot – Sample code using lightning-messages to show Error

```
<template>
    <lightning-record-edit-form object-api-name="Contact">
        <lightning-messages> </lightning-messages>
        <lightning-input-field field-name="Name"> </lightning-input-field>
        <lightning-button
            class="slds-m-top_small"
            type="submit"
            label="Create new"
        >
        </lightning-button>
    </lightning-record-edit-form>
</template>
```

With this explanation,

the correct answer is:
"lightning-messages." is correct because it is used to display the error message automatically.
Reference:
https://developer.salesforce.com/docs/component-library/bundle/lightning-record-edit-form/documentation

---

Q.47. As part of new integration, a developer is asked to implement a new custom search functionality that is capable of performing unrestricted queries and can account for all values within a custom picklist field, Type__c, on the Opportunity object. The search feature must also account for NULL values.
The organization-wide default for the Opportunity object is set to Public Read-only, and a new custom index has been created for the Type__c field. There are more than 5 million Opportunity records within the environment, and a considerable amount of the existing records have NULL values for the picklist. Which technique should the developer implement to maximize performance when querying NULL values?

Answered Wrong

Ask Instructor

1. Create a formula field at substitutes NULL values for a string of text, create an index for the formula field, then use the formula within the WHERE clause.
2. Perform two SOQL queries; one to query Opportunities where *Type__c != NULL*, and another to query where *Type__c = NULL*, then join the result set using Apex.
(Correct Answer)
3. Use a SOSL query to return All opportunities that have a value of NULL in any field.
(Selected Answer)
4. Use the OR operator to combine WHERE clauses to strictly search for each value within the picklist, including *Type__c = NULL*.

A newly added scenario and an explanation will be given earlier.

Create Notes

**Q.48. A developer wrote an Apex method to update a list of Contacts and wants to make it available for use by Lightning web components.
Which annotation should the developer add to the Apex method to achieve this?**

Answered Right

Ask Instructor

1. *@AuraEnabled*
(Correct Answer)(Selected Answer)
2. *@RemoteAction*
3. *@RemoteAction(cacheable=true)*
4. *@AuraEnabled(cacheable=true)*

This question is related to exposing Apex Methods to Lightning Web Components. The scenario given in this question is:

- There is an apex method to update a list of Contacts.
- There is a need to expose this apex method to Lightning Web Component.

The ask is:

- Which annotation should the developer add to the Apex method to achieve this?

As per Salesforce documentation, to expose an apex method to a lightning web component:

- The method must be static
- The method must be either global or public.
- Annotate the method with @AuraEnabled.

Please Note:

- To improve runtime performance, set `@AuraEnabled(cacheable=true)` to cache the method results on the client.
- To set `cacheable=true`, a method must only get data. It can't mutate data.

With this explanation,

the correct answer is:
"@AuraEnabled." because to expose an apex method to a lightning web component, annotate the method with @AuraEnabled.
Option **"@AuraEnabled(cacheable=true)."** is **not correct** because to set `cacheable=true`, a method must only get data but in the scenario given in this question, the apex method is updating the data so cacheable=true can not be used.
References:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/lwc.apex_expose_method
https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/controllers_server_apex_auraenabled_annotation.htm

Create Notes

Q.49. Refer to the code below:

```
List<Opportunity> opportunities = [SELECT Id, Amount from Opportunity ];
for (Opportunity opp: opportunities) {
    // perform operation here
}
```

**When the code runs, it results in a System Limit Exception with the error message: Apex heap size too large.
What should be done to fix this error?**

Answered Right

Ask Instructor

1. Use a SOQL *for* loop to process the data.
(Correct Answer)(Selected Answer)
2. Convert the *List* into List into a Set.
3. Use *Limits.getLimitHeapSize().*
4. Use a try/catch block to catch the error.

This is a scenario-based question related to an error. The scenario given in this question is:

- A sample apex code is given.
- When the code runs, it results in a System Limit Exception with the error message: Apex heap size too large.

