

KTest

更に上のクオリティ 更に上のサービス



問題集

<http://www.ktest.jp>

1年で無料進級することに提供する

Exam : **70-513J**

Title : TS: Windows
Communication Foundation
Development with
Microsoft .NET Framework 4

Version : DEMO

1. あなたは、次のように実装されている Windows Communication Foundation (WCF) サービスを作成している。

(行番号は参照のためだけに含まれています。)

```
01 [ServiceContract]
02 [ServiceBehavior(IncludeExceptionDetailsInFaults = true)]
03 public クラス OrderService
04 {
05     [OperationContract]
06     公共ボイドに submitOrder (注文 anOrder)
07     {
08         トライ
09         {
10             ...
11         }
12         キャッチ (DivideByZeroException EX)
13         {
14             ...
15         }
16     }
17 }
```

あなたは、例外のスタックトレースの詳細がクライアントに送信されるエラー情報に含まれていないことを確認する必要があります。

あなたはどうすればいいのでしょうか？

A. 次の行と 14 行に置き換える：

スロー；

B. 次の行と 14 行に置き換える：

新たな `FaultException` の `<Order>` を `(anOrder, ex.ToString ())` スロー；

C. ライン 05 の後、次の行を追加する：

```
[FaultContract(typeof(FaultException<Order>))]
```

次の行と 14 行に置き換える：

EX を投げる；

D. ライン 05 を変更し、次の行を追加する：

```
[FaultContract(typeof(FaultException<Order>))]
```

次の行と 14 行に置き換える：

新しいの `FaultException` の `<Order>` (`anOrder, "ゼロ除算例外"`) を投げる

Answer: D

2. あなたは、Windows Communication Foundation (WCF) サービスを作成している。あなたは、サービス層で内部の実装を公開したくない。

あなたは、合計という操作で算術という名前のサービスとして以下のクラスを公開する必要がある：

パブリッククラス電卓

```
{
    public int Add(int x, int y)
    {
```

```
}  
}
```

あなたはどちらのコードセグメントを使うべきでしょうか？

A.[ServiceContract(Namespace="Arithmetic")]

```
public class Calculator  
{  
    [OperationContract(Action="Sum")]  
    public int Add(int x, int y)  
    {}  
}
```

B.[ServiceContract(ConfigurationName="Arithmetic")]

```
public class Calculator  
{  
    [OperationContract(Action="Sum")]  
    public int Add(int x, int y)  
    {}  
}
```

C.[ServiceContract(Name="Arithmetic")]

```
public class Calculator  
{  
    [OperationContract(Name="Sum")]  
    public int Add(int x, int y)  
    {}  
}
```

D.[ServiceContract(Name="Arithmetic")]

```
public class Calculator  
{  
    [OperationContract(ReplyAction="Sum")]  
    public int Add(int x, int y)  
    {}  
}
```

Answer: C

3. あなたは、Windows Communication Foundation (WCF) サービスのデータコントラクトを開発しています。

データコントラクトのデータは、往復に参加しなければならない。厳格なスキーマ妥当性は必要がありません。

あなたは、契約は、前方互換性があり、新たなデータメンバがそれに追加することができますことを確認する必要があります。

あなたは、データコントラクトクラスにどのインターフェースを実装する必要がありますか？

- A.ICommunicationObject
- B.IExtension<T>
- C.IExtensibleObject<T>
- D.IExtensibleDataObject

Answer: D

4. Windows Communication Foundation (WCF) アプリケーションでは、いくつかのデータメンバを持つデータコントラクトを使用しています。

あなたは、データコントラクトのシリアル化されたインスタンスが逆シリアル化されるときに、データ・メンバーのいずれかが存在しない場合は、**SerializationException** スローするアプリケーションが必要です。

あなたはどうすればいいのでしょうか？

- A. データ契約に **KnownType** 属性を追加します。
データメンバ宣言のそれぞれのデフォルト値を設定します。
- B. データ契約に **KnownType** 属性を追加します。
ユニークな整数値に各データメンバの **Order** プロパティを設定します。
- C. **false** に各データメンバの **EmitDefaultValue** プロパティを設定します。
- D. **true** に各データメンバの **IsRequired** プロパティを設定します。

