**Vendor:**Oracle

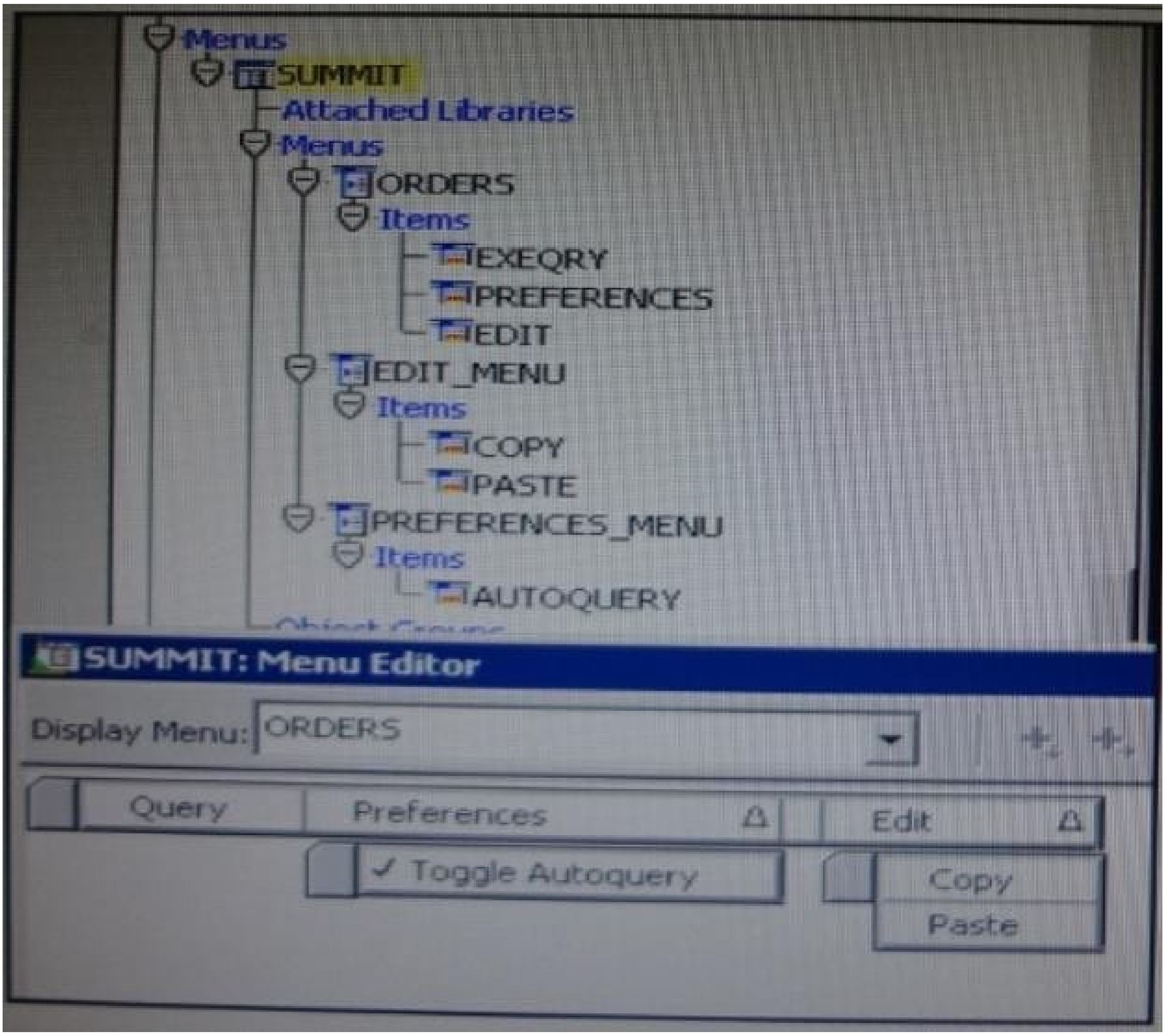
**Exam Code:**1Z0-151

**Exam Name:**Oracle Fusion Middleware 11g: Build Applications with Oracle Forms

**Version:**Demo

**QUESTION 1**

View the Exhibit.



The Summit menu is attached to the Orders form. The Toggle Autoquery menu item is a check box that toggles whether a query is automatically performed when the Orders form is first invoked. If the check box is deselected, users must manually query.