The ask is:

- What should be done to fix this error?

Let's learn about Heap Size in Apex:

- Salesforce enforces an Apex Heap Size Limit of 6MB for synchronous transactions and 12MB for asynchronous transactions.
- The "Apex heap size too large" error occurs when too much data is being stored in memory during processing.

Please Note: From the sample code given in this question, we can observe that the query to fetch the opportunity does not have any WHERE clause and it can fetch lot of records and iterating over those records may through heap size error.

As per Salesforce documentation, best practice to for running within the Apex heap size:

- Don't use class level variables to store a large amount of data.
- **Utilize SOQL For Loops** to iterate and process data from large queries.
- Construct methods and loops that allow variables to go out of scope as soon as they are no longer needed.

With this explanation, the correct answer is:
"Use a SOQL for loop to process the data." because this is one of the best practices to run within the apex heap size.
Reference:
https://help.salesforce.com/s/articleView?id=000321537&type=1

Q.50. A developer created a Lightning web component that allows users to input or text value that is used to search for Accounts by calling an Apex method. The Apex method returns a list of AccountWrappers and is called imperatively from a JavaScript event Handler.

```
01:
02:  public class AccontSearcher {
03:
04:      public static List<AccountWrapper> search(String term) {
05:          List<AccountWrapper>  wrappers = getMatchingAccountWrappers(term);
06:          return wrappers;
07:      }
08:
09:
10:      public class AccountWrapper {
11:          public Account { get; set;}
12:          public Decimal matchProbability { get; set; }
13:      }
14:      // ... other methods, including getMatchingAccountWrappers implementation...
15:  }
```

Which two changes should the developer make so the apex method functions correctly?
Choose 2 answers.

Answered Right

Ask Instructor

1. Add *@AuraEnabled* to line 09.
2. Add *@AuraEnabled* to line 03.
(Correct Answer)(Selected Answer)
3. Add *@AuraEnabled* to lines 11 and 12.
(Correct Answer)(Selected Answer)
4. Add *@AuraEnabled* to line 01.

Create Notes

This question is related to exposing Apex Methods to Lightning Web Components and returning data from Apex Server-Side controller.

The scenario given in this question is:

- There is a Lightning web component that allows users to input or text value that is used to search for Accounts by calling an Apex method.
- The Apex method returns a list of AccountWrappers and is called imperatively from a JavaScript event Handler.
- The apex method is not functioning correctly.

The ask is:
• Which two changes should the developer make so the apex method functions correctly?
As per Salesforce documentation, to expose an apex method to a lightning web component:

- The method must be static
- The method must be either global or public.
- Annotate the method with @AuraEnabled.

Returning Data from an Apex Server-Side Controller

- When an instance of an Apex class is returned from a server-side action, the instance is serialized to JSON by the framework. Only the values of public instance properties and methods annotated with @AuraEnabled are serialized and returned.

Now, coming back to the question, we can observe from the sample code given:

- The "search ()" method is called by Lightning Web Component but this method is not annoted with @AuraEnabled.
  - o We need to use @AuraEnabled for search () method at LINE-03.
- The public properties at LINE-11 and LINE-12 are not annoted with @AuraEnabled
  - o We need to use @AuraEnabled for public properties at LINE-11 and LINE-12.

With this explanation, the correct answers are:
"Add @AuraEnabled to line 03."
and
"Add @AuraEnabled to lines 11 and 12."

References:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/lwc.apex_expose_method
https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/controllers_server_apex_returning_data.htm

Create Notes

Q.51. An org has a Process Builder process on Opportunity that sets a custom field, CommissionBaseAmount__c, when an Opportunity is edited and the Opportunity's Amount changes. A developer recently deployed an Opportunity before update trigger that uses the CommissionBaseAmount__c and complex logic to calculate a value for a custom field, CommissionAmount__c, when an Opportunity stage changes to closed/won.
Users report that when they changed the Opportunity to Closed/Won and also change the Amount during the same save, the CommissionAmount__c is incorrect.
Which two actions should the developer take to correct this problem?
Choose 2 answers.