Answer: D

5. Windows Communication Foundation (WCF) アプリケーションには、次のデータコントラクトを使用しています。

[DataContract]

公共 Person クラス

```
{
    [DataMember]
    パブリック文字列の firstName;
    [DataMember]
    パブリック文字列 lastName;
    [DataMember]
    公共 int 型の年齢;
    [DataMember]
    公共 int 型の ID;
}
```

あなたは、データコントラクトがシリアル化されるときに次の XML セグメントが生成されることを確認する必要があります。

<Person>

```
<firstName xsi:nil="true"/>
<lastName xsi:nil="true"/>
<ID>999999999</ID>
```

</Person>

あなたはどちらのコードセグメントを使うべきでしょうか？

A.[DataMember]

パブリック文字列の firstName;

[DataMember]

パブリック文字列 lastName;

[DataMember(EmitDefaultValue = true)]

公共 int 型の年齢= 0;

[DataMember(EmitDefaultValue = true)]

公共 int 型の ID =999999999;

B.[DataMember(EmitDefaultValue = false)]

パブリック文字列の firstName= NULL;

[DataMember(EmitDefaultValue = false)]

パブリック文字列 lastName の= NULL;

[DataMember(EmitDefaultValue = true)]

公共 int 型の年齢=-1;

[DataMember(EmitDefaultValue = false)]

公共 int 型の ID =999999999;

C.[DataMember(EmitDefaultValue = true)]

パブリック文字列の firstName;

[DataMember(EmitDefaultValue = true)]

パブリック文字列 lastName;

[DataMember(EmitDefaultValue = false)]

公共 int 型の年齢=-1;

[DataMember(EmitDefaultValue = false)]

公共 int 型の ID =999999999;

D.[DataMember]

パブリック文字列の firstName= NULL;

[DataMember]

パブリック文字列 lastName の= NULL;

[DataMember(EmitDefaultValue = false)]

公共 int 型の年齢= 0;

[DataMember(EmitDefaultValue = false)]

公共 int 型の ID =999999999;

Answer: D

6. 以下は、SOAP エンベロープの例です。

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope">
  <s:Header>
    <h:StoreId xmlns:h="http://www.contoso.com">6495</h:StoreId>
  </s:Header>
  <s:Body>
    <CheckStockRequest xmlns="http://www.contoso.com">
```

```
<ItemId>2469<ItemId>
</CheckStockRequest>
```

```
</s: Body>
```

```
</s:Envelope>
```

あなたは、SOAP エンベロープを生成し、メッセージコントラクトを作成する必要があります。
あなたはどちらのコードセグメントを使うべきでしょうか？

A.[MessageContract(WrapperName="http://www.contoso.com")]

パブリッククラス CheckStockRequest

```
{
    [MessageHeader(Namespace="http://www.contoso.com")]
    公共 int 型 StoreID{ get; set; }
    [MessageBodyMember(Namespace="http://www.contoso.com")]
    公共 int 型 ItemId{ get; set; }
}
```

B.[MessageContract(WrapperNamespace="http://www.contoso.com")]

パブリッククラス CheckStockRequest

```
{
    [MessageHeader(Namespace="http://www.contoso.com")]
    公共 int 型 StoreID{ get; set; }

    [MessageBodyMember(Namespace="http://www contoso.com")]
    公共 int 型 ItemId{ get; set; }
}
```

C.[MessageContract(WrapperNamespace="http://www.contoso.com")]

パブリッククラス CheckStockRequest

```
{
    [MessageHeader(Namespace="http://www.contoso.com")]
    公共 int 型 StoreID{ get; set; }
    公共 int 型 ItemId{ get; set; }
}
```

D.[MessageContract(WrapperNamespace="http://www.contoso.com")]

パブリッククラス CheckStockRequest

```
{
    [MessageHeader(Namespace="http://www.contoso.com")]
    公共 int 型 StoreID{ get; set; }

