Microsoft

Exam 70-491

Recertification for MCSD: Windows Store Apps using C#

Version: 6.0

[Total Questions: 91]

Topic break down

Topic	No. of Questions
Topic 1: Scenario 1	3
Topic 2: Scenario 2	3
Topic 3: Scenario 3	3
Topic 4: Scenario 4	4
Topic 5: Scenario 5	3
Topic 6: Scenario 6	4
Topic 7: Scenario 7	3
Topic 8: Scenario 8	3
Topic 9: Mix Questions	65

Topic 1, Scenario 1

Overview

Fabrikam, Inc. is a non-profit organization that manages three museums located in Miami, New York, and Seattle. All of the museums offer Wi-Fi connectivity and Internet access to visitors.

Existing Environment

General Information

Fabrikam provides visitors with two pamphlets as they enter each museum. One pamphlet contains pictures of the different paintings in the museum. The other pamphlet contains pictures of the sculptures in the museum.
Visitors are encouraged to take pictures of the sculptures and the paintings.

Each museum has a kiosk that provides information to visitors about the exhibits. The kiosk uses a data access component that only runs on an x86 processor.

Requirements

Business Goals

Fabrikam plans to provide a more interactive experience for the visitors.
Fabrikam purchases 200 Windows 8.1 RT devices for each museum. Fabrikam plans to develop an app to replace the paper pamphlets.
Fabrikam plans to minimize development effort and reuse the data access component, if possible.

General Requirements

Fabrikam identifies the following requirements for the app:

- The app must be available from the Windows App store.
- The app must be available to devices that run Windows 8.1 and Windows 8.1 RT.
- If a user switches to a different app, the new app must enter a Not Running state after 10 seconds.
- The app must provide users with the ability to share pictures with other apps.
- Users must be able to search for paintings and sculptures by name from within the app.
- When users type in search terms, the app must present users with a suggested list of painting and sculpture names.

Page Requirements

The app must have four pages: a main page, a group detail page, an item detail page, and a capture photo page.

Main Page:

- The main page must display grouped items.
- Once the users tap on a group on the main page, the app must open the group detail page.
- The main page must display all of the items for a selected group.

Group Detail Page:

- The group detail page must have two groups, named Paintings and Sculptures, and must display a list of the paintings and sculptures in the museum with the name and a small image of the item.
- The group detail page must display a list of all the items in the group. The list must contain the image and the name of the item.
- The app must have a second view of the group details that displays the name, a description, and an image of each item.
- Users must be able to use the mouse wheel or pinch gestures to move between the two views of the group detail page.

Item Detail Page:

- The item detail page must display the name, a full description, and a large image of the item.
- When the user taps the image of an item on the item detail page, an element named FoundNotFoundFlyout must be displayed. The FoundNotFoundFlyout element will be declared in the Resources section of the page.
- As an alternative to tapping an image on the item detail page, users must be able to use a check gesture to mark the item as found.

Capture Photo Page:

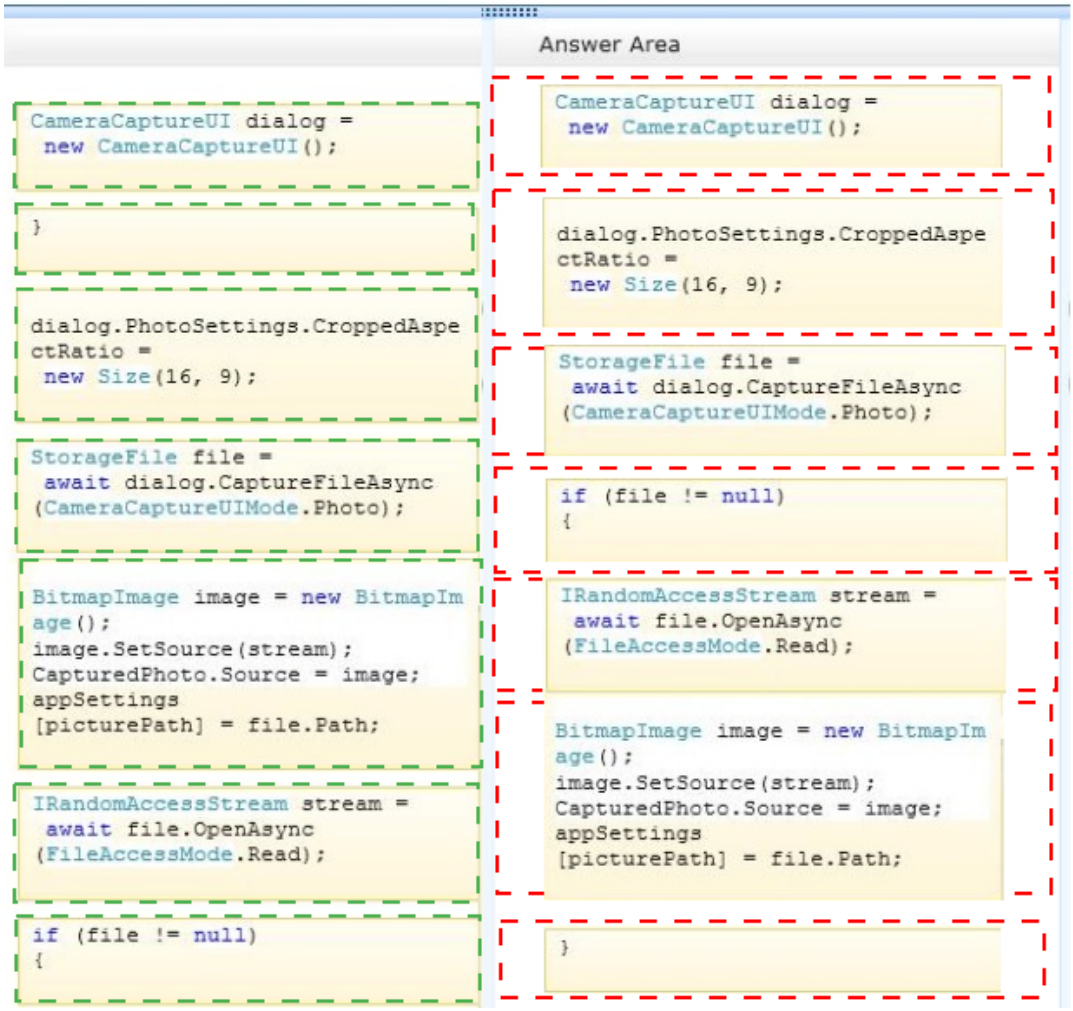
- A page named CapturePhoto will be created to capture and display pictures.
- When a picture is taken, its path must be saved in an application setting property named picturePath.
- Pictures must have an aspect ratio of 16 by 9.
- As new pictures are taken, the app must update the app tile to show the current number of pictures taken.
- A method named UpdatePictureCount will be called any time a new picture is saved. The method will take an integer parameter named pictureCount. The method will use NotificationExtensions library to handle updates.
- The tile will have a text block named outputText.

Question No : 1 DRAG DROP - (Topic 1)

You need to write code for the method that will be called when a user takes a picture. (Develop the solution by arranging the code snippets. You will need all of the code snippets.)


Answer Area
CameraCaptureUI dialog =
new CameraCaptureUI();
}
dialog.PhotoSettings.CroppedAspe
ctRatio =
new Size(16, 9);
StorageFile file =
await dialog.CaptureFileAsync
(CameraCaptureUIMode.Photo);
BitmapImage image = new BitmapIm
age();
image.SetSource(stream);
CapturedPhoto.Source = image;
appSettings
[picturePath] = file.Path;
IRandomAccessStream stream =
await file.OpenAsync
(FileAccessMode.Read);
if (file != null)
{

Answer:



Question No : 2 DRAG DROP - (Topic 1)

You need to recommend a solution to share images from the capture photo page.

You have the following code. (Line numbers are included for reference only.)

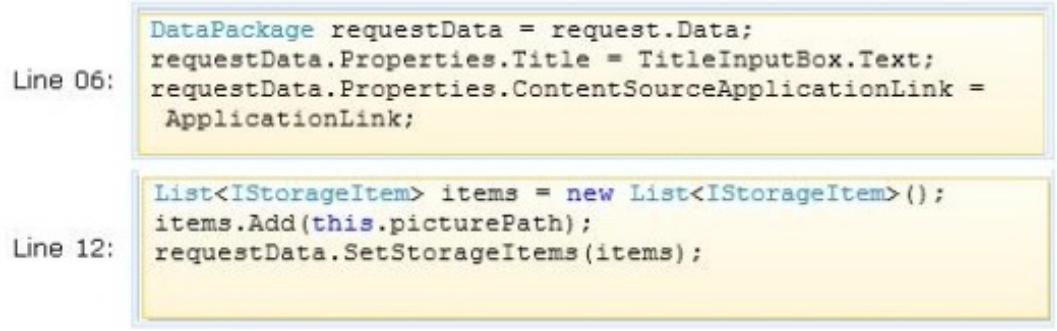
```
01 protected override bool GetShareContent(DataRequest request)
02 {
03     bool succeeded = false;
04     if (this.picturePath != null)
05     {
06
07       RandomAccessStreamReference imageStream =
08        RandomAccessStreamReference.CreateFromFile(this.picturePath);
09         requestData.Properties.Thumbnail = imageStream;
10         requestData.SetBitmap(imageStream);
11         succeeded = true;
12
13     }
14     else
15     {
16         request.FailWithDisplayText(
17          "Select an image you would like to share and try again.");
18     }
19     return succeeded;
20 }
```

Which code segments should you recommend inserting at lines 06 and 12? (To answer, drag the appropriate code segments to the correct locations. Each code segment may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.)


