## Microsoft

# 111-056

*Microsoft .NET Framework 2.0-DistributedApplication Development*

**Answer:** B

**QUESTION:** 227
A Web service exposes the following Web method.
public: [WebMethod(CacheDuration=60)]
array<Byte> ^GetImage(String ^imageId) { ...}
The Web method generates responses that are greater than 1 MB in size. You need to configure the Web method to minimize memory usage on the server. Which two actions should you perform? (Each correct answer presents part of the solution. Choose two.)

A. Apply a SoapDocumentMethod attribute to the method declaration with the ParameterStyle property set to SoapParameterStyle.Wrapped.
B. Apply a SoapDocumentMethod attribute to the method declaration with the ParameterStyle property set to SoapParameterStyle.Bare.
C. Set the BufferResponse property of the WebMethod attribute to True.
D. Set the BufferResponse property of the WebMethod attribute to False.
E. Set the value of the CacheDuration property of the WebMethod attribute to 0.
F. Set the value of the CacheDuration property of the WebMethod attribute to 300.

**Answer:** D, E

**QUESTION:** 228
You create a serviced component. You install the component into the COM+ catalog. COM+ runs on the server. A Windows-based application that is installed on multiple desktop computers must use the component. The component must run on the server, but the Windows-based application must send component method calls over the network to the component. The communications protocol used is DCOM. You need to ensure that the Windows-based application can connect to the component. What should you do?

A. In the client application, create the serviced component using the Marshal class to bind to the following moniker."tcp://MyComponents.MyServicedComponent"
B. In the client application, create the serviced component using the Marshal class to bind to the following moniker."dcom://MyComponents.MyServicedComponent"
C. Generate a proxy component for the serviced component and install it on the desktop computers that are running the client application. In the client application, create a reference to the serviced component's assembly.
D. Generate a proxy component for the serviced component and install it on the desktop computers that are running the client application. In the client application, create a reference to

the serviced component's type library.

**Answer:** C

**QUESTION:** 229
A Windows-based application receives messages from a message queue named PriorityQueue. The client application sets the Priority property on the message before sending it. However, received Message objects do not have the Priority property assigned. The following code is used to receive the messages. (Line numbers are included for reference only.)
01 MessageQueue^ queue = gcnew MessageQueue(".\\PriorityQueue");
02 Message^ m = queue->Receive();
03 Console::WriteLine(m->Priority);
You need to ensure that the Windows-based application that receives the messages can access the Priority property. What should you do?

A. Replace line 01 with the following code segment.MessageQueue^ queue = gcnew MessageQueue(".\\PriorityQueue", QueueAccessMode::ReceiveAndAdmin);
B. Insert the following line of code between lines 01 and 02.queue->MessageReadPropertyFilter->Priority = true;
C. Insert the following code segment between lines 01 and 02.queue->Formatter = gcnew System::Messaging::XmlMessageFormatter(gcnew array<Type^> { String::typeid });
D. Insert the following code segment between lines 01 and 02.DefaultPropertiesToSend^ s = gcnew DefaultPropertiesToSend();s->Priority = MessagePriority::High;queue-DefaultPropertiesToSend = s;

**Answer:** B

**QUESTION:** 230
You are writing an installation application for a Windows Forms application. The Windows Forms application requires a private message queue named MyQueue. You need to ensure that the message queue exists after installation. What should you do?

A. Add an Installer class to the Windows Forms application project. Write the following code in the AfterInstall event for the Installer class.MessageQueue^ q = gcnew MessageQueue(".\\Private$\\MyQueue");
B. Add an Installer class to the Windows Forms application project. Write the following code in the AfterInstall event for the Installer class.System::Collections::Queue^ q = gcnew System::Collections::Queue();q->Enqueue(".\\Private$\\MyQueue");

C. Add an Installer class to the Windows Forms application project. Write the following code in the AfterInstall event for the Installer class.System::Collections::Generic::Queue<Message^> q = gcnew System::Collections::Generic::Queue<Message^>();q.Enqueue(gcnew Message(path));
D. Add an Installer class to the Windows Forms application project. Write the following code in the AfterInstall event for the Installer class.MessageQueue^ q = MessageQueue::Create(".\\Private$\\MyQueue");

**Answer:** D

**QUESTION:** 231
You are writing an application that will run on a portable computer. The application uses a private queue named MyAppQueue to store messages. You need to ensure that the message is retained when the portable computer is restarted. What should you do?