    [MessageBodyMember]
    公共 int 型 ItemId{ get; set; }
}
```

Answer: D

7. あなたは、Windows Communication Foundation (WCF) には、SOAP メッセージのいくつかのタイプを送信するクライアントを開発しています。

サービスメソッドが `PostData` のという。 `PostData` のは、現在、次のように定義される：

[OperationContract]

無効 `PostData` の (注文データ)；

それは、任意の SOAP メッセージを受信できるように、`PostData` のを変更する必要があります。

あなたはどちらのコードセグメントを使うべきでしょうか？

A.[OperationContract(IsOneWay=true, Action="", ReplyAction="")]

void PostData(Order data);

B.[OperationContract(IsOneWay=true, Action="", ReplyAction = "")]

void PostData(BodyWriter data);

C.[OperationContract]

void PostData(BodyWriter data);

D.[OperationContract]

void PostData(Message data);

Answer: D

8. `TestService` という名前のクラスは、次のインターフェースを実装します。

[ServiceContract]

パブリックインターフェイスの `ITestService`

{

 [OperationContract]

 DateTime GetServiceTime();

}

`TestService` というのは、ASP.NET アプリケーションでホストされています。

あなたは、JSON としてフォーマットされたデータを返すように `GetServiceTime` 方法を可能にするためにアプリケーションを変更する必要があります。

リクエスト URL は、`/ServiceTime` で終わる場合のみ、これを行う必要があります。

あなたはどうすればいいのでしょうか？

A. `GetServiceTime` メソッドにこの属性を追加します。

[WebInvoke(Method="POST")]

Web.config ファイルで、の `system.serviceModel/行動/ endpointBehaviors` にこの要素を追加します。

```
<behavior name="Json">
```

```
    <enableWebScript />
```

```
</behavior>
```

web.config ファイルで、次のように `system.serviceModel/サービスコレクション` に `TestService` というを設定する：

```
<service name="TestService">
```

```
    <endpoint address="/ServiceTime"
```

```
        contract="TestService"
```

```
        behaviorConfiguration="Json"
```

```
        binding="webHttpBinding" />
```

```
</service>
```


B. `GetServiceTime` メソッドにこの属性を追加します。

`[WebInvoke(Method="GET", UriTemplate="/ServiceTime", ResponseFormat=WebMessageFormat.Json)]`
`web.config` ファイルで、次のように `system.serviceModel/` サービスコレクションに `TestService` というを設定します。

```
<service name="TestService">
  <endpoint address="/ServiceTime"
    contract="TestService"
    binding="webHttpBinding"/>
</service>
```

C. `GetServiceTime` メソッドにこの属性を追加します。

`[WebGet(ResponseFormat=WebMessageFormat.Json, UriTemplate="/ServiceTime")]`

以下の内容を `Jsonversion.svc` という名前の新しい `SVC` ファイルを作成します。

```
<% @ServiceHost Service="TestService"
Factory="System.ServiceModel.ActivationWebServiceHostFactory" %>
```

D. `GetServiceTime` メソッドにこの属性を追加します。

`[WebGet(UriTemplate="Json)/ServiceTime"]`

以下の内容を `Jsonversion.svc` という新しい `SVC` ファイルを作成します。

```
<% @ServiceHost Service="TestService"
Factory="System.ServiceModel.ActivationWebServiceHostFactory" %>
```

Answer: C

9. あなたは、RESTful な方法で操作を実装 Windows Communication Foundation (WCF) サービスを作成しています。

あなたは、削除操作を追加する必要があります。次のように `delete` メソッドを実装します。

ポイド `DeleteItems` (列番号) ;

あなたは、クライアントが `HTTP DELETE` 操作でサービスを呼び出すときに、このメソッドを呼び出すように `WCF` を設定する必要があります。

あなたはどうすればいいのでしょうか？

- A. 動作に `WebInvoke` を追加 (`UriTemplate="/Items/{id}", Method="DELETE"`) 属性です。
- B. 動作に `HttpDelete` の `attribute` を追加します。
- C. `RemovedActivityAction` パラメータを使用して文字列パラメータを置き換えます。
- D. `RemovedActivityAction` と戻り値の型を交換してください。

Answer: A

10. Windows Communication Foundation (WCF) サービスは、次のサービスコントラクトを使用します。

`[ServiceContract]`

パブリックインターフェイスの `IService`