Code Segments
List<IStorageItem> items = new List<IStorageItem>();
items.Add(this.picturePath);
requestData.SetDataProvider(items);
List<IStorageItem> items = new List<IStorageItem>();
items.Add(this.picturePath);
requestData.SetStorageItems(items);
DataPackage requestData = request.Source;
requestData.Properties.Title = TitleInputBox.Text;
requestData.Properties.ContentSourceApplicationLink =
ApplicationLink;
DataPackage requestData = request.Data;
requestData.Properties.Title = TitleInputBox.Text;
requestData.Properties.ContentSourceApplicationLink =
ApplicationLink;
Answer Area
Line 06:
Code segment
Line 12:
Code segment

Answer:

Explanation:



Note: •

Scenario: The app must provide users with the ability to share pictures with other apps.

Question No : 3 HOTSPOT - (Topic 1)

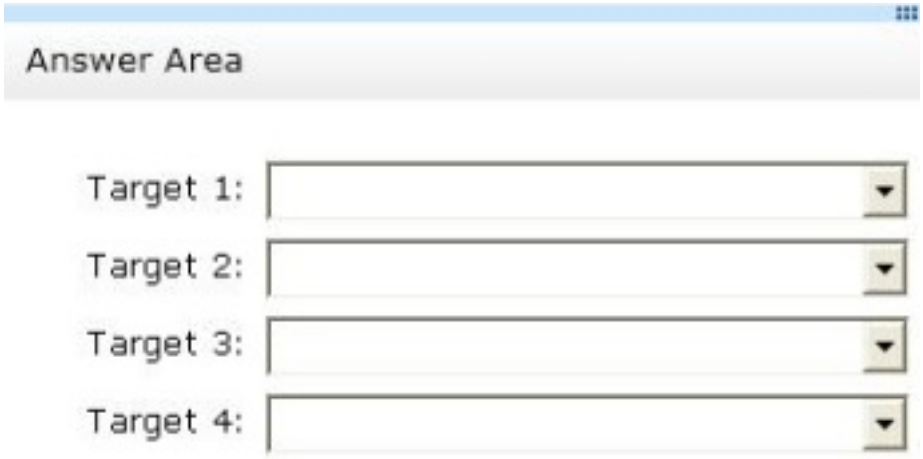
You need to write code to comply with the search requirements of the item detail page.

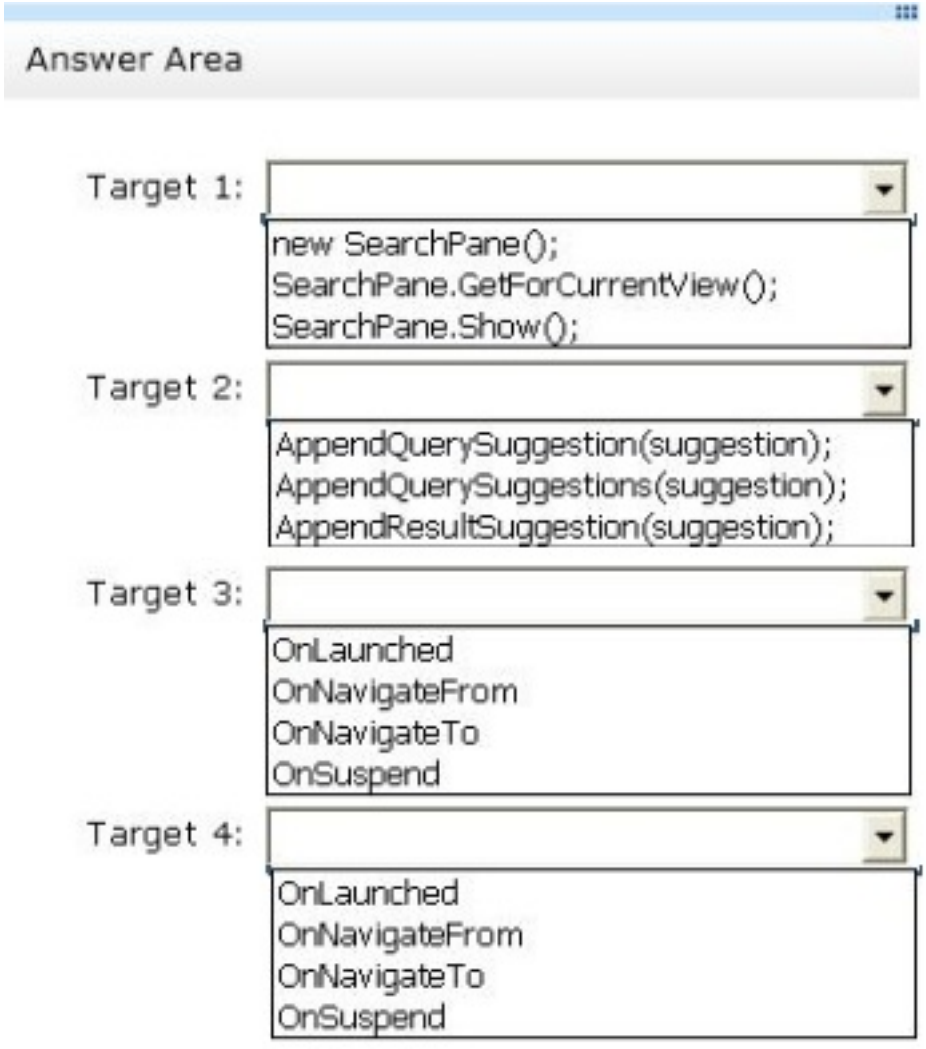
You have the following code:

```
public sealed partial class ItemDetail : Page
{
    private SearchPane searchPane;
    private static readonly string[] suggestionList =
        {
            "Painting1", "Painting2", "Painting3", "Painting4",
            "Sculpture1", "Sculpture2", "Sculpture3", "Sculpture4"
        };
    public ItemDetail()
    {
        this.InitializeComponent();
        searchPane = Target 1
    }
    private void OnSearchPaneSuggestionsRequested(SearchPane sender,
        SearchPaneSuggestionsRequestedEventArgs e)
    {
        var queryText = e.QueryText;
        if (!string.IsNullOrEmpty(queryText))
        {
            var request = e.Request;
            foreach (string suggestion in suggestionList)
            {
                if (suggestion.StartsWith(queryText,
                    StringComparison.CurrentCultureIgnoreCase))
                {
                    request.SearchSuggestionCollection.Target 2
                }
            }
        }
    }
    protected override void Target 3(NavigationEventArgs e)
    {
        searchPane.SuggestionsRequested +=
            new TypedEventHandler<SearchPane,
            SearchPaneSuggestionsRequestedEventArgs>
            (OnSearchPaneSuggestionsRequested);
    }
    protected override void Target 4(NavigationEventArgs e)
    {
        searchPane.SuggestionsRequested -=
            new TypedEventHandler<SearchPane,
            SearchPaneSuggestionsRequestedEventArgs>
            (OnSearchPaneSuggestionsRequested);
    }
}
```