A. Write the following code segment to send the message.MessageQueue^ q = gcnew MessageQueue(".\\$Private\\MyAppQueue");Message^ m = gcnew Message("My message body");m->UseJournalQueue = true;q->Send(m, "message");
B. Write the following code segment to send the message.MessageQueue^ q = gcnew MessageQueue(".\\$Private\\MyAppQueue");Message^ m = gcnew Message("My message body");m->Recoverable = true;q->Send(m, "message");
C. Write the following code segment to send the message.MessageQueue^ q = gcnew MessageQueue(".\\$Private\\MyAppQueue");Message^ m = gcnew Message("My message body"); MessageQueueTransaction^ trans = gcnew MessageQueueTransaction();trans->Begin();q->Send(m, "message", trans);trans->Commit();
D. Write the following code segment to send the message.MessageQueue^ q = gcnew MessageQueue(".\\$Private\\MyAppQueue");Message^ m = gcnew Message("My message body");m->Priority = MessagePriority::High;q->Send(m, "message");

**Answer:** B

**QUESTION:** 232
A message queue named SecureQueue requires incoming messages to be authenticated. When an application attempts to send a message to SecureQueue, the following exception is thrown.User's internal Message Queuing certificate does not exist.The following code is used to send the message. (Line numbers are included for reference only.)
01 MessageQueue^ mq = gcnew MessageQueue(".\\SecureQueue");
02 Message^ m = gcnew Message("Test Message");
03 m->UseAuthentication = true;04 mq->Send(m);
You need to ensure that a message can be sent to SecureQueue without the exception being thrown. What should you do?

A. Insert the following line of code between lines 03 and 04.m->AuthenticationProviderType = CryptographicProviderType::RsqSig;
B. Insert the following line of code between lines 03 and 04.m->AuthenticationProviderName = "RsqSig";
C. Replace line 03 with the following line of code.m->AttachSenderId = true;
D. Replace line 03 with the following line of code.m->EncryptionAlgoritm = EncryptionAlgorithm::Rc4;

**Answer:** C

**QUESTION:** 233
You are writing an application that reads messages from a message queue. The name of the message queue is stored in a member variable named queueName. When a message is read, the application processes the message. The code for the application is as follows:
public ref class MyApp { MessageQueue^ queue;
public: MyApp() { queue = gcnew
MessageQueue(queueName, QueueAccessMode::Receive);
queue->ReceiveCompleted += gcnew ReceiveCompletedEventHandler(this,
&MyApp::ReceivedMessage);
queue->BeginReceive();
}
private: bool KeepListening() { ... }
private: void ProcessMessage(Message^ m) { ... }};
You need to ensure that the application continues to read messages when the KeepListening method returns True, and stops when the KeepListening method returns False. What should you do?

A. Use the following implementation of the ReceivedMessage method.private: void ReceivedMessage(Object^ sender, ReceiveCompletedEventArgs^ e) { ProcessMessage(queue->EndReceive(e->AsyncResult)); if (KeepListening() == true) { queue->BeginReceive(); }}
B. Use the following implementation of the ReceivedMessage method.private: void ReceivedMessage(Object^ sender, ReceiveCompletedEventArgs^ e) { queue = gcnew MessageQueue(queueName, QueueAccessMode::Receive); ProcessMessage(queue->EndReceive(e->AsyncResult)); if (KeepListening() == true) { queue->BeginReceive(); }}
C. Use the following implementation of the ReceivedMessage method.private: void ReceivedMessage(Object^ sender, ReceiveCompletedEventArgs^ e) { queue = gcnew MessageQueue(queueName, QueueAccessMode::Receive);
ProcessMessage(queue::EndReceive(e->AsyncResult)); if (KeepListening() == false) { queue->ReceiveCompleted -= gcnew ReceiveCompletedEventHandle(this,