```
{
  [OperationContract]
  文字列操作 1 (文字列 s);
}
```

あなたは、操作 1 操作コントラクトは、`HTTP POST` 要求に応答することを確認する必要があります。

あなたはどちらのコードセグメントを使うべきでしょうか？

A. `[OperationContract]`

[WebInvoke(Method="POST")]

文字列操作 1 (文字列 s);

B.[OperationContract]

[WebGet(UriTemplate="POST")]

文字列操作 1 (文字列 s);

C.[OperationContract(ReplyAction="POST")]

文字列操作 1 (文字列 s);

D.[OperationContract(Action="POST")]

文字列操作 1 (文字列 s);

Answer: A

11. Windows Communication Foundation (WCF) サービスは、一方向および要求応答操作でコントラクトを実装します。

サービスは、TCP トランスポートを介して公開されています。クライアントは、サービスと通信するようにルータを使用しています。

ルータは、以下のように実現される。(行番号は参照のためだけに含まれています。)

```
01 ServiceHost host = new ServiceHost(typeof(RoutingService));
02 host.AddServiceEndpoint(
03     typeof(ISimplexDatagramRouter),
04     new NetTcpBinding(), "net.tcp://localhost/Router"
05 );
06 List<ServiceEndpoint> lep = new List<ServiceEndpoint>();
07 lep.Add(
08     new ServiceEndpoint(
09         ContractDescription.GetContract(
10             typeof(ISimplexDatagramRouter)
11         ),
12     new NetTcpBinding(),
13     new EndpointAddress("net.tcp://localhost:8080/Logger")
14 );
15 );
```

```
16 RoutingConfiguration rc = new RoutingConfiguration();
```

```
17 rc.FilterTable.Add(new MatchAllMessageFilter(), lep);
```

```
18 host.Description.Behaviors.Add(new RoutingBehavior(rc));
```

要求応答操作が失敗している。あなたは、ルータは、一方向、要求 - 応答操作を処理できることを確認する必要があります。

あなたはどうすればいいのでしょうか？

A. 次のように行 03 に変更します。

typeof 演算 (IRequestReplyRouter)

B. 次のように行 03 に変更します。

typeof 演算 (IDuplexSessionRouter)

C. 次のように 10 行目を変更します。

typeof 演算 (IRequestReplyRouter)

D. 次のように 10 行目を変更します。

typeof 演算 (IDuplexSessionRouter)

Answer: B

12. あなたは、次のように定義されている既存の Windows Communication Foundation (WCF) サービスを変更している:

[ServiceContract]

パブリックインターフェイスの IMessageProcessor

```
{
    [OperationContract]
    ProcessMessages () を無効;
}
```

パブリッククラス MessageProcessor: IMessageProcessor

```
{
    公共空間 ProcessMessage は ();
    SubmitOrder();
}
```

submitOrder は、別のサービスを呼び出します。重い負荷の下で予想通り ProcessMessage メソッドは実行されません。

あなたは、複数のメッセージの処理を有効にする必要があります。 ProcessMessage メソッドが要求を処理していないときに新しいメッセージのみ、処理されなければなりません。

またはコールが返すことに submitOrder することが待っている時です。

あなたは MessageProcessor クラスにどの属性を適用する必要がありますか?

- A.CallbackBehavior(ConcurrencyMode=ConcurrencyMode.Reentrant)
- B.CallbackBehavior(ConcurrencyMode=ConcurrencyMode.Multiple)
- C.ServiceBehavior(ConcurrencyMode=ConcurrencyMode.Reentrant)
- D.ServiceBehavior(ConcurrencyMode=ConcurrencyMode.Multiple)

Answer: C

13. Windows Communication Foundation (WCF) サービスは、net.tcp の:// www.contoso.com/ MyService でメッセージをリッスンします。

それは http://www.contoso.com/MyService で論理アドレスを持っています。次のように WCF クライアントのコンフィギュレーションは、次のとおりです。

```
<endpoint address="http://www.contoso.com/MyService"
    binding="netTcpBinding"
    bindingConfiguraton="NetTcpBinding_IMyService"
    contract="ServiceReference1.IMyService"
    name="NetTcpBinding_IMyService"/>
```

生成された構成では、サーバーと通信するためのクライアントのために十分な情報を提供していません。それがサーバーと通信できるように、クライアントを更新する必要があります。

あなたはどうすればいいのでしょうか?

- A. クライアント構成では、アドレスに属性の値を変更する net.tcp の:// www.contoso.com/ MyService です。
- B. クライアント構成では、アドレスに属性の値を変更する net.tcp の:// www.contoso.com/ MyService では = http://www.contoso.com/MyService を聞きます。

C. クライアントをインスタンス化した後、任意のサービス操作を呼び出す前に、このコード行を追加します。

