KillTest

更に上のクオリティ　更に上のサービス

問題集

Exam : **MB2-498**

Title : Extending Microsoft CRM 3.0

Version : DEMO

1. You have been asked to create a web portal for your company's customers which allows each customer to view the data stored against their Microsoft CRM account, but not any other accounts. The company has over 1000 customers who may use this portal, and you want to minimize the licensing cost, while maintaining sufficient security. What users and licenses should you use for this scenario?

A. Create one Microsoft CRM user account with a Microsoft CRM client license that is used by all customers. Use this account for authentication in the portal and for accessing Microsoft CRM data.

B. Create a separate Microsoft CRM user account for each customer, each with a Microsoft CRM client license. Use each account for authentication in the portal and for accessing Microsoft CRM data.

C. Create a separate, non-Microsoft CRM authentication mechanism for each customer in the portal. Purchase a Microsoft CRM External Connector License. Create one proxy account which the portal uses to access Microsoft CRM data.

D. Create a separate, non-Microsoft CRM authentication mechanism for each customer in the portal. Retrieve Microsoft CRM data directly from the SQL database using filtered views, hence no Microsoft CRM licenses are required.

Answer: C

2. Your company has a requirement that whenever a user creates a case in Microsoft CRM, a record is written to a custom application. As the custom application can be slow to update, you want to ensure that the code you write does not impact the Microsoft CRM user's experience. You also want to ensure that only data which is saved to Microsoft CRM is written to the custom application. How should you write the logic to update the custom application?

A. As a Pre-Callout.

B. As a Post-Callout.

C. Using client-side Java Script that runs on a form's OnSave event.

D. As a workflow assembly that is called from a workflow rule that runs on the Create event.

Answer: D

3. You need to build some custom validation code that will force Microsoft CRM users to enter values in certain fields of an oppportunity once its CloseProbability exceeds 50%. If a user fails to enter appropriate data an error message displays, and they are prevented from saving the opportunity. How can you achieve this?   Choose the 2 that apply.

A. Use a Post-Callout.

B. Use a Pre-Callout.

C. Use client-side Java script.

D. Use a workflow assembly.

Answer: BC

4. You are creating an ASP .NET application to extend your company's Microsoft CRM implementation, and you want the application to resemble the Microsoft CRM application. How should you achieve this with the minimum effort, while ensuring your extensions are fully supported?

A. Reference the cascading style sheets used by the Microsoft CRM application.

B. Use the Microsoft CRM ASP .NET web controls within your custom pages.

C. Create .aspx web pages that inherit from the class Microsoft.Crm.Application.BasePage.

D. Use the styles in the cascading style sheet file template.css that is supplied with the Microsoft CRM 3.0 SDK.

Answer: D

5. You are creating custom web pages that extend Microsoft CRM contact functionality, using the Microsoft CRM web service. You want your web pages to reflect any changes made to the Microsoft CRM entity and attribute display names. How should you do this, with the minimum of additional development work?

A. Read the display names from the Microsoft CRM web service MetadataService.

B. Read the display names from the Microsoft CRM web service CrmService.

C. Store all text in a satellite .Net assembly. Create a separate copy of the satellite assembly for each deployment with the relevant text.

D. Use .Net Reflection within your web pages to determine the entity and attribute display names at runtime from the WSDL generated by the Microsoft CRM web service.

Answer: A

6. You want to display Microsoft CRM data within an ASP .Net DataGrid web control. It is important that only data the user has permission to see is displayed. Which of the following techniques can you use? Choose the 2 that apply.

A. Use the RetrieveMultiple method of the CrmService web service to retrieve the Microsoft CRM data as a BusinessEntityCollection, and bind this to the DataGrid.

B. Use the RetrieveMultiple method of the CrmService web service to retrieve the Microsoft CRM data as a DataSet, and bind this to the DataGrid.