&MyApp::ReceivedMessage); }}
D. Use the following implementation of the ReceivedMessage method.private: void ReceivedMessage(Object^ sender, ReceiveCompletedEventArgs^ e) { ProcessMessage(queue->EndReceive(e->AsyncResult)); if (KeepListening() == false) { queue.ReceiveCompleted -= new ReceiveCompletedEventHandler(this, &MyApp::ReceivedMessage); }}

**Answer:** A

**QUESTION:** 234
Users report that a Windows-based application does not run properly. When users attempt to complete a particular action, the following error message text appears.Unable to find assembly 'myservices, Version=1.0.0.0,Culture=neutral, PublicKeyToken=29b5ad26c9de9b95'.You discover that the error occurs when the Windows-based application attempts to call functionality in a serviced component that was registered by using the following command.regsvcs.exe myservices.dllYou need to ensure that the application can call the functionality in the serviced component without throwing the exception. What should you do?

A. Run the following tool from the command line.gacutil.exe /i myservices.dll
B. Run the following tool from the command line.regasm.exe myservices.dll
C. Copy the serviced component assembly into the following folder.
C:\Program Files\ComPlus Applications
D. Copy the serviced component assembly into the following folder.C:\WINDOWS\system32\Com

**Answer:** A

**QUESTION:** 235
You are writing a .NET Framework remoting client application. The server application raises events to the client application using an interface named IBroadcaster and an event wrapper. The event wrapper and the interface are located in an assembly named General that is common to the client and server. The server configuration file is as follows:
<system.runtime.remoting>
<application> <channels> <channel ref="http" port="2020"> <serverProviders>
<formatter ref="binary" typeFilterLevel="Full" /> </serverProviders> <clientProviders>
<formatter ref="binary" /> </clientProviders> </channel> </channels> <service>
<wellknown mode="Singleton" type="Server.Broadcaster, Server"
objectUri="Broadcaster.soap" />
</service> </application></system.runtime.remoting>
You need to ensure that the server can raise events on the client application. What should you do?

A. Configure the remoting configuration file for the client application to use Server.Broadcaster, Server as the type in the wellknown element. Do not specify a port for the channel.
B. Configure the remoting configuration file for the client application to use Server.Broadcaster, Server as the type in the wellknown element. Specify 0 as the port for the channel.
C. Configure the remoting configuration file for the client application to use General.IBroadcaster, General as the type in the wellknown element. Do not specify a port for the channel.
D. Configure the remoting configuration file for the client application to use General.IBroadcaster, General as the type in the wellknown element. Specify 0 as the port for the channel.

**Answer:** D

**QUESTION:** 236
A .NET Framework remoting server hosts a class library that contains the following class.public ref class SimpleMathClass :
public MarshalByRefObject{
public: void LogData(DataRow^ dr) { ... }};
Users of a Windows-based client application report that the application often becomes nonresponsive. You discover that the application makes calls to the LogData method that take several seconds to return. You need to ensure that calls to the LogData method can be processed without causing the client application to become nonresponsive. What should you do?

A. Modify the class to implement the IAsynchResult interface.
B. Modify the method so that it is an anonymous method.
C. Apply the OneWayAttribute attribute to the LogData method.
D. Apply the SoapRpcMethodAttribute attribute to the LogData method and initialize its OneWay member to True.

**Answer:** C

**QUESTION:** 237
You write a .NET Framework remoting application that broadcasts messages to client computers through a central server by raising events on the client computers. Message details are contained in an argument in the event delegate, as shown in the following code segment.
public ref class BroadcastEventArgs {
public: String^ Message;

public: String^ Sender;
public: DateTime TimeSent;};
public delegate void MessageArrivedHandler(BroadcastEventArgs^ args);
You need to ensure that the client computer can access the message details contained in the event argument. Your solution cannot change the common language runtime (CLR) security restrictions. What should you do?

A. Add the well-known type named BroadcastEventArgs in the server and configure the WellKnownObjectMode mode as SingleCall.
B. Change the BroadcastEventArgs class so that it extends the MarshalByRefObject object.
C. Apply the Serializable attribute to the BroadcastEventArgs object.
D. In the client application's remoting configuration file, make the BroadcastEventArgs class a client-activated object (CAO) that points to the server.