```
EndpointBehaviors.Add (新しい EndpointDiscoveryBehavior () {enabled = true の場合は});
```

D. クライアントをインスタンス化した後、任意のサービス操作を呼び出す前に、このコード行を追加します。

```
client.Endpoint.Behaviors.Add (新しい ClientViaBehavior (新しい Uri ("net.tcp の:// www.contoso.com/ IMyService")));
```

Answer: D

14. Windows Communication Foundation (WCF) サービスでは、コンソールアプリケーションで自己ホストされています。

サービスは、MyApplication という名前空間で定義されている IDataAccess 契約を、実装します。

サービスは IDataAccess を実装するクラスの名前を DataAccessService に実装されています

また、インターフェースとは、MyApplication という名前空間で定義されています。次のようにホスティングコードです。(行番号は参照のためだけに含まれています。)

01 静的な無効メイン (文字列[] args)

```
02 {
03     ServiceHost のホスト;
04     ...
05     host.Open();
06     Console.ReadLine();
07     host.Close();
08 }
```

あなたは、ServiceHost のインスタンスを作成し、ホスト変数に割り当てる必要があります。また、サービスホストをインスタンス化する必要があります。

どのコード行は、行 04 に挿入する必要がありますか?

A.host = new ServiceHost("MyApplication.DataAccessService");

B.host = new ServiceHost("MyApplication.DataAccess");

C.host = new ServiceHost(typeof(IDataAccess));

D.host = new ServiceHost(typeof(DataAccessService));

Answer: D

15. Windows Communication Foundation (WCF) サービスには、次のコントラクトを実装します。

[ServiceContract]

パブリックインターフェースの IHelloService

```
{
    [OperationContract(WebGet(UriTemplate="hello?name={name}"))]
    文字列 SayHello (文字列名);
}
```

次のように実装は、次のとおりです。

パブリッククラス HelloService の: IHelloService

```
{
    パブリック文字列 SayHello (文字列名)
```

```

    {
        "こんにちは" +名前を返す;
    }
}

```

サービスは自己ホスト型であり、次のようにホストコードは、次のとおりです。

```

WebServiceHost svcHost = CreateHost();
svcHost.Open();
Console.ReadLine();
svcHost.Close();

```

あなたは、サービスが `http://localhost:8000/HelloService` でホスト単一のエンドポイントを持っているように `CreateHost` を実装する必要があります。

あなたはどちらのコードセグメントを使うべきでしょうか？

A. `WebServiceHost` は `Svchost` で=新しい `WebServiceHost` は (`typeof` 演算 (`HelloService` の));

```

svcHost.AddServiceEndpoint (typeof 演算 (IHelloService)
    新しい WebHttpBinding を (WebHttpSecurityMode.None)
    "http://localhost:8000/HelloService");

```

`Svchost` を返す;

B. `Uri` ベースアドレス=新しい `Uri` ("http://localhost:8000 と");

`WebServiceHost` は `Svchost` で=新しい `WebServiceHost` は (`typeof` 演算 (`HelloService` の)、ベースアドレス);

```

svcHost.AddServiceEndpoint (typeof 演算 (IHelloService)
    新しい WebHttpBinding を (WebHttpSecurityMode.None)
    "HelloService");

```

`Svchost` を返す;

C. `WebServiceHost` は `Svchost` で=新しい `WebServiceHost` は (新しい `HelloService` の ());

```

svcHost.AddServiceEndpoint (typeof 演算 (IHelloService)
    新しい WebHttpBinding を (WebHttpSecurityMode.None)
    "http://localhost:8000/HelloService");

```

`returnn svcHost`

D. `Uri` `baseAddress` = new `Uri`("http://localhost:8000/");

`WebServiceHost` `svcHost` = new `WebServiceHost`(new `HelloService`(), `baseAddress`);