Answered Right

Ask Instructor

1. Call the trigger from the process.
(Correct Answer)(Selected Answer)
2. Uncheck the recursion checkbox on the process.
3. Use a static Boolean variable in the trigger.
(Correct Answer)(Selected Answer)
4. Call the process from the trigger.

A newly added scenario and an explanation will be given earlier.

Q.52. As part of point-to-point integration, a developer must call and external web service which, due to high demand, takes a long time to provide for response. As part of the request, the developer must collect key inputs from the end user before making the callout.
Which two elements should the developer use to implement these business requirements?
Choose 2 answers.

Answered Wrong

Ask Instructor

1. Apex method that returns a Continuation object. (Correct Answer)
2. Screen Flow. (Selected Answer)
3. Process Builder.
4. Lightning web component. (Correct Answer)(Selected Answer)

This is a scenario-based question related to integration. The scenario given in this question is:

- There is need to call an external web service.
- Due to high demand, the external web service takes a long time to provide for response.
- As part of the request, the developer must collect key inputs from the end user before making the callout.

The ask is:

- Which two elements should the developer use to implement these business requirements?

This can be achieved using Continuations. Let's learn about making Long-Running Callouts with Continuations.
Use the **Continuation class** in Apex to make a long-running request to an external Web service. Process the response in a callback method. An asynchronous callout made with a continuation doesn't count toward the Apex limit of 10 synchronous requests that last longer than five seconds.
To make a long-running callout, define an Apex method that returns a Continuation object. We can use `@AuraEnabled (continuation = true)`
As per the requirement, the developer must collect key inputs from the end user before making the callout. This can be achieved using Lightning Web Component and then the apex class can be invoked from **Lightning Web Component.**

With this explanation, the correct answers are:
"Apex method that returns a Continuation object."
and
"Lightning web component."

Reference:
https://developer.salesforce.com/docs/atlas.en-us.lightning.meta/lightning/apex_continuations.htm

Create Notes

Q.53. A developer used custom settings to store some configuration data that changes occasionally. However, tests are now failing in some of the sandboxes that were recently refreshed.
What should be done to eliminate this issue going forward?

Answered Right

Ask Instructor

1. Set the setting type on the custom setting List.
2. Replace custom settings with custom metadata. (Correct Answer)(Selected Answer)
3. Set the setting type on the custom setting to Hierarchy.
4. Replace custom settings with static resources.

This is a scenario-based question related to using Custom Setting in Salesforce.

The scenario given in this question is:

- Custom settings are being used to store some configuration data that changes occasionally.
- However, tests are now failing in some of the sandboxes that were recently refreshed.

The ask is:

- What should be done to eliminate this issue going forward?

Custom Settings:
Custom Setting is used create custom sets of data and associate custom data for an organization, profile, or specific user. All custom settings data is exposed in the application cache, which enables efficient access without the cost of repeated queries to the database.
Please Note:
While custom settings data is included in sandbox copies, it is treated as data for the purposes of Apex test isolation. Apex tests must use `SeeAllData=true` to see existing custom settings data in the organization. As a best practice, create the required custom settings data in your test setup.
That means, while writing test class, we need to create test records for Custom Setting. There might be a situation that the custom setting records are changed and the test class creates improper custom setting records and therefore, its getting failed.

The more suitable way is to use Custom Metadata instead of Custom Setting.
Custom metadata is customizable, deployable, packageable, and upgradeable application metadata. First, you create a custom metadata type, which defines the form of the application metadata. Then you build reusable functionality that determines the behavior based on metadata of that type.

- You can control the visibility of custom metadata type while adding it in package.
- Custom metadata type is visible in test class without “SeeAllData” annotation. That means, while writing test class, there is no need to create test records if we use Custom Metadata.