In addition to using the menu, users want to be able to toggle the autoquery preference directly from the form. You add a button named Toggle Autoquery with the following When-Button- Pressed trigger:

```
DECLARE
mi_id MENUITEMS;
```

```
BEGIN

mi_id ;=FIND_ITEM (\\'Preferences.AutoQuery\\')

/* Determine the current checked static of the AutoCommit menu checkbox item And toggle the checked

state*/

IF GET_ITEM_PROPERTY (mi_id, CHECKED) = \\'TRUE\\' THEN

SET_ITEM_PROPERTY (mi_id, CHECKED, PROPERTY_FALSE);

ELSE

SET_ITEM_PROPERTY (mi_id, CHECKED, PROPERTY_TRUE);

END IF;

END;
```

However, the trigger does not compile. What three changes must you make so that the trigger compiles successfully?

A. Change FIND_ITEM to FIND_MENU_ITEM.

B. Change \\'preferences.AutoQuery\\' to \\'orders.preferences.AutoQuery\\'.

C. Change \\'preferences.AutoQuery\\' to \\'AutoQuery\\'.

D. Change \\'preferences.AutoQuery\\' to \\'ORDERS.PREFERENCES>AUTOQUERY\\'.

E. Change \\'preferences.AutoQuery\\' to \\'AUTOQUERY\\'.

F. Change GET_ITEM_PROPERTY to GET_MENU_ITEM_PROPERTY

G. Change SET_ITEM_PROPERTY to SET_MENU_ITEM_PROPERTY

H. Change PROPERTY_FALSE to \\'FALSE\\'.

I. Change PROPERTY_TRUE to \\'TRUE\\'.

Correct Answer: AFG

A: Note: FIND_MENU_ITEM built-in Description Searches the list of menu items and returns a menu item ID when it finds a valid menu item with the given name. You must define an appropriately typed variable to accept the return value. Define the variable with a type of MenuItem.

Note 2:

FIND_ITEM built-in

Description

Searches the list of items in a given block and returns an item ID when it finds a valid item with the given name. You must define an appropriately typed variable to accept the return value. Define the variable with a type of Item.

Example (with FIND_MENU_ITEM, GET_MENU_ITEM_PROPERTY,

SET_MENU_ITEM_PROPERTY)

FIND_MENU_ITEM examples

```
/*
** Built-in: FIND_MENU_ITEM
** Example: Find the id of a menu item before setting
** multiple properties
*/
PROCEDURE Toggle_AutoCommit_Mode IS
mi_id MenuItem;
val VARCHAR2(10);
BEGIN
mi_id := Find_Menu_Item(\\'Preferences.AutoCommit\\');
/*
** Determine the current checked state of the AutoCommit ** menu checkbox item
*/
val := Get_Menu_Item_Property(mi_id,CHECKED);
/*
** Toggle the checked state
*/
IF val = \\'TRUE\\' THEN
Set_Menu_Item_Property(mi_id,CHECKED,PROPERTY_FALSE);
ELSE
Set_Menu_Item_Property(mi_id,CHECKED,PROPERTY_TRUE);
END IF;
END;
```

---

**QUESTION 2**

You have a text item in your form named Object1. You want to create Object2 as an exact duplicate of Object1. You want to be able to change the properties of Object2. However if you change Object1, you do not want Object2 to change.

Which method of reusing Object1 would be best for these requirements?

A. Copying Object1

B. Subclassing Object1

C. Creating a property class from Object1

D. Putting Object1 in an Object library

E. Referencing Object1 in a PL/SQL library

Correct Answer: A

When you copy an object, a separate, unique version of that object is created in the target module. Any objects owned by the copied object also get copied. Changes made to a copied object in the source module do not affect the copied object in the target module.

Incorrect answers:

B: With subclassing you can make an exact copy, and then alter the properties of some objects if desired.

If you change the parent class, the changes also apply to the properties of the subclassed object that you

have not altered. However, any properties that you override remain overridden.

This provides a powerful object inheritance model.

---

**QUESTION 3**

Iin an Order Entry form, the Order_Items block has a text item called Shipping Date, which is of the Date data type. The Lowest Allowed Value and the Highest Allowed Value properties are not defined for this item.

Assuming that the following are defined, which three are valid settings for the initial Value = property of the Shipping_Date Item?