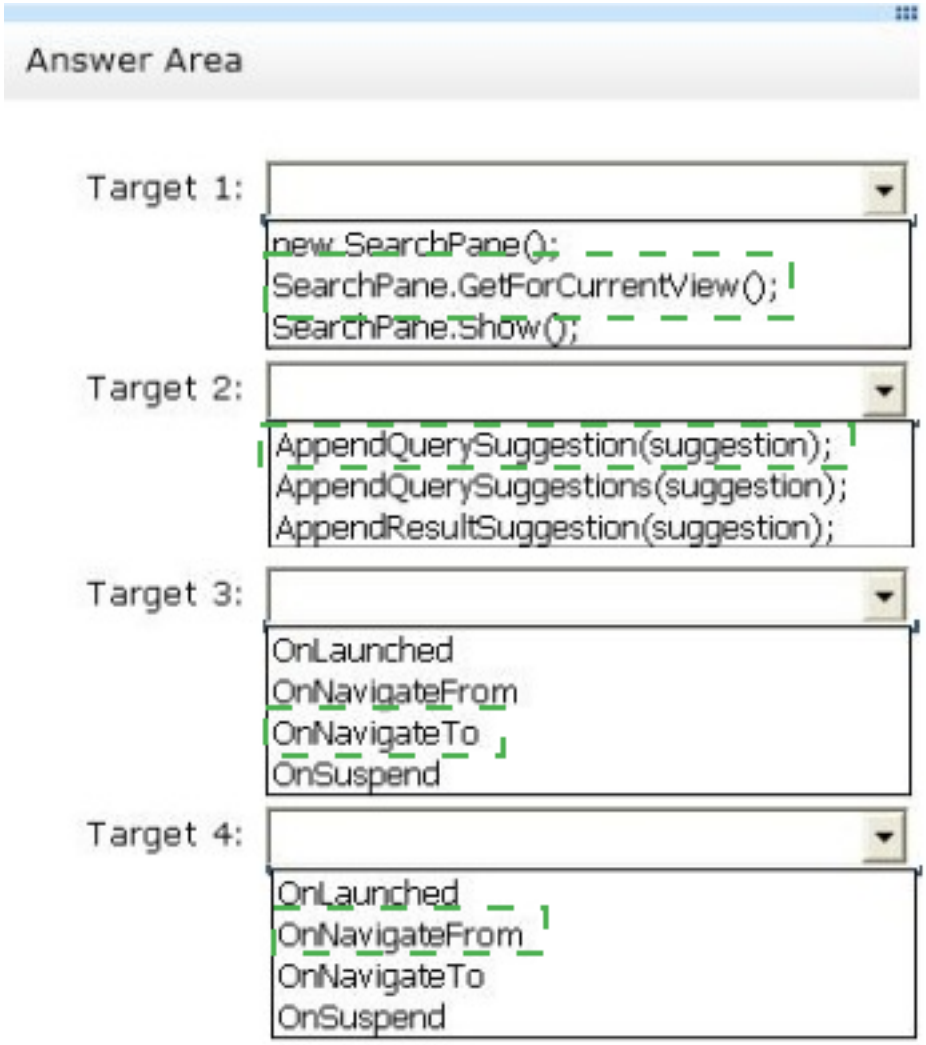
Which code snippets should you insert in Target 1, Target 2, Target 3, and Target 4 to

complete the code? (To answer, select the correct code snippet from each drop-down list in the answer area.)





Answer:



Topic 2, Scenario 2

Overview

Fabrikam, Inc. is a realtor in the United States.

Fabrikam grants its customers access to a web site, where they can search for houses for rent and for sale. Its customers can enter basic requirements, such as location, number of rooms, dimensions, and a price range. The web site displays a list of houses that meet the customers' criteria. The customers can then view more details about each house and can add a listing to a favorites list.

Requirements

Business Goals

Fabrikam plans to provide a more interactive experience for its customers. Fabrikam is creating a video tour for each listing. The video tours can be used to visit the property virtually.

Fabrikam plans to create a Windows Store app on Windows 8.1 RT and Windows 8.1 Pro devices.

General Requirements

Fabrikam identifies the following general requirements for the app:

- The app interface must be available in English, Spanish, and French.
- The app must provide the customers with the ability to perform searches the same way that the current web site does.
- It is expected that the customers will view more than 3,000 pictures annually. The main page of the app must show a list of the last 10 pictures that were viewed.
- If pictures are added to a listing that is in a customer's favorites list, the pictures must be downloaded automatically from Microsoft Azure. This must occur if the app is suspended or not running.

Printing Requirements

Customers must be able to print the details of a listing from the details page by clicking a button within the app. You plan to add the following XAML markup to the listing details page:

<Button x:Name="btnPrint" Content-"Print" Click-"InvokePrint" />.

Video Tour Requirements

Fabrikam identifies the following requirements for the video tours:

- Customers must be able to play the video tour on a different device by using a button within the app.
- When a customer clicks the details of a listing, the app must start downloading the video tour in the background.
- When the app starts, the app must verify whether there are any pending downloads, and resume any paused downloads.
- The last five viewed video tours that are not on the customer's favorites list must be cached for subsequent viewing.
- Customers must be able to download all of the video tours for the properties that they added to their favorites list.
- The property details page must contain a MediaElement control that will be used to play the video tour of the property.
- When downloading the video tours, the app must remain responsive, and each download must be processed on a separate thread.

Package appxmanifest

```
01 <Extension Category="windows.backgroundTasks"
02   EntryPoint="Tasks.DownloadPictures">
03 <BackgroundTasks>
04
05 </BackgroundTasks>
06 </Extension>
```

Question No : 4 DRAG DROP - (Topic 2)

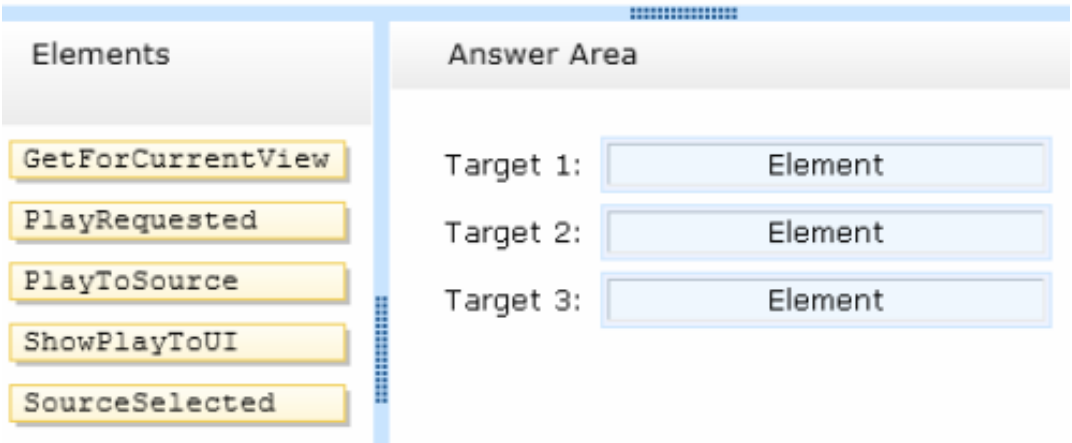
You add a MediaElement named VideoTour and a button named playToButton to the properties details page.

You need to ensure that video tours can be played to other devices.

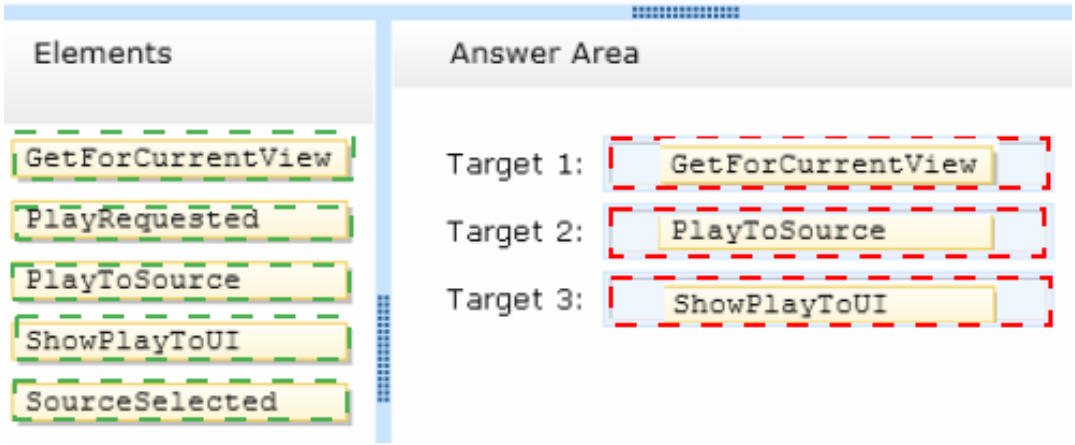
You have the following code: (Line numbers are included for reference only.)

```
01 PlayToManager playToManager = null;
02 CoreDispatcher dispatcher = null;
03 protected override void OnNavigatedTo(NavigationEventArgs e)
04 {
05    dispatcher = Window.Current.CoreWindow.Dispatcher;
06    playToManager = PlayToManager.Target 1();
07    playToManager.SourceRequested += playToManager_SourceRequested;
08 }
09 void playToManager_SourceRequested(PlayToManager sender,
10    PlayToSourceRequestedEventArgs args)
11 {
12    var deferral = args.SourceRequest.GetDeferral();
13    var handler = dispatcher.RunAsync(CoreDispatcherPriority.Normal, () =>
14    {
15       args.SourceRequest.SetSource(VideoTour.Target 2);
16       deferral.Complete();
17    });
18 }
19 private void playToButton_Click(object sender, RoutedEventArgs e)
20 {
21    playToManager.Target 3();
22 }
```

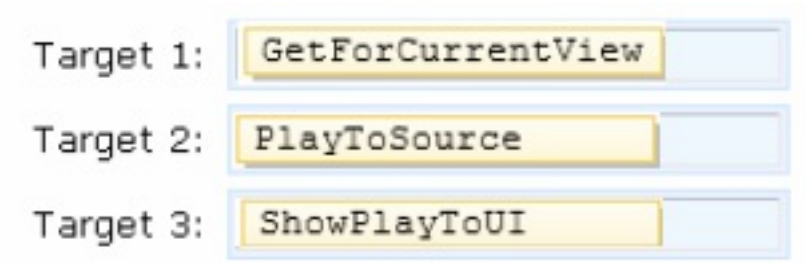
Which elements should you include in Target 1, Target 2 and Target 3 to complete the code? (To answer, drag the appropriate elements to the correct targets in the answer area. Each element may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.)



