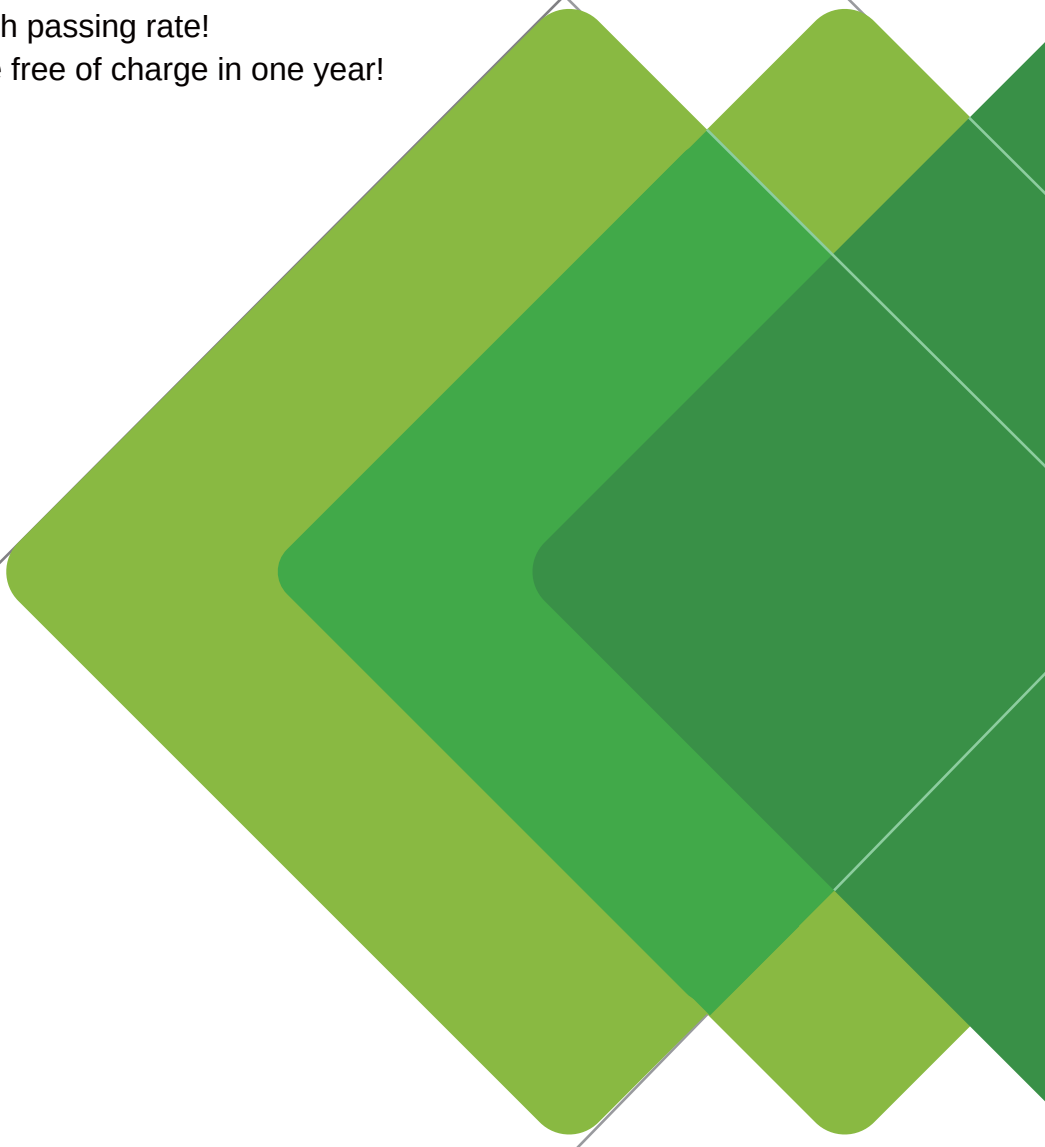
IT-DUMPS Q&A

Accurate study guides, High passing rate!
IT-dumps provides update free of charge in one year!