A. $$DBDATE$$

B. :Orders.Order_Date + 3

C. SYSDATE

D. :GLOBAL.SHIP_DATE

E. :PARAMETER.SHIP

F. ORDER_ITEMS_SEQ

Correct Answer: ACD

A: Use $$DATE$$, $$DATETIME$$, and $$TIME$$ to obtain the local system date/time; use $$DBDATE$$,

$$DBDATETIME$$, and $$DBTIME$$ to obtain the database date/time, which may differ from the local system date/time when, for example, connecting to a remote database in a different time zone. ,, Use these variables only to set the value of the Initial Value, Highest Allowed Value or Lowest Allowed Value property.

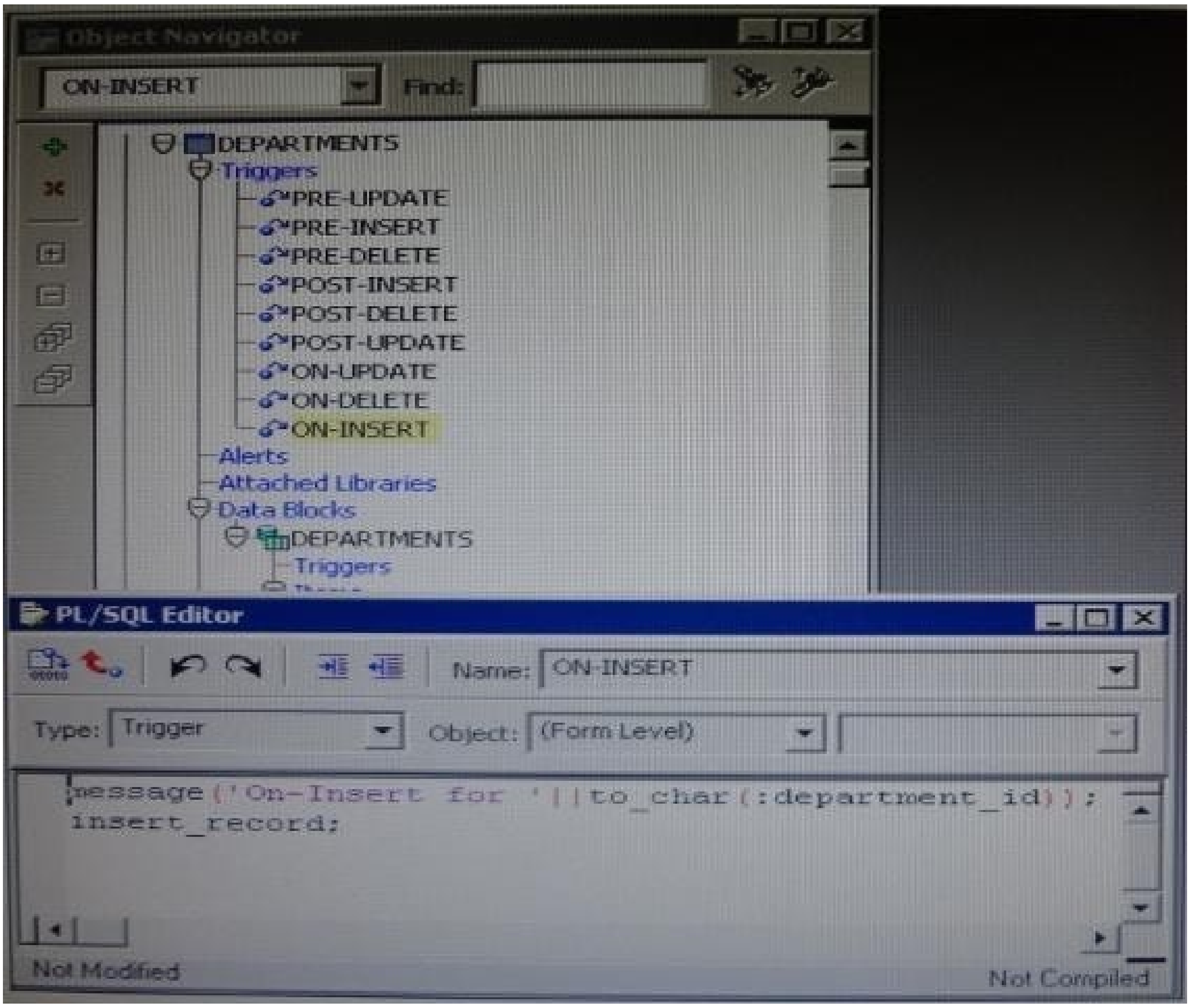
Incorrect answers:

B: This would work within a trigger.

---

**QUESTION 4**

View the Exhibit.