Answer:



Explanation:



http://msdn.microsoft.com/en-us/library/windows/apps/windows.media.playto.aspx

Question No : 5 - (Topic 2)

You create a mobile service to send push notifications to the app.

You configure the service and the app to work with Windows Push Notification Services (WNS).

You add the following code to the App.xaml.cs file:

```
using Windows.Networking.PushNotifications;
...
public static PushNotificationChannel pushChannel
{get; private set; }
private async void GetChannel()
{
  pushChannel = await PushNotificationChannelManager.
    CreatePushNotificationChannelForApplicationAsync();
}
```

You need to ensure that the app can access the push notification channel.

What should you do first?

A. Add a call to GetChannel in the OnLaunched event handler of the app.
B. Set the Uri property of pushChannel in the OnActivated event handler of the app.
C. Set the Uri property of pushChannel in the OnLaunched event handler of the app.
D. Add a call to GetChannel in the OnActivated event handler of the app.

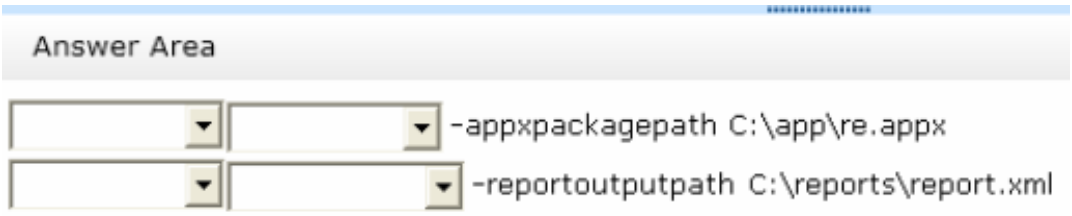
Answer: A

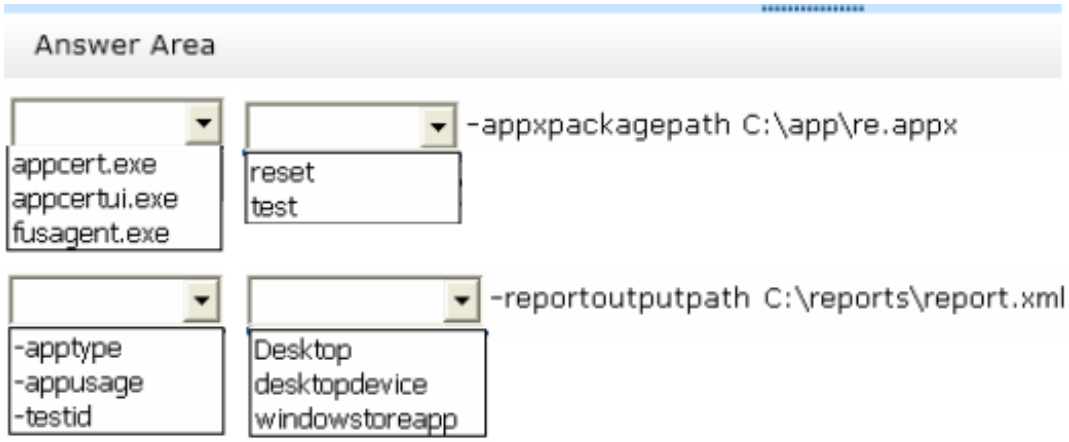
Explanation: http://msdn.microsoft.com/en-us/library/windows/apps/windows.ui.xaml.application.onlaunched.aspx

Question No : 6 HOTSPOT - (Topic 2)

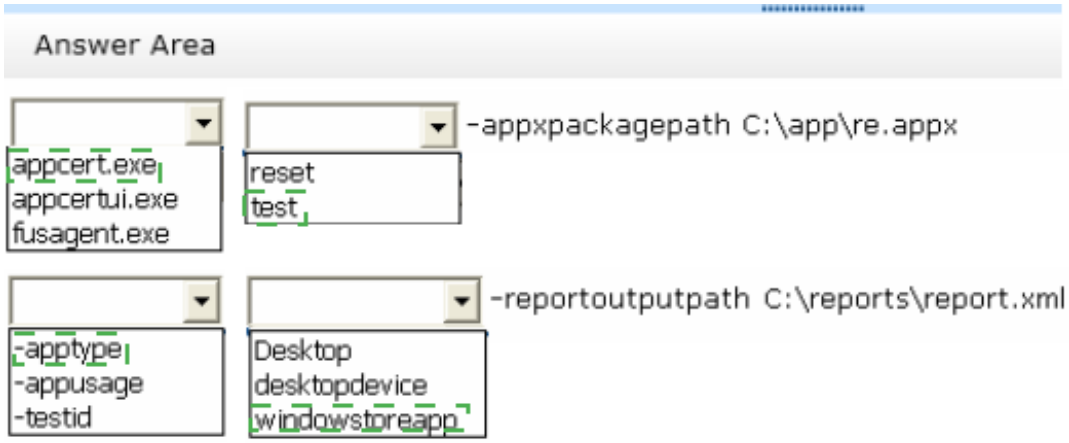
You need to verify whether the app conforms to the Windows Store requirements.

What command should you run? (To answer, select the appropriate options in the answer area.)

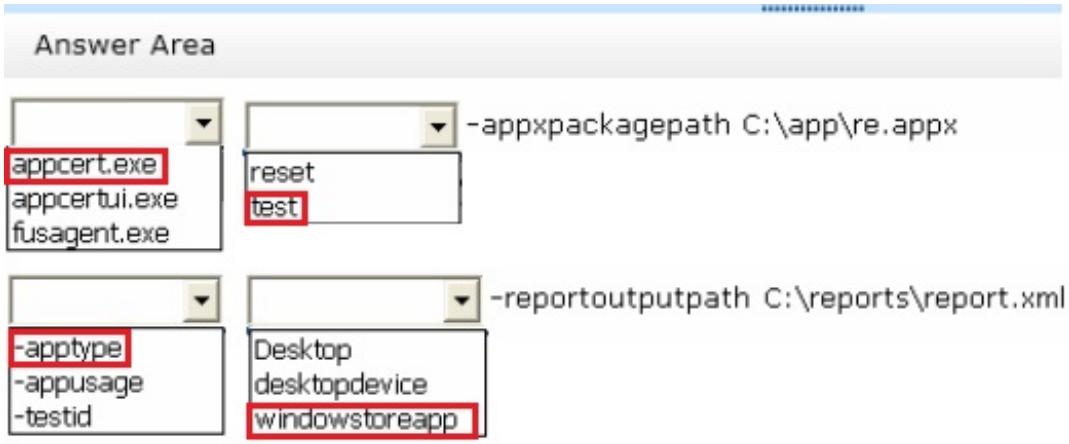




Answer:



Explanation:



http://msdn.microsoft.com/en-us/library/windows/apps/hh694081.aspx

Topic 3, Scenario 3

Background

You are developing a Windows Store app. The app will allow ornithologists to photograph migrating geese, taking note of the location, heading, and weather conditions at the time each photo is taken.

Business Requirements

The app must adhere to the following requirements:

- Create and store photographs of migrating geese.
- Record the location and weather conditions where the photograph was taken.
- Record the heading and time that the photograph was taken.
- Allow the user to display the information on any device that supports the PlayTo feature.

Technical Requirements

General:

The app must meet the following technical requirements:

- The app must store images and image metadata in the Pictures Library.
- The metadata logic must be encapsulated within a reusable component named LogicComponent1.
- The metadata logic must be available to Windows Store apps written in Visual Basic, C#, JavaScript, and C++.

Hardware:

- The app requires a device with camera, compass, and GPS features.
- The app requires a device with Internet capabilities.

CurrentEnvironment.es

```
CE01 namespace CurrentEnvironment
CE02 {
CE03   public sealed class Environment
CE04   {
CE05     private Compass _compass = null;
CE06     private LightSensor _light = null;
CE07     public IAsyncOperation<EnvironmentalStatus> GetCurrentEnvironmentAsync()
CE08     {
CE09       LoadSensors();
CE10       return (IAsyncOperation<EnvironmentalStatus>)AsyncInfo.Run(
CE11         (System.Threading.CancellationToken ct) => InternalGetCurrentEnvironmentAsync());
CE12     }
CE13
CE14     private async Task<EnvironmentalStatus> InternalGetCurrentEnvironmentAsync()
CE15     {
CE16       EnvironmentalStatus es = new EnvironmentalStatus();
CE17       es.Location = await GetLocationAsync();
CE18       ...
CE19       es.Temperature = await GetWeatherAsync();
CE20       es.Time = DateTime.UtcNow.ToString();
CE21
CE22       return es;
CE23     }
CE24
CE25     private async Task<string> GetLocationAsync()
CE26     {
```

```
CE27        var locator = new Geolocator();
CE28        Geoposition location = await locator.GetGeopositionAsync();
CE29        string curPosition = location.Coordinate.Latitude.ToString() + ", "
CE30          + location.Coordinate.Longitude.ToString();
CE31        if(_compass != null)
CE32          curPosition += ", " + _compass.GetCurrentReading().HeadingTrueNorth.Value;
CE33        return curPosition;
CE34      }
CE35
CE36      private async Task<string> GetWeatherAsync()
CE37      {
CE38        IList<WeatherData> weatherData = GooseLogic.GetWeatherData();
CE39      }
CE40
CE41      private void LoadSensors()
CE42      {
CE43
CE44        {
CE45          _compass = Compass.GetDefault();
CE46        }
CE47      }
CE48    }
CE49
CE50    public struct EnvironmentalStatus
CE51    {
CE52      public string Location;
CE53      public string Time;
CE54      public string Temperature;
CE55    }
CE56 }
```