**Answer:** C

**QUESTION:** 238
A .NET Framework remoting server hosts a class library that contains the following class.public ref class SimpleMathClass : public MarshalByRefObject { public: int LogData(DataRow dr) { //Lengthy database calls ... }}; Users of a Windows-based client application report that the application often becomes nonresponsive. You discover that the application makes calls to the LogData method that take several seconds to return. The return value is required to generate reports in the client application. You need to ensure that calls to the LogData method can be processed without making the client application nonresponsive. What should you do?

A. Apply the OneWay attribute to the LogData method.
B. In the client code, declare a delegate that has the same signature as the LogData method.On the client application's main thread, call the delegate's BeginInvoke method and pass in the necessary data.On a second thread, call the EndInvoke method on the delegate to get the results.
C. Call the LogData method by using the ThreadPool::QueueUserWorkItem method.
D. In the client code, declare a delegate that has the same signature as the LogData method.On the client application's main thread, call the delegate's Invoke method and pass in the necessary data.On a second thread, call the GetObjectData method on the delegate to get the results.

**Answer:** B

**QUESTION:** 239
An application calls a Web method asynchronously by using the following code. (Line numbers

are included for reference only.)

```
01   void ProcessData() {
02   ProcessingService^ serviceProxy = gcnew        ProcessingService();
03   IAsyncResult^ asyncResult = nullptr;
04   asyncResult = serviceProxy->BeginProcess(data, nullptr,        nullptr);
05   while (!asyncResult->IsCompleted) {
06  Thread::Sleep(1000);
07    }
08
09  serviceProxy->EndProcess(asyncResult);
10  }
```

You need to ensure that the application can process and log any exceptions raised by the Web method. What should you do?

A. Replace line 09 with the following code.try { serviceProxy->EndProcess(asyncResult);}catch (Exception^ ex) { LogException(ex);}
B. Replace lines 05, 06, and 07 with the following code.try { while (!asyncResult->IsCompleted) { Thread::Sleep(1000); }}catch (Exception^ ex) { LogException (ex);}
C. Replace line 08 with the following code.if (dynamic_cast<Exception^>(asyncResult->AsyncState) != nullptr) Console::WriteLine (asyncResult->AsyncState->ToString());
D. Replace line 04 with the following code.try { asyncResult = serviceProxy->BeginProcess(data, nullptr, nullptr);}catch (Exception^ ex) { LogException (ex);}

**Answer:** A

**QUESTION:** 240

You are writing an application that calls a Web service. The application must call the Web service asynchronously and also perform a small amount of processing while the Web service is running. The return value from the Web service is required for additional processing. You need to ensure that the application can call the Web service asynchronously and also process the return value. Your solution must keep processor cycles to a minimum. What should you do?

A. Implement the Web service call as follows:ProcessService^ serviceProxy = gcnew ProcessService();IAsyncResult^ asyncResult = serviceProxy->BeginProcess(data, nullptr, nullptr);String^ ret = serviceProxy->EndProcess(asyncResult);PerformProcessing();PerformAdditionalProcessing(ret);
B. Implement the Web service call as follows:ProcessService^ serviceProxy = gcnew ProcessService();IAsyncResult^ asyncResult = serviceProxy->BeginProcess(data, nullptr, nullptr);PerformProcessing();String^ ret =

serviceProxy->EndProcess(asyncResult);PerformAdditionalProcessing(ret);
C. Implement the Web service call as follows:ProcessService^ serviceProxy = gcnew ProcessService();IAsyncResult^ asyncResult = serviceProxy->BeginProcess(data, gcnew AsyncCallback(&Processor::ProcessHandler),serviceProxy);PerformProcessing();PerformAdditionalProcessing(ret);
D. Implement the Web service call as follows:ProcessService^ serviceProxy = gcnew ProcessService();IAsyncResult^ asyncResult = serviceProxy->BeginProcess(data,nullptr,nullptr);PerformProcessing();while (!asyncResult->IsCompleted){
}String^ ret = serviceProxy->EndProcess(asyncResult);PerformAdditionalProcessing(ret);

**Answer:** B