Name: ON-INSERT

Type: Trigger

Object: (Form Level)

```
message('On-Insert for '||to_char(:department_id));
insert_record;
```

Not Modified

Not Compiled

The Departments form has form level triggers defined as shown in the Exhibit. You have enabled array DML for the Departments block by setting its DML array size to 5 and setting the Primary Key property of Department id to Yes. You set runtime preferences for Forms Builder to use array processing.

You want to test that array processing is actually occurring, so, as shown in the Exhibit, the On- Update, On-Delete, and on insert triggers have code that:

1.

Displays a message to indicate which trigger is firing for which record?

2.

Does the thing (either updates, deletes, or inserts)

The other triggers simply display a message, indicating the trigger that is firing.

You run the form from Forms Builder, but from the messages, you can tell that array processing is not occurring. What should you change so that array DML gets implemented?

A. Delete the "On-" triggers; these triggers remove default processing, including array processing.

B. Set the DML array size to a value of 10 or greater; for array sizes loss than 10, array processing Is inefficient, and so is not performed.

C. Use the SYNCHRONIZE built-in after the MESSAGE built-in; otherwise, the message are not displayed until the processing is finished, giving the appearance that array processing is not taking place.

D. Set the Enforce Primary Key property of the Department block to Yes; it is not enough to just set the Department_Id to be the primary key.

Correct Answer: A

*

ON-INSERT

Override default processing for inserting a new row into the database during "post" processing Override the doDML() method in your entity class, and if theoperation equals DML_INSERT, then write code instead ofcalling the super.

*

ON-DELETE

Override default processing for deleting a row removed from the data block from the database during "post" processing Override the doDML() method in your entity class, and if theoperation equals DML_DELETE, then write code instead ofcalling the super.

*

ON-UPDATE

Override default processing for updating a row changed in the data block from the database during "post"

processing

Note: Use Array Processing

Both Forms Developer and Reports Developer are able to take advantage of the Oracle database server\\'s

array processing capabilities. This allows records to be fetched from the database in batches instead of

one at a time, and results in significantly fewer calls to the database. The downside of array processing is

that more space is required on the execution platform for storing the arrays of records returned.

If load on the network becomes a major bottleneck in the production environment, then set the Developer product\\'s runtime ARRAYSIZE parameter to as large a value as possible for the execution environment.

Note 2:

Query Array Processing

The default behavior of Forms is to process records one at a time. With array processing, astructure

(array) containing multiple records is sent to or returned from the server for processing.Forms supports

both array fetch processing and array data manipulation language (DML)processing. For both querying

and DML operations, you can determine the array size to optimizeperformance for your needs.

Note 3: You can enable array processing for queries by:

*Setting preferences:

-Select Edit > Preferences.

-Click the Runtime tab.

-Select the Array Processing check box.

* Setting block properties: -In the Object Navigator, double -click the node of the block to display the PropertyPalette. -Under the Records category, set the Query Array Size property to a number that represents the number of records in the array for array processing

---

**QUESTION 5**

The Orders form has the following triggers defined:

1.

Post-Text-Item on Customer_Id

2.

Pre-Text-Item on Sales_Rep_Id

3.

When-New-Item-instance on Sales_Rep_Id

The form\\'s Validation Unit property is set to Record.

When the user navigates from Customer_Id to Sales_Rep_Id, which triggers fire and what is the sequence?

A. 1, 2, and 3

B. 1, 3, and 2

C. 1 and 2 only

D. 2 and 3 only

E. 3 only

Correct Answer: A

All three triggers fire when you navigate

---

**QUESTION 6**

You have been assigned to maintain a forms application that was designed by a developer who has left the company.

The form uses different property classes to standardize the appearance of objects in the form. For example, all buttons should be of the same width.

The CV_Tools canvas contains several buttons. The Print invoice button is not quite wide enough to display its complete label.

In forms Builder, you open the Button_PC property class and change its Width property to a higher number. All the buttons become larger except the Print invoice button, which remains its original size. What could have caused this problem?

A. The Print invoice button was copied from, rather than subclassed from, the Button_PC property class.

B. The subclass information property for the Print invoice button is blank.

C. The Width property of the Print invoice button is an inherited property.

D. The Width property of the Button_PC property class is a variant property.

E. The X Position of the print invoice button is a variant property.

F. The X Position Property is not included in the Button_PC property

Correct Answer: A

---

**QUESTION 7**

What happens when you click Run Form Debug in Forms Builder?

A. The form runs on your local machine by using a debug executable client.

B. The form runs in a three-tier environment by using the application server URL that you specify in runtime preferences.