MainPage.xaml.es

```
MP01 private async void CapturePhoto_Click(object sender, RoutedEventArgs e)
MP02 {
MP03   try
MP04   {
MP05     CameraCaptureUI cameraUI = new CameraCaptureUI();
MP06     Size aspectRatio = new Size(16, 9);
MP07     cameraUI.PhotoSettings.CroppedAspectRatio = aspectRatio;
MP08
MP09     StorageFile file = await cameraUI.CaptureFileAsync(CameraCaptureUIMode.Photo);
MP10     if (file != null)
MP11     {
MP12       var newFile = await Windows.Storage.KnownFolders.PicturesLibrary.CreateFileAsync(file.Name);
MP13       await file.CopyAndReplaceAsync(newFile);
MP14       BitmapImage bitmapImage = new BitmapImage();
MP15       using (IRandomAccessStream fileStream = await newFile.OpenAsync(FileAccessMode.Read))
MP16       {
MP17         bitmapImage.SetSource(fileStream);
MP18       }
MP19       capturedPhoto.Source = bitmapImage;
MP20
MP21       var env = new CurrentEnvironment.Environment();
MP22       var envData = await env.GetCurrentEnvironmentAsync();
MP23
MP24       Info.Text = envData.Location;
MP25     }
MP26     else
MP27     {
MP28       Info.Text = "An error has occurred";
MP29     }
MP30   }
MP31   catch (Exception ex)
MP32   {
MP33   ...
MP34   }
MP35 }
```

Package.appxmanifest

```
PA01 <?xml version="1.0" encoding="utf-8"?>
PA02 <Package xmlns="http://schemas.microsoft.com/appx/2010/manifest">
PA03  <Identity Name="7d32c109-5e1d-432a-a53f-df00440658f0" Publisher="CN=Admin" Version="1.0.0.0"/>
PA04  <Properties>
PA05    <DisplayName>GooseTracker</DisplayName>
PA06    <PublisherDisplayName>Admin</PublisherDisplayName>
PA07    <Logo>Assets\StoreLogo.png</Logo>
PA08  </Properties>
PA09  <Prerequisites>
PA10    <OSMinVersion>6.2.1</OSMinVersion>
PA11    <OSMaxVersionTested>6.2.1</OSMaxVersionTested>
PA12  </Prerequisites>
PA13  <Resources>
PA14    <Resource Language="x-generate"/>
PA15  </Resources>
PA16  <Applications>
PA17    <Application Id="App" Executable="$targetnametoken$.exe" EntryPoint="GooseTracker.App">
PA18      <VisualElements DisplayName="GooseTracker" Logo="Assets\Logo.png" SmallLogo="Assets
\SmallLogo.png"
          Description="GooseTracker" ForegroundText="light" BackgroundColor="#464646">
PA19        <DefaultTile ShowName="allLogos"/>
PA20        <SplashScreen Image="Assets\SplashScreen.png"/>
PA21      </VisualElements>
PA22    </Application>
PA23  </Applications>
PA24  <Capabilities>
PA25
PA26    <Capability Name="internetClient"/>
PA27    <DeviceCapability Name="webcam"/>
PA28    <DeviceCapability Name="location"/>
PA29  </Capabilities>
PA30 </Package>
```

GooseTracker.csproj

```
GO01 <Project ToolsVersion="4.0" DefaultTargets="Build"
       xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
GO02
GO03    <ItemGroup>
GO04
GO05    </ItemGroup>
GO06 </Project>
```

GoosePlayTo.es

```
PT00  public class GoosePlayTo
PT01  {
PT02    private Windows.Media.PlayTo.PlayToManager playToManager;
PT03    private Windows.UI.Core.CoreDispatcher dispatcher;
PT04    private MediaElement element;
PT05
PT06    public GoosePlayTo(MediaElement element)
PT07    {
PT08      dispatcher = Window.Current.CoreWindow.Dispatcher;
PT09      playToManager = Windows.Media.PlayTo.PlayToManager.GetForCurrentView();
PT10      playToManager.SourceRequested += SourceRequested;
PT11      this.element = element;
PT12    }
PT13
PT14    private void SourceRequested(Windows.Media.PlayTo.PlayToManager sender,
PT15      Windows.Media.PlayTo.PlayToSourceRequestedEventArgs args)
PT16    {
PT17      var def = args.SourceRequest.GetDeferral();
PT18      var evthander = dispatcher.RunAsync(Windows.UI.Core.CoreDispatcherPriority.Normal,
PT19          () =>
PT20          {
PT21            args.SourceRequest.SetSource(element.PlayToSource);
PT22            def.Complete();
PT23          }
PT24      );
PT25    }
PT26
PT27    private async void LoadFile(Windows.Storage.StorageFile videoFile, string contentType)
PT28    {
PT29      var stream = await videoFile.OpenAsync(Windows.Storage.FileAccessMode.Read);
PT30
PT31    }
PT32
PT33    private void Play()
PT34    {
PT35      element.Play();
PT36    }
PT37
PT38    private void Pause()
PT39    {
PT40      element.Pause();
PT41    }
PT42  }
```

Camera.cs

```
CA01 public class Camera: Windows.Media.Devices.IMediaDeviceController
CA02 {
CA03     private Windows.Media.Capture.MediaCapture media;
CA04     private Windows.Media.Devices.VideoDeviceController video;
CA05     public double WhiteBalance
CA06     {
CA07       get
CA08       {
CA09         double wbValue = -1.0;
CA10
CA11         return wbValue;
CA12       }
CA13     }
CA14
CA15     public bool SupportsBacklightCompensation
CA16     {
CA17       get
CA18       {
CA19
CA20       }
CA21     }
CA22
CA23     public Camera()
CA24     {
CA25       media = new Windows.Media.Capture.MediaCapture();
CA26       ...
CA27       video = media.VideoDeviceController;
CA28
CA29
CA30     }
CA31 }
```

Question No : 7 - (Topic 3)

Users report performance issues when getting the location information associated with a photo. You suspect the app is encountering performance issues in the GetLocationAsync() method of the Environment class.

You need to enhance the performance of the GetLocationAsync() method of the app.

What should you do?

A. Remove the Compass initialization from the LoadSensors() method and initialize it within the GetLocationAsync() method.
B. set the ReportInterval property of the Compass object to 16.
C. set the ReportInterval property of the Compass object to 0.
D. Move the locator variable to a class level variable and initialize it in the Environment

constructor.

Answer: D

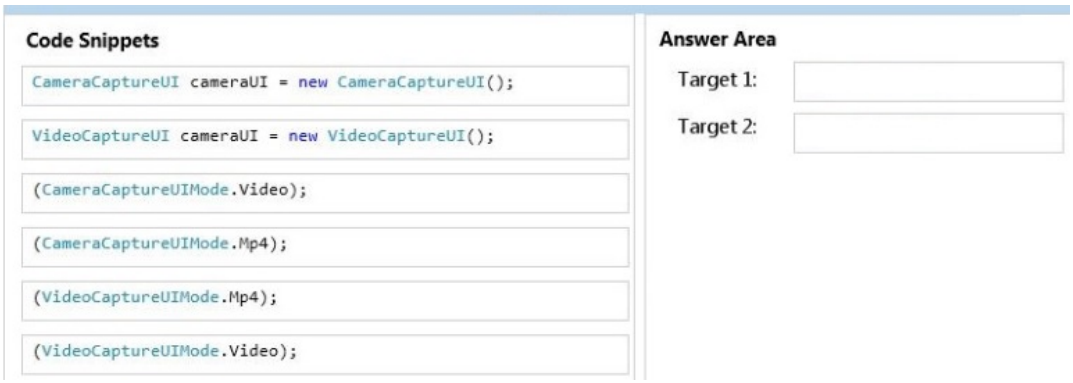
Question No : 8 DRAG DROP - (Topic 3)

You need to allow users to capture video instead of photos.

