

# Easy CramBible Lab



**70-511**

**TS: Windows Applications  
Development with  
Microsoft .NET Framework 4**

**\*\* Single-user License \*\***

This copy can be only used by yourself for educational purposes.

**Web:** <http://www.crambible.com/>

**E-mail:** [web@crambible.com](mailto:web@crambible.com)

**Important Note****Please Read Carefully****Study Tips**

This product will provide you questions and answers along with carefully compiled and written by our experts. Try to understand the concepts behind the questions instead of cramming the questions.

Go through the entire document at least twice so that you make sure that you are not missing anything.

**Latest Version**

We are constantly reviewing our products. New material is added and old material is revised. Free updates are available for 90 days after the purchase. You should check your member zone at CramBible an update 3-4 days before the scheduled exam date.

Here is the procedure to get the latest version:

1. Go to [www.CramBible.com](http://www.CramBible.com)
2. Click on Member zone/Log in
3. The latest versions of all purchased products are download from here. Just click the links.

For most updates, it is enough just to print the new questions at the end of the new version, not the whole document.

**Feedback**

Feedback on specific questions should be send to [web@CramBible.com](mailto:web@CramBible.com). You should state: Exam number and version, question number, and login ID.

Our experts will answer your mail promptly.

**Copyright**

Each pdf file contains a unique serial number associated with your particular name and contact information for security purposes. So if we find out that a particular pdf file is being distributed by you, CramBible reserves the right to take legal action against you according to the International Copyright Laws.

## Exam A

### QUESTION 1

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

You write the following code fragment.

```
<StackPanel TextBox.PreviewTextInput="StackPanel_PreviewTextInput">
<TextBox Name="TxtBoxA"/>
<TextBox Name="TxtBoxB"/>
<TextBox Name="TxtBoxC"/>
</StackPanel>
```

You create an event handler named StackPanel\_PreviewTextInput. You also have a collection of strings named Keywords.

You need to ensure that TxtBoxA and TxtBoxB do not contain any of the strings in the Keywords collections.

Which code segment should you use?

- A. 

```
private void StackPanel_PreviewTextInput( object sender, TextCompositionEventArgs e)
{ FrameworkElement feSource = sender as FrameworkElement;
  if (feSource.Name == "TxtBoxA" || feSource.Name == "TxtBoxB")
  { foreach(string keyword in Keywords)
    {
      if(e.Text.Contains(keyword)) {
        e.Handled = false;
        return;
      }
    }
  } e.Handled = true;
}
```
- B. 

```
private void StackPanel_PreviewTextInput( object sender, TextCompositionEventArgs e) {
  FrameworkElement feSource = e.Source as FrameworkElement;
  f (feSource.Name == "TxtBoxA" || feSource.Name == "TxtBoxB")
  f (feSource.Name == "TxtBoxA" || feSource.Name == "TxtBoxB") {
  foreach(string keyword in Keywords)
  {
    if(e.Text.Contains(keyword)) {
      e.Handled = false;
      return;
    }
  } e.Handled = true;
}
```
- C. 

```
private void StackPanel_PreviewTextInput( object sender, TextCompositionEventArgs e)
{
  FrameworkElement feSource = sender as FrameworkElement;
  if (feSource.Name == "TxtBoxA" || feSource.Name == "TxtBoxB")
  { foreach(string keyword in Keywords)
    { if(e.Text.Contains(keyword)) {
      e.Handled = true;
      return; }
    } e.Handled = false;
  }
}
```

```

D. private void StackPanel_PreviewTextInput( object sender, TextCompositionEventArgs e)
    { FrameworkElement feSource = e.Source as FrameworkElement;
      if (feSource.Name == "TextBoxA" || feSource.Name == "TextBoxB")
      {
        foreach(string keyword in Keywords)
        { if(e.Text.Contains(keyword)) {
          e.Handled = true;
          return;
        } } e.Handled = false;
      }
    }
}

```

**Answer: D**

### QUESTION 2

You use Microsoft .NET Framework 4 to create a Windows Presentation Foundation (WPF) application. The application contains a composite user control that includes a TextBox control named txtInput. The user control will be hosted in a window and will have handlers for the text-changed event of txtInput. You need to ensure that the application meets the following requirements:

- Creates a text-changed event handler named Audit\_TextChanged for the txtInput control.
- Executes Audit\_TextChanged even when specific handlers mark the event as handled.