C. The form runs in a three tier environment by using the [debug] configuration in the FORMSWEB.CFG file.

D. The form runs in a simulated three-tier environment by using an applet viewer that is included with the product to enable debugging.

Correct Answer: B

Directly from the class materials from Oracle: As in the case when you run a form from Forms Builder with the Run Form button, the Run Form Debug button runs the form in a three-tier environment. It takes its settings from the Preferences window that you access by selecting Edit > Preferences from the main menu and clicking the Runtime tab.

---

**QUESTION 8**

To avoid overloading the database during busy times, you decide to restrict the queries that are executed on the Orders form so that users query by either Order ID or Customer ID during these times. Which trigger is most appropriate for the code to enforce this restriction?

A. When-New-Form-instance

B. When-New-Block-instance

C. On-Query

D. Pre-Query E. Post-Query

Correct Answer: D

About controlling queries with Pre-Query and Post-Query triggers

The Pre-Query and Post-Query triggers allow control over query processing. They can be defined at the

form or block level. Most often, attach them to specific blocks to control the query functionality of those

blocks.

The Pre-Query trigger fires just before Form Builder issues the SELECT statement to the database, after

the operator has defined the example record by entering query criteria in Enter Query mode.

Inside a Pre-Query trigger, the example record defined by the query criteria is the current record. This

means that trigger code can read and set the values of items in the example record using standard

:block_name.item_name syntax.

A Pre-Query trigger can be used to disallow query conditions that might be invalid. When a form is in Enter

Query mode, normal validation is suspended and no validation triggers fire as they do in Normal mode.

The Pre-Query trigger thus allows you to verify that any values entered by the operator are valid query

conditions.

When invalid query conditions have been entered, you can abort the query by raising the

FORM_TRIGGER_FAILURE built-in exception in the Pre-Query trigger.

You can also call SET_BLOCK_PROPERTY to modify the block\\'s WHERE and ORDER BY clauses from

within the Pre-Query trigger, to further restrict or order the records the query will retrieve.

---

**QUESTION 9**

Which statement is true about flexible code?

A. It is designed for reuse.

B. It typically includes hard-coded object names.

C. It is more difficult to maintain.

D. It is more difficult to write, so it decreases developer productivity.

E. It is specific to a particular module.

Correct Answer: A

---

**QUESTION 10**

To troubleshoot a problem with a form, you have added a call to the MESSAGE () built-in at the beginning of the When-Validate-Item trigger of the Customer_Id then in the Orders Block of the Orders form. The message simply states that the trigger is firing.

You run the form, make a change in Customer_Id, and then tab out of the item but the message does not appear. What are two possible causes for this problem?

A. The form is in Enter-Query mode.

B. The item is using an LOV for validation.

C. The validation unit of the form needs to be changed.

D. The MESSAGE () built-in is not allowed in validation triggers.

E. There is a syntax error in the call to the MESSAGE() built-in.

F. Validation for the Customer_Id item failed.

Correct Answer: AB

A: WHEN-VALIDATE-ITEM does not fire in ENTER-QUERY mode.

B: LOVs in ENTER-QUERY Mode

LOVs in ENTER-QUERY mode should be used sparingly, as Query Find is the preferred method for a user to locate records.

You should only code them where they dramatically improve the usability of ENTER-QUERY mode, and you expect this mode to be used regularly despite Query Find.

An LOV in ENTER-QUERY mode should display all values that the user can query, not just currently valid values.

EXAMPLE: An LOV for vendors in a purchase order form in enter-query mode shows all vendors that could ever be placed on a PO, not just the set of vendors that currently are allowed to be placed on a PO.

Do not reuse the entry LOV in ENTER_QUERY mode unless it provides the correct set of data for both modes.

Important: WHEN-VALIDATE-ITEM does not fire in ENTER-QUERY mode. Therefore, you cannot depend on the WHEN-VALIDATE-ITEM trigger to clear hidden fields when selecting from an ENTER-QUERY LOV.

Note: Validation occurs when you press enter, when you navigate away from the item, or when you save your block to the database.

Note 2: When-Validate-Item trigger Description Fires during the Validate the Item process. Specifically, it fires as the last part of item validation foritems with the New or Changed validation status. Definition Level form, block, or item Legal Commands SELECT statements, unrestricted built-ins Enter Query Mode no Usage Notes

*

Use a When-Validate-Item trigger to supplement Form Builder default item validation processing.