With this explanation,

the correct answer is:
“Replace custom settings with custom metadata.”

References:
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_customsettings.htm
https://help.salesforce.com/s/articleView?id=sf.custommetadatatypes_about.htm&type=5

Create Notes

Q.54. A developer is creating a Lightning web component to display a calendar. The component will be used in multiple countries. In some locales, the first day of the week is a Monday, or a Saturday, or a Sunday.
What should the developer do to ensure the calendar displays accurately for users in every locale?

Answered Wrong

Ask Instructor

1. Use a custom metadata type to store key/value pairs. (Selected Answer)
2. Query the *FirstDayOfWeek* field from the Locale for the current user.
3. Use *UserInfo.getLocale()* in the component.
4. Import the *firstDayOfWeek* property in the component. (Correct Answer)

This is scenario-based question which is related to user locale. The scenario given in this question is:

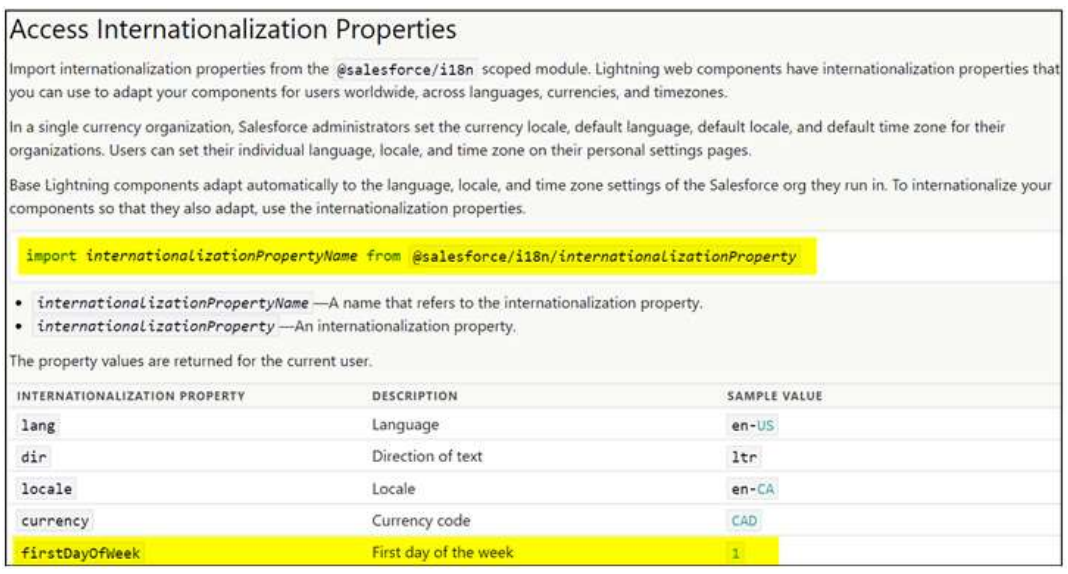
- There is lightning web component to display calendar.
- This component will be used in multiple countries.
- In some locales, the first day of the week is a Monday, or a Saturday, or a Sunday.

The ask is:

- What should the developer do to ensure the calendar displays accurately for users in every locale?

Let's go through each option and draw conclusion.
"Use a custom metadata type to store key/value pairs." is **not correct** because we can easily get the locale for user without storing it in metadata.
"Query the `FirstDayOfWeek` field from the Locale for the current user." is **not correct** because this is invalid.
"Use `UserInfo.getLocale()` in the component." is **not correct** because we can not use `UserInfo.getLocale()` in component. To use it, we need to use Apex Class methods.
"Import the `firstDayOfWeek` property in the component." is **correct** because we can import and use it in lightning web component.

Screenshot–Access Internationalization Properties



Access Internationalization Properties

Import internationalization properties from the `@salesforce/i18n` scoped module. Lightning web components have internationalization properties that you can use to adapt your components for users worldwide, across languages, currencies, and timezones.