C. Use the RetrieveMultiple method of the CrmService web service to retrieve the Microsoft CRM data as a BusinessEntityCollection. Use custom code to iterate through the BusinessEntityCollection to add the data to the DataGrid.

D. Use classes in the System.Data .Net Framework namespace to populate a DataSet with the results of a SQL query against Microsoft CRM filtered views. Bind this DataSet to the DataGrid.

Answer: CD

7. You are building a custom web application to update Microsoft CRM data via the Microsoft CRM

D.

Answer: C

10. You want to return multiple values from a method in a workflow .NET assembly. How should you do this?

A. Define multiple parameters as By Reference, and return values through these parameters.

B. Define the return value as a string. Concatenate all the values into the string, and use string manipulation functions within the workflow rule to split the different values.

C. Define the return value of type XML. Build all the values into the one XML document, and use the retval element in the workflow.config file.

D. Define the return value of type entity. Return a DynamicEntity with the values set a properties of the DynamicEntity.

Answer: C

11. You work for an international organization that, for security and localization reasons, has several Microsoft CRM servers. You have developed a workflow assembly that is intended to be deployed to all the Microsoft CRM servers. The assembly has a strong name. Some of the CRM servers will already have workflow assemblies that are specific to that server. What steps should you take to deploy the assembly with the minimum of effort?

A. For each server, update their workflow.config file with the method definitions in your assembly.

Copy the assembly to the assembly directory within Microsoft CRM.

B. Create a workflow.config file which contains all the required configuration. Replace each server's workflow.config file with your workflow.config file.

Copy the assembly to the assembly directory within Microsoft CRM.

C. Create a file <YourAssembly>.workflow.config which contains all the required configurations. For each server, add this to the directory which contains the workflow.config file.

Copy the assembly to the assembly directory within Microsoft CRM.

D. Install the assembly into the Global Assembly Cache on each server.

Answer: A

12. You are writing a Post-Callout which updates Microsoft CRM data via the CrmService web service. How can you ensure that the updates are made using the context of the user whose changes caused the callout to run?

A. Create a parameter of datatype caller. Read the data in the userid XML element of the data passed to this parameter, and assign it to the CallerIdValue of the CrmService object.

B. Read the user information from UserId property of the userContext parameter. Assign this value into the CallerIdValue of the CrmService object.

C. Read the user information from UserId property of the userContext parameter. Create a new instance of the NetworkCredential class using the UserId, and assign this credential to the Credentials property of the CrmService object.

D. Create a parameter of datatype caller. Read the data in the userid XML element of the data passed to this parameter, and assign this credential to the Credentials property of the CrmService object.

Answer: B

13. Within a PostCreate callout method, what is the quickest way to determine the primary key of the newly created entity?

A. Read the value of the InstanceId property of the CalloutEntityContext parameter passed to the callout method.

B. Within the callout.config.xml file, specify that the primary key attribute should be passed as a postvalue. Read this value from the postEntityImageXml parameter passed to the callout method.

C. Read the value of the EntityID parameter passed to the callout method.

D. It is not possible to determine the primary key of the newly created entity, as the primary key is only assigned after all callouts have completed.

Answer: A

14. You have written a PostUpdate callout that performs calculations on the Microsoft CRM order record when it is changed, and updates the record with the results of the calculation. When you try to update an order through the Micrsoft CRM interface it takes a long time, and ultimately gives a timeout error. During the update you notice the processor utilization on the server rises significantly. What should you do to resolve these problems?

A. In callout.config.xml, add the following attribute to the subscription element for the callout:

timeout="2"

B. In callout.config.xml, add the following attribute to the subscription element for the callout:

onerror="ignore"

C. Rewrite the callout code to ensure that the updates to the order object only occur once as a result of changes by the user, and do not repeat endlessly as a result of the update made by the callout.

D. Rewrite the callout code to make the updates to the order record asynchronously.

Answer: C

15. You have created a callout assembly, MyCompany.MyCallout, and deployed it on a Microsoft CRM server. You now want to debug the callout. To which process should you attach the debugger?