Which code segment should you add to the constructor of the user control

- A. txtInput.TextChanged+=Audit\_TextChanged;
- B. AddHandler(TextBox.TextChangedEvent, new RoutedEventHandler(Audit\_TextChanged), true);
- C. EventManager.RegisterClassHandler(typeof(TextBox),TextBox.TextChangedEvent, new RoutedEventHandler(Audit\_TextChanged), true);
- D. EventManager.RegisterClassHandler(typeof(TextBox),TextBox.TextChangedEvent, new RoutedEventHandler (Audit\_TextChanged), false);

**Answer: B**

### QUESTION 3

You use Microsoft .NET Framework 4 to create a Windows Presentation Foundation (WPF) application. The application contains a composite user control that includes a TextBox control named txtInput. The user control will be hosted in a window and will have handlers for the text-changed event of txtInput. You need to ensure that the application meets the following requirements:

```
AddHandler(TextBox.TextChangedEvent, new RoutedEventHandler(Audit_TextChanged), true);
```

Which of the following statements are **TRUE** ? (choose all that apply)

- A. A text-changed event handler, named Audit\_TextChanged, was Created for the txtInput control.
- B. Audit\_TextChanged will stop running because the event is marked as handled by certain event handlers.
- C. Even through the event is marked handled by certain event handlers, Audit\_TextChanged will still run.
- D. Audit\_TextChanged will continue to run until the event is marked as handled.

**Answer: AC**

### QUESTION 4

You use Microsoft .NET Framework 4 to create a Windows Presentation Foundation (WPF) application. You create a window that contains a Button control and a MenuItem control. Both controls are labeled "Add sugar."

The Command properties of the Button and MenuItem controls are set to the same RoutedCommand named AddSugarCommand.

You write the following code segment.

```
private void CanAddSugar (object sender, CanExecuteRoutedEventArgs e) { ... }
```

You need to ensure that when the **CanAddSugar** method sets **e.CanExecute to false**, the MenuItem and Button controls are **disabled**.

What should you do?

- A. **Create** an event handler for the CanExecuteChanged event of the AddSugarCommand command.  
**Call** the CanAddSugar method from within the event handler.
- B. **Inherit** the AddSugarCommand from the RoutedUICommand class instead of the RoutedCommand class.  
**Call** the CanAddSugar method from within the constructor of the AddSugarCommand command.
- C. **Add** a CommandBinding object to the CommandBinding property of the MenuItem control.  
**Set** the CanExecute property of the CommandBinding object to the CanAddSugar method.
- D. **Add** a CommandBinding object to the CommandBindings property of the window.  
**Set** the Command property of CommandBinding to the AddSugarCommand command.  
**Set** the CanExecute property of the CommandBinding object to the CanAddSugar method.

**Answer: D**

#### QUESTION 5

You use Microsoft .NET Framework 4 to create a Windows Presentation Foundation (WPF) application. The application has a window named MainWindow that has a **StackPanel** control named **sp** as the root element. You want to create a Button control that contains a TextBlock control with the **"Save"** Text property.

You need to create the control **dynamically** and **add** the control to **sp**.

Which code segment should you write in the constructor of the MainWindow class?

- A. Button btn = new Button();  
TextBlock text = new TextBlock();  
text.Text = "Save";  
**btn.Content = text;**  
**sp.DataContext = btn;**
- B. Button btn = new Button();  
TextBlock text = new TextBlock();  
text.Text = "Save";  
**btn.Content = text;**  
**sp.Children.Add(btn);**
- C. Button btn = new Button();  
TextBlock text = new TextBlock();  
text.Text = "Save";  
**sp.Children.Add(btn);**  
**sp.Children.Add(text);**
- D. Button btn = new Button();  
TextBlock text = new TextBlock();  
text.Text = "Save";  
**btn.ContentTemplateSelector.SelectTemplate(text, null);**  
**sp.Children.Add(btn);**

**Answer: B**

#### QUESTION 6

You create a Windows client application by using Windows Presentation Foundation (WPF). The application contains the following code fragment.

```
<Window.Resources>
```

```

<DataTemplate x:Key="detail">
<!--...-->
</DataTemplate>
</Window.Resources>
<StackPanel>
<ListBox Name="lbDetails">
</ListBox>
<Button Name="btnDetails">Details</Button>
</StackPanel>

```

You need to assign lbDetails to use the detail data template when btnDetails is clicked. Which code segment should you write for the click event handler for btnDetails?

- A. lbDetails.ItemsPanel.FindName("detail",lbDetails);
- B. **var tmpl = (ControlTemplate)FindResource("detail");** lbDetails.Template = tmpl;
- C. **var tmpl = (DataTemplate)FindName("detail");** lbDetails.ItemTemplate = tmpl;
- D. **var tmpl = (DataTemplate)FindResource("detail");** lbDetails.ItemTemplate=tmpl;

**Answer: D**

### QUESTION 7

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**

### QUESTION 8

You use Microsoft .NET Framework 4 to create a Windows Presentation Foundation (WPF) application. You write the following code fragment.

```

<StackPanel>
<StackPanel.Resources>
<Style TargetType="{x:Type Button}">
<EventSetter Event="Click" Handler="ButtonHandler"/>
</Style>
</StackPanel.Resources>
<Button Name="OkButton">Ok</Button>
<Button Name="CancelButton" Click="CancelClicked">Cancel</Button>

```