*

It is possible to write a When-Validate-Item trigger that changes the value of an item that FormBuilder is validating. If validation succeeds, Form Builder marks the changed item as Valid anddoes not re-validate it. While this behavior is necessary to avoid validation loops, it does make itpossible for your application to commit an invalid value to the database.

*

The setting you choose for the Defer Required Enforcement property can affect the When- Validate-Item trigger. See Defer_Required_Enforcement for details. On Failure If fired as part of validation initiated by navigation, navigation fails, and the focus remains on theoriginal item. Fires In Validate the Item

---

**QUESTION 11**

You are implementing a JavaBean in a form, the bean has no visible component on the form when invoked, the bean displays an input text where users enter a zip code, the bean has a single method that returns a short weather forecast for that zip code as a character value.

How can you retrieve that value so that you can display it to the user?

A. Use FBEAN.ENABLE_EVENT to register a listener for the bean event. Obtain the value of SYSTEM.CUSTOM_ITEM_EVENT in a When-Custom item Event trigger, and then use that value as an argument to the MESSAGE built in.

B. Use FBEAN.REGISTER_BEAN as an argument to the MESSAGE built in to invoke the bean\\'s method and return the value as a message displayed to the user.

C. Use FBEAN.REGISTER_BEAN to register the bean, so that when the user enters a zip code into the bean s input text, the value is automatically displayed in the bean area item.

D. Use FBEAN.ENABLE_EVENT to register a listener for the bean event. Obtain the value of SYSTEM.CUSTOM_EVENT_PARAMETER in a When_Custom-item-Event trigger, and then use that value as an argument to the MESSAGE built-in.

Correct Answer: D

When a user interacts with a JavaBean at run time, it usually causes an event to occur. You can use FBEAN.ENABLE_EVENT to register a listener for the event, so that when the event occurs Forms will fire the When-

Custom-Item-Event trigger. In this trigger, you can code a response to the event. The :SYSTEM.CUSTOM_ITEM_EVENT and :SYSTEM.CUSTOM_EVENT_PARAMETERS variables contain the name of the event and information the bean is sending to the form.

---

**QUESTION 12**

Which setting of :SYSTEM.MESSAGE_LEVEL suppresses all system messages?

A. 0

B. 10

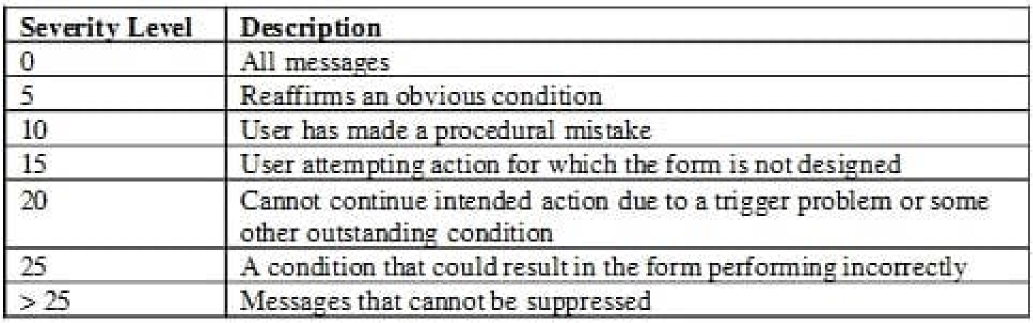
C. 25

D. 50

E. You cannot use :SYSTEM.MESSAGE_LEVEL to suppress all system messages

Correct Answer: E

Controlling System Messages You can prevent system messages from being issued, based on their severity level. FormsBuilder classifies every message with a severity level that indicates how critical or trivial the information is; the higher the numbers, the more critical the message. There are six levels that you can affect.



| Severity Level | Description |
|---|---|
| 0 | All messages |
| 5 | Reaffirms an obvious condition |
| 10 | User has made a procedural mistake |
| 15 | User attempting action for which the form is not designed |
| 20 | Cannot continue intended action due to a trigger problem or some other outstanding condition |
| 25 | A condition that could result in the form performing incorrectly |
| > 25 | Messages that cannot be suppressed |

Note: Suppressing Messages According to Their Severity In a trigger, you can specify that only messages above a specified severity level are to be issuedby the form. You do this by assigning a value to the MESSAGE_LEVEL system variable. Formsthen issues only those messages that are above the severity level defined in this variable.The default value for MESSAGE_LEVEL (at form startup) is 0. This means that messages of all severities are displayed.