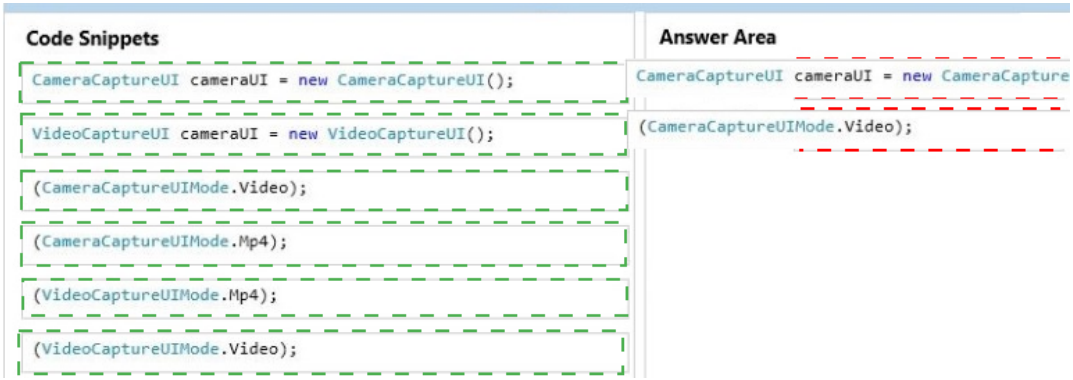
You have the following code:

```
try
{
   Target 1
   cameraUI.VideoSettings.Format =
    CameraCaptureUIVideoFormat.Mp4;
   StorageFile file = null;
   file = await cameraUI.CaptureFileAsync
   Target 2
   if (file !=null)
}
```

Which code snippets should you include in Target 1 and Target 2 to complete the code? (To answer, drag the appropriate code snippets to the correct targets in the answer area. Each code snippet may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.)



Answer:



Explanation:

http://msdn.microsoft.com/en-us/library/windows/apps/windows.media.capture.cameracaptureui.aspx

Question No : 9 - (Topic 3)

You place a breakpoint at line MP31 in the app.

When you debug the app, the debugger continuously catches a System.UnauthorizedAccess exception.

You need to resolve the exception.

What should you do?

A. Wrap lines CE43 through CE46 in a try-catch statement.
B. At line MP10, change the code segment to the following line of code.
read if(cameraUI != null)
C. Move line CE09 to CE16.
D. At line PA25, insert the following line of code.
<Capability Name="picturesLibrary"/>

Answer: D

Explanation: http://msdn.microsoft.com/en-us/library/windows/apps/hh464936.aspx

Topic 4, Scenario 4

Background

You are developing a Windows Store app named Picture Sharer. The app will allow users to capture, modify, caption, and share pictures.

Application Structure

The ShareImageButton and GetContactsButton controls use the same foreground color. The foreground color might change in the future.

The following code defines a custom button style named ButtonStyleRed:

```
<Style TargetType="Button" x:Key="ButtonStyleRed">
  <Setter Property="Foreground" Value="#FFC34343"/>
  <Setter Property="BorderBrush" Value="#FFC34343"/>
  ...
</Style>
```

Relevant portions of the app files are shown. (Line numbers are included for reference only and include a two-character prefix that denotes the specific file to which they belong.)

Business Requirements

The app must meet the following business requirements:

- Allow users to capture and retrieve pictures, modify pictures by adding a shading effect, and add captions to images.
- Support only Landscape and Landscape-flipped orientation.
- Allow users to capture and retrieve pictures, modify pictures by adding a shading effect, and add captions to images.
- Support only Landscape and Landscape-flipped orientations.
- Ensure that users can select and modify images from the PictureChooserPage page.
- Ensure that users can change the magnification of the selected image and resize the image by using pinch and stretch gestures. Scaling should be fluid and precisely controlled by the user.

The app must be localized for the French Canadian market

Technical Requirements

The app must meet the following technical requirements:

- Scroll bars must not be visible.
- The CaptionTextBlock and CaptionTextBox controls must appear side by side, without overlapping and on the same line. The CaptionTextBox control should appear to the right of the CaptionTextBlock control.
- The ContactPicker object must be filtered to display only email addresses.
- Minimize the code that is required to implement optical zoom functionality.

You must perform the following tasks:

- Handle the Click event of the GetPictureButton control to switch from the current page to the PictureChooserPage page.
- After the user selects an image on the PictureChooserPage page, ensure that the app navigates back to the PictureSharerMainPage page.
- Track the current screen orientation and page size by updating the _currentViewState, _currentHeight and _currentWidth fields every time the screen orientation or page size changes.
- Create a style named ButtonStyleWhite that inherits all the style settings of the ButtonStyleRed style except the border color; the border color must be white. The ButtonStyleWhite style must automatically update with any changes that are made to the ButtonStyleRed style.
- Create a resource named ButtonForegroundColor to implement the button foreground color so that it can be referenced in XAML by using the following standard syntax: Foreground="{StaticResource ButtonForegroundColor}"
- Ensure that the OnNavigatedToO method updates the current picture when a new picture is selected.
- Change the background for the root Grid element to a vertical gradient that transitions from black at the top to maroon at the bottom. Create a resource named GridBackgroundGradientBrush to hold the requested gradient.

While testing the app, you observe the following results:

- An exception is being thrown in the GetContactsCompleted event handler when the retrieved email address is assigned to the RecipientsTextBlock control. The exception message states: "The application called an interface that was marshalled for a different thread."
- When users navigate away from the PictureSharerMainPage page, information that was entered in the CaptionTextBox control is lost.

PictureSharerMainPage.xaml

```
XA01 <Page
XA02   x:Class="PictureSharer.PictureSharerMainPage"
XA03   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
XA04   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
XA05   xmlns:local="using:PictureSharer"
XA06   xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
XA07   xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
XA08   mc:Ignorable="d">
XA09
XA10   <Grid Background="{StaticResource ApplicationPageBackgroundBrush}">
XA11     <Image x:Name="SelectedImage" Source="Images/blank.jpg" Width="800" Height="800" />
XA12     <TextBlock x:Name="RecipientsTextBlock"/>
XA13     <StackPanel>
XA14       <TextBlock x:Name="CaptionTextBlock" Text="Caption"/>
XA15       <TextBox x:Name="CaptionTextBox"/>
XA16     </StackPanel>
XA17     <Button x:Name="ShareImageButton" Click="ShareImageButton_Click" Foreground="#FFC34343">
XA18      Send Image</Button>
XA19    <Button x:Name="GetContactsButton" Click="GetContactsButton_Click" Foreground="#FFC34343">
XA20      Get Contacts</Button>
XA21    <Button x:Name="GetPictureButton" Click="GetPictureButton_Click" Foreground="#FFC34343" >
XA22      Get Picture</Button>
XA23   </Grid>
XA24 </Page>
```

PictureSharerMainPage.xaml.es

```
CS01 public sealed partial class PictureSharerMainPage : Page
CS02 {
CS03   private ApplicationViewState _currentViewState;
CS04   private double _currentHeight, _currentWidth;
CS05   public PictureSharerMainPage()
CS06   {
CS07     this.InitializeComponent();
CS08
CS09
CS10   }
CS11   protected override void OnNavigatedTo(NavigationEventArgs e)
CS12   {
CS13
CS14   }
CS15   private void GetContactsButton_Click(object sender, RoutedEventArgs e)
CS16   {
CS17     var picker = new ContactPicker();
CS18
CS19     var results = picker.PickSingleContactAsync();
CS20     results.Completed += GetContactsCompleted;
CS21   }
CS22   private void GetContactsCompleted(IAsyncOperation<ContactInformation> op,
CS23     AsyncStatus status)
CS24   {
CS25     var emailList = new List<string>();
CS26     var contact = op.GetResults();
CS27     if (contact.Emails.Count == 0)
CS28       return;
CS29     foreach (var info in contact.Emails)
CS30       emailList.Add(info.Value);
CS31     var email = string.Join(";", emailList);
CS32     RecipientsTextBlock.Text = email;
CS33   }
CS34   private void ShareImageButton_Click(object sender, RoutedEventArgs e)
CS35   {
CS36     SendImageToCloud();
CS37   }
CS38   private void SendImageToCloud()
CS39   {
CS40     ...
CS41   }
CS42
CS43 }
```

PictureChooserPage.xaml

```
PC01 <Page
PC02  x:Class="PictureSharer.PictureChooserPage"
PC03  xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
PC04  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
PC05  xmlns:local="using:PictureSharer"
PC06  xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
PC07  xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
PC08  mc:Ignorable="d">
PC09   <Grid Background="{StaticResource ApplicationPageBackgroundBrush}">
PC10     <Image x:Name="SelectedImage" Source="Images/image1234.jpg"/>
PC11     <Button Content="Back" HorizontalAlignment="Left" Margin="227,25,0,0"
PC12      VerticalAlignment="Top" Width="75" Click="Button_Click_1"/>
PC13   </Grid>
PC14 </Page>
```

Question No : 10 - (Topic 4)

You need to create the ButtonForegroundColor resource.

Which code segment should you insert at line XA09?