A. w3wp.exe

B. CrmWorkflowService.exe

C. MyCompany.MyCallout.dll

D. MyCompany.MyCallout.exe

Answer: A

16. You have created a new callout assembly that contains a PostCreate callout for the Lead entity. You have copied the assembly to the correct directory on the Microsoft CRM server, and added the callout configuration details to the callout.config.xml file. However, when you create a new lead, the callout does not fire. No errors appear in the event log, and previously deployed callouts work correctly What should you do to resolve the problem ?

A. Run iisreset on the Microsoft CRM server.

B. Publish the lead entity via the Microsoft CRM Customization user interface.

C. Install the callout assembly in the Global Assembly Cache on the Microsoft CRM Server.

D. Restart the CrmWorkflowService.

Answer: A

17. You have a PostUpdate callout that has been deployed to your Microsoft CRM Server. You are now trying to debug the callout using the Visual Studio 2003 debugger. You have attached the debugger to the correct process, and set a breakpoint on first line of code in the PostUpdate method, but the code is not stopping on the breakpoint. You are sure the callout code is running. What could be the problem?

A. You are using the wrong version of the debugger. You need to use Visual Studio 2005 to debug callouts.

B. You have not deployed a program database (pdb) file for the callout assembly to the server.

C. The callout assembly does not have a strong name.

D. In callout.config.xml, the subscription element for the callout has the attribute onerror="ignore".

Answer: B

18. You are writing a PostUpdate callout that needs to identify when the value of an attribute is changed. How should you do this?

A. Ensure there is a prevalue and a postvalue element for the attribute in callout.config.xml.

Deserialize the preImageEntityXml and postImageEntityXml parameters passed to the callout, and compare the values for the attribute.

B. Ensure there is a prevalue and a postvalue element for the attribute in callout.config.xml.

Compare the string values of the preImageEntityXml and postImageEntityXml parameters.

C. Ensure there is a postvalue element for the attribute in callout.config.xml.

Use the Retrieve method of the CrmService web service to read the current value of the attribute.

Deserialize the postImageEntityXml parameters passed to the callout and compare the value of the

attribute to that returned by the Retrieve method.

D. Ensure there is a postvalue element for the attribute in callout.config.xml.

Use the Retrieve method of the CrmService web service to read the current value of the attribute.

Compare this value to the postImageEntityXml parameter.

Answer: A

19. You have two callouts (CalloutA and CalloutB) that both fire on the PostUpdate event of the Task entity.

How can you ensure that CalloutA always runs before CalloutB, with the minimum of code changes?

A. Add a CalloutOrder attribute to the subscription element for each callout in the callout.config.xml file.

Set the value of the CalloutOrder attribute for CalloutA to be a lower integer than the value for CalloutB.

B. Remove the definition for CalloutB from callout.config.xml. Call CalloutB directly at the end of the code for CalloutA.

C. In callout.config.xml, change the event attribute for CalloutA to PreCallout.

D. In callout.config.xml, ensure the subscription element for CalloutA appears earlier in the file than that for CalloutB.

Answer: D

20. You are writing a Post-Callout that should identify when the CloseProbability attribute of the Opportunity entity is changed, and by how much, based on the parameters passed into the callout. Which of the following partial callout configurations could you use for this callout? Select two. Each correct answer forms a complete solution   Choose the 2 that apply.

A. <subscription ?>

<prevalue>closeprobability</prevalue>

</subscription>

B. <subscription ?>

<postvalue>closeprobability</postvalue>

</subscription>

C. <subscription ?>

<prevalue>closeprobability</prevalue>

<postvalue>closeprobability</postvalue>

</subscription>

D. <subscription ?>

<prevalue>closeprobability</prevalue>

<postvalue>@all</postvalue>

</subscription>

Answer: CD