In a single currency organization, Salesforce administrators set the currency locale, default language, default locale, and default time zone for their organizations. Users can set their individual language, locale, and time zone on their personal settings pages.

Base Lightning components adapt automatically to the language, locale, and time zone settings of the Salesforce org they run in. To internationalize your components so that they also adapt, use the internationalization properties.

```
import internationalizationPropertyName from @salesforce/i18n/internationalizationProperty
```

- *internationalizationPropertyName* —A name that refers to the internationalization property.
- *internationalizationProperty* —An internationalization property.

The property values are returned for the current user.

INTERNATIONALIZATION PROPERTY	DESCRIPTION	SAMPLE VALUE
lang	Language	en-US
dir	Direction of text	ltr
locale	Locale	en-CA
currency	Currency code	CAD
firstDayOfWeek	First day of the week	1

Reference:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/create_i18n

---

Q.55. A developer recently released functionality to production that performs complex commission calculations in Apex code called from an Opportunity trigger. Users report that the calculations seem incorrect because the values they see for commissions are wrong.
The developer has representative test data in their developer sandbox.
Which three tools or techniques should the developer use to execute the code and pause it at key lines to visually inspect values of various Apex variables?
Choose 3 answers.

Answered Wrong

Ask Instructor

1. Developer Console.
(Correct Answer)(Selected Answer)
2. Visual Studio Code.
(Correct Answer)(Selected Answer)
3. Breakpoints.
(Selected Answer)
4. Apex Replay Debugger.
(Correct Answer)
5. Workbench.

Create Notes

This question is related to debugging apex code.

The scenario given in this question is:

- The developer has representative test data in their developer sandbox.

The ask is:

- Which three tools or techniques should the developer use to execute the code and pause it at key lines to visually inspect values of various Apex variables.

The correct answers are:
Option **"Developer Console."** is correct because we can Use Developer Console checkpoints to debug your Apex classes and triggers.
Option **"Apex Replay Debugger."** is correct because we can debug code using the Launch Apex Replay Debugger with Current File command. We can invoke this command on an Apex test file, Anonymous Apex file or an Apex log file.
Option **"Visual Studio Code."** is correct because it is used with Apex Replay Debugger.

References:
https://help.salesforce.com/s/articleView?id=sf.code_dev_console_checkpoints_setting.htm&type=5
https://developer.salesforce.com/tools/vscode/en/apex/replay-debugger

---

Q.56. Cloud Kicks(CK) has custom Order and Order Line objects that represent orders placed by its customers.
A developer has a new requirement that CK's external enterprise resource planning (ERP) system must be able to integrate with Salesforce to create orders for existing customers in Salesforce whenever an order is placed in the ERP system.
What should the developer use to create the orders in Salesforce?

Answered Wrong

Ask Instructor

1. ConnectAPI (Correct Answer)
2. Change Data Capture API (Selected Answer)
3. SObject Tree API
4. Event Monitor API

A newly added scenario and an explanation will be given earlier.

Q.57. Consider the following code snippet:

```
public class searchFeature{
    public static List<List<sObject>> searchRecords(string searchquery) {
        return [FIND searchquery IN ALL FIELDS RETURNIG Account, Opportunity, Lead]; A
    }
}
```

developer created the following test class to provide the proper code coverage for the snippet above:

```
@isTest
private class searchFeature_Test{

    @TestSetup
    private static void makeData(){
          // insert opportunities,  accounts and lead
    }

    @isTest
    private static searchRecords_Test(){
        List<List<sObject>> records = searchFeature.searchRecords('Test');
        System.assertNotEquals(records.size(),0);
    }
}
```

However, when the test runs, no data is returned and the assertion fails.
Which edit should the developer make to ensure the test class runs successfully?