A.
```
<Page.Resources>
  <SolidColorBrush x:Key="ButtonForegroundColor" Color="#FFC34343"/>
</Page.Resources>
```

B.
```
<Page.Resources>
  <Style TargetType="Button" x:Key="ButtonForegroundColor">
    <Setter Property="Foreground" Value="#FFC34343"/>
  </Style>
</Page.Resources>
```

C.
```
<Page.Resources>
  <ButtonForegroundColor>#FFC34343</ButtonForegroundColor>
</Page.Resources>
```

D.
```
<Page.Resources>
  <Color x:Key="ButtonForegroundColor">#FFC34343</Color>
</Page.Resources>
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: A

Explanation: * Create a resource named ButtonForegroundColor to implement the button foreground color so that it can be referenced in XAML by using the following standard syntax: Foreground-"{StaticResourceButtonForegroundColor}"
* SolidColorBrush

Question No : 11 - (Topic 4)

You need to track the screen orientation and page size.

Which code segment should you insert at line CS09?

A.
```
this.SizeChanged += (object sender, WindowSizeChangedEventArgs e) =>
{
  _currentViewState = Windows.UI.ViewManagement.ApplicationView.GetForCurrentView();
  _currentHeight = e.Size.Height;
  _currentWidth = e.Size.Width;
};
```

B.
```
this.SizeChanged += (object sender, WindowSizeChangedEventArgs e) =>
{
  _currentViewState = Windows.UI.ViewManagement.ApplicationView.Value;
  _currentHeight = e.Size.Height;
  _currentWidth = e.Size.Width;
};
```

C.
```
this.SizeChanged += (object sender, SizeChangedEventArgs e) =>
{
  _currentViewState = Windows.UI.ViewManagement.ApplicationView.Value;
  _currentHeight = e.NewSize.Height;
  _currentWidth = e.NewSize.Width;
};
```

D.
```
this.SizeChanged += (object sender, SizeChangedEventArgs e) =>
{
  _currentViewState = Windows.UI.ViewManagement.ApplicationView.GetForCurrentView();
  _currentHeight = e.NewSize.Height;
  _currentWidth = e.NewSize.Width;
};
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: A

Question No : 12 - (Topic 4)

You need to configure the Picture Sharer app to support only the required device orientations.

What should you do?

A. In the App.xaml file, configure the Portrait and Portrait-flipped orientations.
B. In the Package.appxmanifest file, configure the Snapped and Filled orientations.
C. In the PictureSharerMainPage.xaml file, configure the Landscape and Landscape-flipped orientations.
D. In the App.xaml file, configure the Portrait and Landscape orientations.
E. In the App.manifest file, configure the Portrait and Portrait-flipped orientations.
F. In the Package.appxmanifest file, configure the Landscape and Landscape-flipped orientations.

Answer: F

Question No : 13 - (Topic 4)

You need to localize the Picture Sharer app in the required language.

Which actions should you perform? (Each correct answer presents part of the solution. Choose all that apply.)

A. Add a Uid attribute to any XAML elements that must be localized.
B. Create a folder named fr-CA at the root of the project.
C. Create a resource file named resources.res.
D. Create a resource file named resources.resw.
E. Create a folder named es-ES at the root of the project.
F. Add a Name attribute to any XAML elements that must be localized.

Answer: A,B,D

Explanation: * (A) To localize a certain property of a XAML element you only need to add a x:Uid="SomeKey" attribute to the element and add the appropriate resource to the .resw file.
* (B) The app must be localized for the French Canadian market.
* (BD) Example: A French language resource named "Greeting" whose value is " Bonjour!". To create the resource file, add a folder named fr-FR to your project, and then add a resource file named Resources.resw to the folder.
* In Windows Store apps, you designate the names of localized resource files by creating a folder to store the resources and images of a supported culture. You can then describe the

resource by using the culture name (such as "ko-kr") followed by the default resource name and resource file extension (such as "ko-kr\Resources.resw").

URL: http://msdn.microsoft.com/en-us/library/windows/apps/hh694557(v=vs.110).aspx

Topic 5, Scenario 5

Background

You are developing a Windows Store style e-reader app.

Business Requirements

- Users must be able to upload e-books and documents and download them to e-reader devices.
- Users must be able to set a password to restrict access to their e-books and documents.
- Users must be able to create and store encrypted metadata about their e-books and documents.
- The app must replace system-generated error messages with custom-defined messages. These custom messages must come from a list of approved text.
- User actions such as printing pages, saving users' current locations in documents, and taking notes should be enabled from buttons on an AppBar control.
- The app must provide trial functionality that will expire after 14 days. If the app expires while it is running, the app must display an expiration message to the user and prompt the user to purchase the app.

Technical Requirements

General:

- Configuration files must be read-only. All user settings must be stored in the Contoso Settings Service.
- The SocialPoller background task must run the code in the DoWork() method to collect content from the Contoso feed.
- The UI must always remain responsive to user actions.

Security:

- Secured e-book and document passwords must to be encrypted so that only the user who created the passwords can retrieve the metadata associated to the e-books and documents.
- The system must log all exceptions through the auditing object and notify technicians of the issue.

Storage:

- The app must cache the next two chapters to the local device for users to read while disconnected from the network. This cache must be persisted if a reboot is

performed.

- User state such as the current location in an e-book or document must be stored in the Microsoft Azure SQL database.
- User settings such as font sizes and colors must be stored through the Contoso Settings Service.

Network:

- Communication between the app and e-book vendors must occur over an encrypted communication channel.
- Communication must use certificates to enable the SSL connection.

Trial Functionality:

- The isPrintEnabled variable must determine if the user can print.
- The isMarketEnabled variable must determine if the user can use the marketplace.
- The isTrialEnabled variable must determine if the application is still in trial mode.

Printing:

- The default printing options are portrait orientation and grayscale color mode.
- The app must enable the user to select the media size and printing orientation.

SocialPoller.es

```
SP01 using System;
SP02 using System.Collections.Generic;
SP03 using System.Linq;
SP04 using System.Net.Http;
SP05 using System.Text;
SP06 using System.Threading;
SP07 using System.Threading.Tasks;
SP08 using Windows.ApplicationModel.Background;
SP09 namespace Ereader.Background
SP10 {
SP11  public class SocialPoller : IBackgroundTask
SP12  {
SP13
SP14    public async Task<string> DoWork()
SP15    {
SP16      HttpClient client = new HttpClient();
SP17      client.BaseAddress = new Uri("http://feed.contoso.com/");
SP18      HttpResponseMessage response = await client.GetAsync(client.BaseAddress,
            HttpCompletionOption.ResponseContentRead);
SP19      string content = await response.Content.ReadAsStringAsync();
SP20      return content;
SP21    }
SP22  }
SP23 }
```

Auditor.cs

```
AU01 using System;
AU02 using System.Collections.Generic;
AU03 using System.Linq;
AU04 using System.Text;
AU05 using System.Threading.Tasks;
AU06 namespace Ereader.Code
AU07 {
AU08   public class Auditor
AU09   {
AU10     public enum ErrorType
AU11     {
AU12       General,
AU13       NullReference,
AU14       InvalidCast,
AU15       Network
AU16     }
AU17     public static string GetMessage(ErrorType type)
AU18     {
AU19       string output = String.Empty;
AU20       switch (type)
AU21       {
AU22         case ErrorType.General:
AU23           output = "An unknown error has occurred.";
AU24           break;
AU25         case ErrorType.NullReference:
AU26           output = "An attempt was made to reference an unknown object.";
AU27           break;
AU28       }
AU29       return output;
AU30     }
AU31     public static async void WriteAuditAsync(string errorMessage)
AU32     {
AU33       ...
AU34     }
AU35   }
AU36 }
```

ContentPage.es

```
CP01 namespace Ereader.Model.BookObjects
CP02 {
CP03   public class ContentPage
CP04   {
CP05     public int ID { get; set; }
CP06     public string Content { get; set; }
CP07   }
CP08 }
```

Book.cs

```
BO01 using System;
BO02 using System.Collections.Generic;
BO03 namespace Ereader.Model.BookObjects
BO04 {
BO05   public class Book
BO06   {
BO07     public int ID { get; set; }
BO08     public string Title { get; set; }
BO09     public string ShortDescription { get; set; }
BO10     public string LongDescription { get; set; }
BO11     public string Author { get; set; }
BO12     public List<ContentPage> Pages { get; set; }
BO13     public DateTime ReleaseDate { get; set; }
BO14     public string Cover { get; set; }
BO15     public Book() { }
BO16   }
BO17 }
```

SocialPost.es

```
SP01 namespace Ereader.Model.Social
SP02 {
SP03   public class SocialPost
SP04   {
SP05     public string Message { get; set; }
SP06     public string Username { get; set; }
SP07     public string UserId { get; set; }
SP08     public string Source { get; set; }
SP09     public SocialPost() { }
SP10   }
SP11 }
```

Page1.xaml.es

```
PG01 using System;
PG02 using Windows.ApplicationModel.Background;
PG03 using Windows.Graphics.Printing;
PG04 using Windows.UI.Xaml.Controls;
PG05 using Windows.UI.Xaml.Navigation;
PG06 using Windows.UI.Xaml.Printing;
PG07 namespace Ereader
PG08 {
PG09   public sealed partial class Page1 : Page
PG10   {
PG11     private PrintManager printManager = null;
PG12     private IPrintDocumentSource printDocumentSource = null;
PG13     private PrintDocument printDocument = null;
PG14
PG15     public Page1()
PG16     {
PG17       this.InitializeComponent();
PG18       var builder = new BackgroundTaskBuilder { Name = "SocialPollerTask" };
PG19
PG20       BindData();
PG21     }
PG22
PG23
PG24     private void BindData()
PG25     {
PG26       lvBooklist.DataContext = App.Books;
PG27       lvBooklist.ItemsSource = App.Books;
PG28     }
PG29
PG30     private void printManager_PrintTaskRequested(PrintManager sender, PrintTaskRequestedEventArgs e)
PG31     {
PG32       Windows.Graphics.Printing.PrintTask printTask = e.Request.CreatePrintTask("Print Page Title",
             GetPrintSource => GetPrintSource.SetSource(printDocumentSource));
PG33
PG34
PG35     }
PG36   }
PG37 }
```

App.xaml.cs

```
AX01 namespace Ereader
AX02 {
AX03   sealed partial class App : Application
AX04   {
AX05     private static List<Book> _books = new List<Book>();
AX06     public static List<Book> Books { get { return _books; } }
AX07     private Windows.ApplicationModel.Store.LicenseInformation  licenseInformation =
           Windows.ApplicationModel.Store.CurrentAppSimulator.LicenseInformation;
AX08     private bool isPrintingEnabled = true;
AX09     private bool isMarketEnabled = true;
AX10     private bool isTrialComplete = false;
AX11     public App()
AX12     {
AX13       this.InitializeComponent();
AX14       this.Suspending += OnSuspending;
AX15       for (int i = 0; i < 10; i++)
AX16       {
AX17         _books.Add(new Book()
AX18           {
AX19             ...
AX20           });
AX21       }
AX22
AX23     }
AX24   }
AX25 }
```

Question No : 14 - (Topic 5)

You need to protect the metadata for the secure documents.