Exam : **70-511-Csharp**

Title : MCTS: Windows Applications Development with Microsoft .NET Framework 4 Practice Test

Version : Demo

1.You are developing a Windows Presentation Foundation (WPF) application. You need to use XAML to create a custom control that contains two Button controls. From which base class should you inherit?
A. FrameworkElement
B. UIElement
C. UserControl
D. Button
Answer: C

2.You use Microsoft .NET Framework 4 to create a Windows Presentation Foundation (WPF) application. The application has multiple data entry windows. Each window contains controls that allow the user to type different addresses for shipping and mailing. All addresses have the same format. You need to ensure that you can reuse the controls. What should you create?
A. a user control
B. a data template
C. a control template
D. a control that inherits the Canvas class
Answer: A

3.You are developing a Windows Presentation Foundation (WPF) application that displays financial data. The following style is applied to every Label control that displays currency. (Line numbers are included for reference only.)

```
01  <Style x:Key="CurrencyLabel"
02   BasedOn="{StaticResource {x:Type Label}}"
03   TargetType="{x:Type Label}">
04    <Setter Property="Template">
05      <Setter.Value>
06
07      </Setter.Value>
08    </Setter>
09  </Style>
```

You need to ensure that the style is updated to meet the following requirements regarding currency:
@It must be right-aligned.
@It must display the number with the regional currency settings.
Which markup segment should you insert at line 06?
A. <ControlTemplate TargetType="{x:Type Label}"> <ContentPresenter HorizontalAlignment="Right" ContentStringFormat="{}{0:C}" /></ControlTemplate>
B. <ControlTemplate> <ContentPresenter HorizontalAlignment="Right" ContentStringFormat="{}{0:C}" /></ControlTemplate>
C. <ControlTemplate TargetType="{x:Type Label}"> <Label HorizontalAlignment="Right" Content="{Binding StringFormat={}{0:C}}"/></ControlTemplate>
D. <ControlTemplate> <Label HorizontalAlignment="Right" Content="{Binding StringFormat={}{0:C}}"/></ControlTemplate>
Answer: A

4.You use Microsoft .NET Framework 4 to create a Windows Presentation Foundation (WPF) application.

You want to add an audio player that plays .wav or .mp3 files when the user clicks a button. You plan to store the name of the file to a variable named SoundFilePath. You need to ensure that when a user clicks the button, the file provided by SoundFilePath plays. What should you do?
A. Write the following code segment in the button onclick event. System.Media.SoundPlayer player = new System.Media.SoundPlayer(SoundFilePath);player.Play();
B. Write the following code segment in the button onclick event. MediaPlayer player = new MediaPlayer();player.Open(new URI(SoundFilePath), UriKind.Relative));player.Play();
C. Use the following code segment from the PlaySound() Win32 API function and call the PlaySound function in the button onclick event. [sysimport(dll="winmm.dll")]public static extern long PlaySound(String SoundFilePath, long hModule, long dwFlags);
D. Reference the Microsoft.DirectX Dynamic Link Libraries. Use the following code segment in the button onclick event. Audio song = new Song(SoundFilePath);song.CurrentPosition = song.Duration;song.Play();
Answer: B

5.You use Microsoft .NET Framework 4 to create a Windows Presentation Foundation (WPF) application. You add a custom command as a resource. The key of the command is saveCommand.
You write the following code fragment. (Line numbers are included for reference only.)

```
01  <Canvas>
02
03  <Button>
04
05  </Button>
06  </Canvas>
```

You need to ensure that saveCommand is executed when the user clicks the Button control.
What should you do?
A. Insert the following code fragment at line 04.
<Button.Command>
<StaticResource ResourceKey="saveCommand" />
</Button.Command>
B. Insert the following code fragment at line 04.
<Button.CommandBindings>
<CommandBinding Command="{StaticResource saveCommand}" />
</Button.CommandBindings>
C. Insert the following code fragment at line 02.
<Canvas.CommandBindings>
<CommandBinding Command="{StaticResource saveCommand}" />
</Canvas.CommandBindings>
Replace line 03 with the following code fragment. <Button CommandTarget="{Binding RelativeSource={RelativeSource Self}, Path=Parent}">
D. Insert the following code fragment at line 02.
<Canvas.CommandBindings>
<CommandBinding Command="{StaticResource saveCommand}" />
</Canvas.CommandBindings> Replace line 03 with the following code fragment
<Button CommandParameter="{Binding RelativeSource={RelativeSource Self}, Path=Parent}">

Answer: A