Answered Right

Ask Instructor

Create Notes

1. Enclose the method call with *Test.startTest()* and *Test.stopTest()*.
2. Implement the *seeAllData=true* attribute in the *@IsTest* is test annotation.
3. Implement the without sharing keyword in the *searchFeature* Apex class.
4. Implement the *setFixedSearchResults* method in the test class. (Correct Answer)(Selected Answer)

This question is related to writing test class for an apex class which uses SOSL query.

The scenario given in this question is:

- When the test runs, no data is returned and the assertion fails.

The ask is:

- Which edit should the developer make to ensure the test class runs successfully?

Let's learn few concepts.

Adding SOSL Queries to Unit Test

To ensure that test methods always behave in a predictable way, any Salesforce Object Search Language (SOSL) query that is added to an Apex test method returns an empty set of search results when the test method executes. If you do not want the query to return an empty list of results, you can use the **Test.setFixedSearchResults** system method to define a list of record IDs that are returned by the search.

All SOSL queries that take place later in the test method return the list of record IDs that were specified by the **Test.setFixedSearchResults** method.

The list of record IDs specified by the **Test.setFixedSearchResults** method replaces the results that would normally be returned by the SOSL query if it were not subject to any WHERE or LIMIT clauses. If these clauses exist in the SOSL query, they are applied to the list of fixed search results.

Screenshot –Example using **Test.setFixedSearchResults**

```
@isTest
private class SoslFixedResultsTest1 {

    public static testMethod void testSoslFixedResults() {
       Id [] fixedSearchResults= new Id[1];
       fixedSearchResults[0] = '001x0000003G89h';
       Test.setFixedSearchResults(fixedSearchResults);
       List<List<SObject>> searchList = [FIND 'test'
                                         IN ALL FIELDS RETURNING
                                            Account(id, name WHERE name = 'test' LIMIT 1)];
    }
}
```

With this explanation,

the correct answer is:

"Implement the setFixedSearchResults method in the test class."

Reference:

https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_testing_SOSL.htm

Create Notes

Q.58. A custom Aura component, *AddressValidation.cmp*, exists in the system. The Salesforce admin for the organization is unable to find and select the component while creating a quick action for the Account sObject.
What should the developer do to ensure the *AddressValidation.cmp* can be selected when creating a quick action?

Answered Right

Ask Instructor

1. Ensure the component implements the *force:lightningQuickAction* interface.
(Correct Answer)(Selected Answer)
2. Ensure the component implements The *lightning:actionOverride* interface.
3. Ensure the component implements the *force:hasRecordId* interface.
4. Ensure the access attribute of the *aura:component* tag is set to Global.

This question is related to accessing Aura Component from specific places.
The scenario given in this question is:

- There is an existing Aura Component named "AddressValidation.cmp"
- This Aura Component is not accessible while creating Quick Action.

The ask is:

- What should the developer do to ensure the AddressValidation.cmp can be selected when creating a quick action?

This requirement is related to implementing interfaces in Aura Component.
When we try to create an Aura Component we will be given the option to include interfaces, by adding interfaces we are trying to make the components accessible at different places in Lightning Experience.
There are many interfaces we can use in Aura Component depending on the requirement. As per the requirement given in this question, we need to use this Aura Component in quick action.

Lightning Quick Action:
Add the **force:lightningQuickAction** interface to a Lightning component to allow it to be used as a custom action in Lightning Experience or the Salesforce mobile app. You simply add the interface name to the component's implements attribute.

Screenshot – Implementing Interface in Aura Component

```
<!--quickAdd.cmp-->
<aura:component implements="force:lightningQuickAction">

    <!-- Very simple addition -->

    <lightning:input type="number" name="myNumber" aura:id="num1" label="Number 1"/>
    <lightning:input type="number" name="myNumber" aura:id="num2" label="Number 2"/>

    <br/>
    <lightning:button label="Add" onclick="{!c.clickAdd}"/>

</aura:component>
```

With this explanation and code example, the correct answer is

"Ensure the component implements the force:lightningQuickAction interface."
Reference:
https://developer.salesforce.com/docs/component-library/bundle/force:lightningQuickAction/documentation

---

Q.59. A developer is building a Lightning web component that searches for Contacts and must communicate the search results to other Lightning web components when the search completes.
What should the developer do to implement the communication?