```

svcHost.AddServiceEndpoint(typeof(IHelloService),
    new WebHttpBinding(WebHttpSecurityMode.None),
    "HelloService");

```

`returnn svcHost`;

Answer: B

16. あなたは、Windows Communication Foundation (WCF) サービスのクライアントを構築しています。あなたはこのサービスを利用するためにプロキシを作成する必要があります。あなたはどのクラスを使うべきでしょうか？

A. `ChannelFactory<TChannel>`

B. `ServiceHost`

C. `ClientRuntime`

D. `CommunicationObject`

Answer: A

17. あなたが `SampleServiceProxy` という名前で生成されたプロキシを持つ `Windows Communication Foundation (WCF)` クライアントアプリケーションで作業しています。

クライアントアプリケーションが実行されている場合は、次のコードの行 **04** で、チャンネルの障害は（行番号は参照のためだけに含まれています。）

```
01 SampleServiceProxy proxy = new SampleServiceProxy();
02 try
03 {
04     proxy.ProcessInvoice(invoice);
05 }
06 catch
07 {
08     if(proxy.State == CommunicationState.Faulted)
09     {
10         ...
11     }
12 }
13 proxy.UpdateCustomer(customer);
```

あなたはそれが成功した **13** 行目で呼び出しを実行できる状態へのプロキシを返す必要があります。

あなたは、**10** 行目で、どのコード・セグメントを使うべきでしょうか？

- A. `proxy.Close();`
- B. `proxy = new SampleServiceProxy();`
- C. `proxy.Abort();`
- D. `proxy.Open();`

Answer: B

18. `Windows Communication Foundation (WCF)` サービスは、コールバックコントラクトを持っています。このサービス呼び出しますクライアントアプリケーションを開発しています。

あなたは、クライアントアプリケーションが `WCF` サービスと対話できるようにする必要があります。

あなたはどうすればいいのでしょうか？

- A. `OperationContractAttribute` で、`true` に `AsyncPattern` プロパティ値を設定します。
- B. `OperationContractAttribute` では、クライアントのエンドポイントアドレスに `ReplyAction` プロパティ値を設定します。
- C. クライアントでは、`DuplexClientBase` の `<TChannel>` 由来プロキシを作成します。
- D. クライアントでは、`GetCallbackChannel<T>` を使用しています。

Answer: C

19. あなたは、`Windows Communication Foundation (WCF)` サービスを作成している。あなたは、次の要件がある：

メッセージは、`TCP` 経由で送信する必要がある

サービスは、トランザクションをサポートしている必要があります。

メッセージは、バイナリエンコーディングを使ってエンコードされている必要があります。

メッセージは、`Windows` のストリームベースのセキュリティを使用して保護されなければなりません。

あなたがサービスのためにカスタムバインディングを実装する必要があります。順序はバインディングスタックするを構成する必要がありますか？

- A.tcpTransport, windowsStreamSecurity, transactionFlow, binaryMessageEncoding
- B.transactionFlow, binaryMessageEncoding, windowsStreamSecurity, tcpTransport
- C.windowsStreamSecurity, tcpTransport, binaryMessageEncoding, transactionFlow
- D.binaryMessageEncoding, transactionFlow, tcpTransport, windowsStreamSecurity

Answer: B

20. Windows Communication Foundation (WCF) クライアント構成ファイルは、の system.serviceModel 要素に次の XML セグメントを含む。

```
<client>
  <endpoint address="net.tcp://server/ContosoService"
    binding="netTcpBinding"
    contract="Contoso.IContosoService"
    name="netTcp"/>
  <endpoint address="net.pipe://localhost/ContosoService"
    binding="netNamedPipeBinding"
    contract="Contoso.IContosoService"
    name="netPipe" />
</client>
```

あなたは、net.pipe のの:// localhost の/ ContosoService でリスニング・エンドポイントにメッセージを送ることができるチャネルファクトリを作成する必要があります。

あなたはどちらのコードセグメントを使うべきでしょうか？

- A.ChannelFactory<Contoso.IContoso> factory = new ChannelFactory<Contoso.IContoso>("Contoso.IContoso");
- B.ChannelFactory<Contoso.IContoso> factory = new ChannelFactory<Contoso.IContoso>("netNamedPipeBinding");
- C.ChannelFactory<Contoso.IContoso> factory = new ChannelFactory<Contoso.IContoso>("netPipe");
- D.ChannelFactory<Contoso.IContoso> factory = new ChannelFactory<Contoso.IContoso>("net.pipe//localhost/ContosoService");

Answer: C