Answered Right

Ask Instructor

1. Publish an event on an event channel.
2. Fire and application event.
3. Publish a message on a message channel. (Correct Answer)(Selected Answer)
4. Fire a custom component event.

Create Notes

This is a scenario-based question related to communication in Lightning Web Components.

The scenario given in this question is:

- There is a Lightning Web Component which searches for Contacts.
- This component must communicate the search results to other Lightning Web Components when the search completes.
    - That means, the components to which the search results should be communicated are not related.

Let's learn about component communication. There are typically 3 approaches for communication between the components:

1. Parent to Child Communication

- In this approach, parent component can communicate to child component.
- For this we have use the @api decorator to make the children properties / method public available so parent can be able to call it directly using JavaScript API.

2. Child to Parent Communication

- In this approach, child component can communicate to parent component.
- Custom Event is used to make the communication from Child Component to Parent Component. With LWC we can create and dispatch the custom event.

3. Communication between unrelated components(Scenario given in this question)

- To communicate between unrelated components, use LMS (Lightning Message Service)
- LMS is a publish and subscribe service that facilitates communication between Lightning web components, Aura components, and Visualforce pages.

As per the scenario given in this question, the lightning web component should communicate the search results to other Lightning web components.
To achieve this, LMS (Lightning Message Service) should be used which uses Publish-Subscribe model.
With this explanation, the correct answer is:
"Publish a message on a message channel." because it is part of publish-subscribe model.

For Code, Refer:
https://trailhead.salesforce.com/content/learn/projects/communicate-between-lightning-web-components/communicate-between-unrelated-components?trail_id=build-lightning-web-components
For Reference:
https://trailhead.salesforce.com/en/content/learn/projects/communicate-between-lightning-web-components

Create Notes

Q.60. A developer created a Lightning web component for the Account record page that displays the five most recently contacted Contacts for an Account. The Apex method, *getRecentContacts*, returns a list of Contacts and will be wired to a property in the component.

```
01:
02: public class ContactFetcher {
03:
04:     static List<Contact>getRecentContacs(Id accountId) {
05:       List<Contact> contacts = getFiveMostRecent(accountId);
06:       return contacts;
07:     }
08:
09:     private static List<Contact>getFiveMostRecent(Id accountId) {
10:           // ... Implementation...
11:     }
12: }
```

Which two lines must change in the above code to make the Apex method able to be wired? Choose 2 answers.

Answered Right

Ask Instructor

1. Add *@AuraEnabled(cacheable=true)* to line 08.
2. Remove *private* from line 09.

3. Add *@AuraEnabled(cacheable=true)* to line 03.
(Correct Answer)(Selected Answer)

4. Add *public* to line 04. (Correct Answer)(Selected Answer)

This is a scenario-based question related to accessing apex methods from Lightning Web Components.

The scenario given in this question is:

- There is a apex method *getRecentContacts()*, returns a list of Contacts.
- This apex method is wired to a property in the Lightning Web Component.

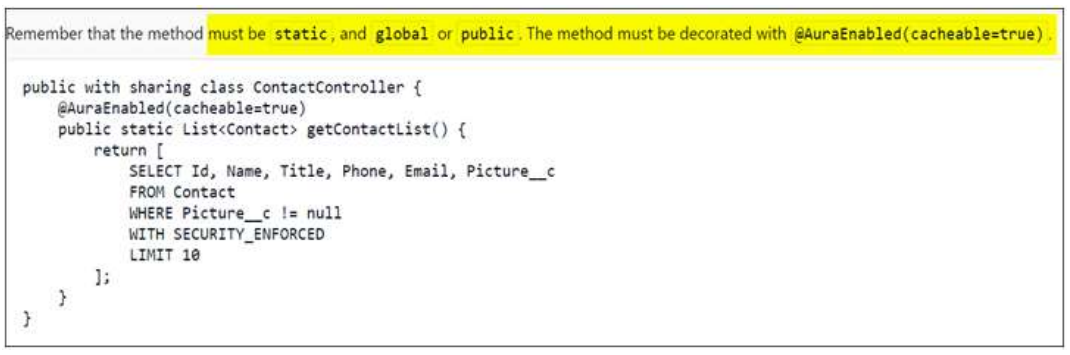
The ask is:

- Which two lines must change in the above code to make the Apex method able to be wired?

Please Note:

1. To expose an Apex method to a Lightning web component, the method must be:
    - Static
    - Either Global or Public
    - Annotate the method with @AuraEnabled.
2. To use @wire to call an Apex method, annotate the Apex method with @AuraEnabled(cacheable=true)

Screenshot – Sample Code as per Salesforce document



Now, coming back to the question and sample code given, the method at LINE-04 does not have any access modifier given like public or global and that method does not annoted with @AuraEnabled annotation.
With this explanation, the correct answers are:
"Add @AuraEnabled(cacheable=true) to line 03."
and
"Add public to line 04."

Reference:
https://developer.salesforce.com/docs/component-library/documentation/en/lwc/lwc.apex_wire_method

---

Q.61. A developer writes a Lightning web component that displays a dropdown list of all custom objects in the org from which a user will select. An Apex method prepares and returns data to the component.
What should the developer do to determine which objects to include in the response?

Answered Wrong

Ask Instructor

1. Check the *iscustom()* value on the sObject describe result.
(Correct Answer)
2. Use the *getCustomObjects()* method from the Schema class.
3. Import the list of all custom objects from *@salesforce/schema.*
(Selected Answer)
4. Check the *getObjectType()* value for 'Custom' and 'Standard' on the sObject describe result.

Create Notes

This is scenario-based question which is related to fetch data from Salesforce.

The scenario given in this question is:

- There is an apex method which returns the list of all custom objects.
- This apex method is called from Lightning Web Components to display the returned list of custom objects in UI.

The ask is:

- What should the developer do to determine which objects to include in the response?
  - That means, what logic can be used to determine if it's a custom object.

To achieve this, we can follow below approach:

1. Use *getGlobalDescribe()* method of Class Schema
This method returns a map of all sObject names (keys) to sObject tokens (values) for the standard and custom objects defined in your organization.

2. Use *getDescribe ()* method of Class *SobjectType*
Returns the describe sObject result for this field.

3. Use *isCustom* method of Class *DescribeSObjectResult*
Returns true if the object is a custom object, false if it is a standard object.

Screenshot – Apex Class & method with logic to get list of Custom Object

ObjectController.apxc

Code Coverage: None API Version: 54

```apex
public class ObjectController {

    /**
     * @purpose: method which return name of all the Custom Object
     * @annotaitons: @AuraEnabled(cacheable=true) because it should be exposed to LWC
     */
    @AuraEnabled(cacheable=true)
    public static List<String> fetchCustomObjectList(){
        List<String> customObjectList = new List<String>();
        for(Schema.SObjectType objTyp : Schema.getGlobalDescribe().Values())
        {
            Schema.DescribeSObjectResult describeSObjectResultObj = objTyp.getDescribe();
            if(describeSObjectResultObj.isCustom()){
                String name = objTyp.getDescribe().getName();
                customObjectList.add(name);
            }
        }
        System.debug('List of Custom Objects = '+customObjectList);
        return customObjectList;
    }
}
```

With this explanation,the correct answer is:

"Check the *iscustom()* value on the *sObject* describe result."

Reference:

https://developer.salesforce.com/docs/atlas.en-us.apexref.meta/apexref/apex_methods_system_sobject_describe.htm

Create Notes