Which protection descriptor should you use for the DataProtectionProvider object?

A. SID
B. WEBCREDENTIALS=userrpassword
C. LOCAL=user
D. USER=current

Answer: C

Explanation: http://msdn.microsoft.com/en-us/library/windows/apps/windows.security.cryptography.dataprotection.dataprotectionprovider.aspx

Question No : 15 - (Topic 5)

You need to enable the capabilities that allow communication according to the technical requirements.

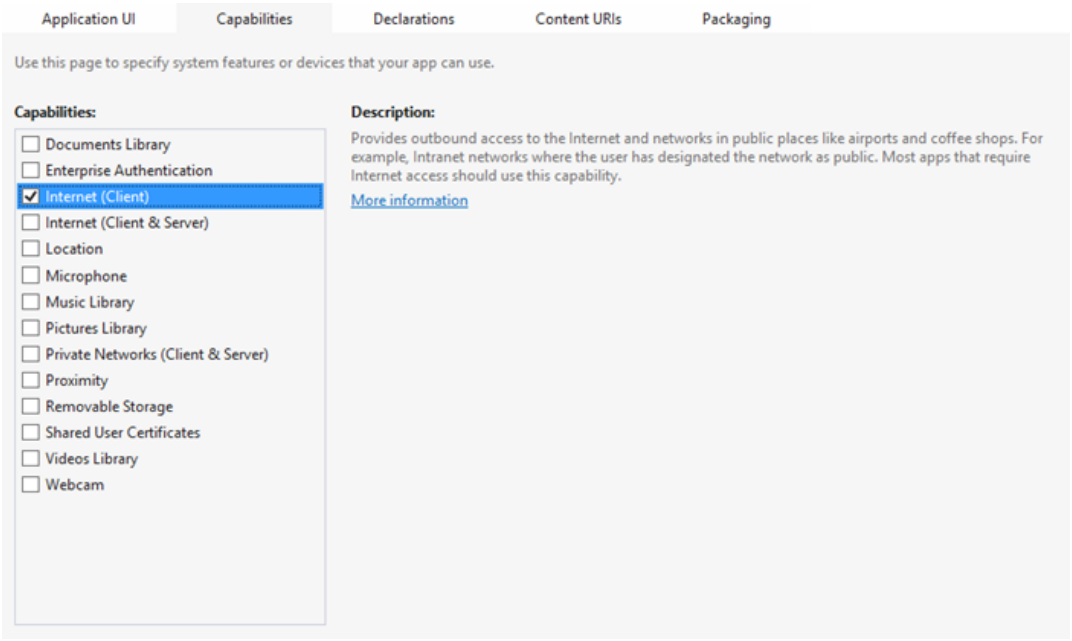
Which capabilities should you enable? (Each correct answer presents part of the solution. Choose all that apply.)

A. Shared User Certificates
B. SSL Certificates
C. Internet (Client)
D. Default Windows Credentials

Answer: B,C

Explanation: B: From scenario:
Communication between the app and e-book vendors must occur over an encrypted communication channel.
Communication must use certificates to enable the SSL connection.

C:

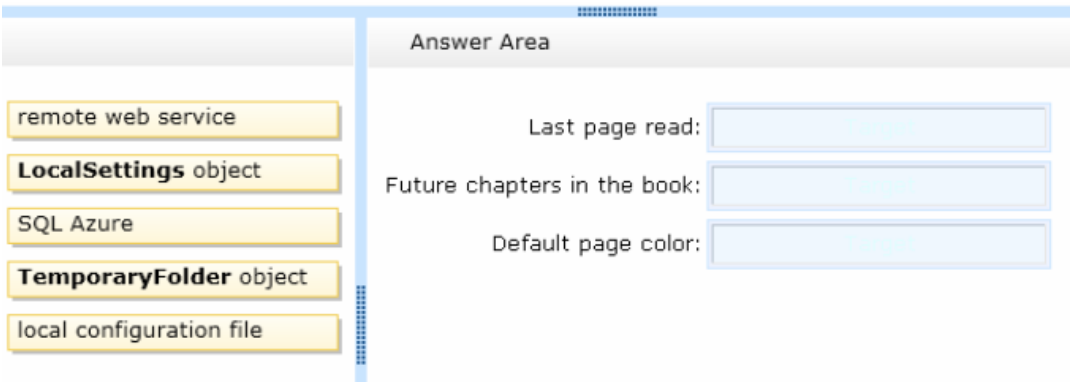


http://msdn.microsoft.com/en-us/library/windows/apps/Hh770532.aspx
http://msdn.microsoft.com/en-us/library/windows/apps/Hh986970.aspx

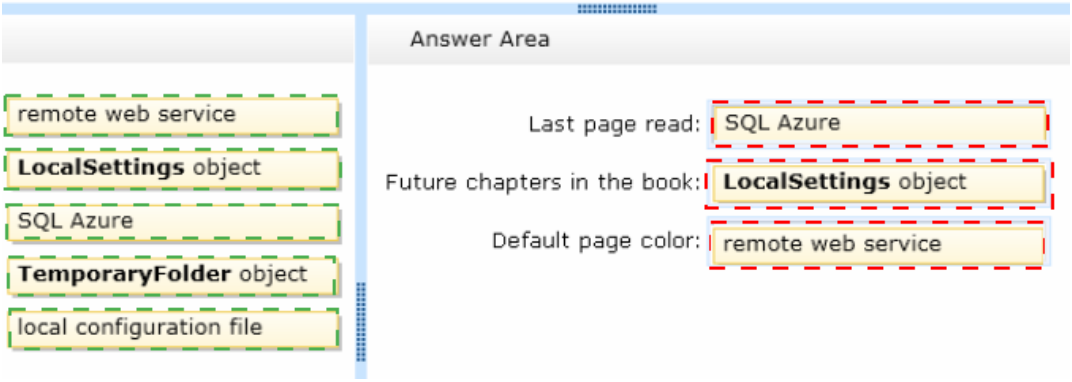
Question No : 16 DRAG DROP - (Topic 5)

You need to meet the app caching requirements.

Which caching technique should you use in each scenario? (To answer, drag the appropriate technique to the correct scenario. Each technique may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.)



Answer:



Topic 6, Scenario 6

Business Requirements

The app must enable users to perform the following tasks:

- Define a feed title.
- Define a list of RSS feeds that the users want to subscribe to.
- View information about topics that are popular on the users' selected social networks.
- Share content that is aggregated by the app.
- Search aggregated content by using only the Search charm.
- Share RSS feed content by using the Share charm.
- Display general help information by using the Settings charm.

The app must list the name of each social network to which the user subscribes. The app must indicate whether the user is authenticated to that social networking site.

The available data sources will be expanded to include JSON data from a third-party social networking site that is hosted by Litware, Inc. An SSL connection to the Litware social network is available.

Technical Requirements

The app has the following technical requirements:

- Retrieve user data from the social network services by using the authentication credentials.
- When making an HTTP request for content, read all content prior to acting on the response.
- When SSL is available, use SSL to retrieve data from social network providers.

The code that is used to retrieve data from RSS feeds must be reusable.

The app must display the information about the user's social network subscriptions in a layout control. The app must display authentication screens from the social networking sites when an authentication screen is available.

The custom XAML code that was provided by the design team must be available for all

ListView controls in the app.

Data from the FeedRetriever class must be presented in a data control.

Two developers will create the SocialRetriever class, with the following assignments:

- Developer1 must update methods for getting data.
- Developed must implement three new methods for exposing data to the user interface.

All methods must be self-contained and must not affect other methods in the SocialRetriever class. Multiple developers must not work in the same file at the same time.

NewsItem.cs

```
NI01 using System;
NI02 using System.Text;
NI03 namespace NewsReader.Code
NI04 {
NI05   public class NewsItem
NI06   {
NI07     public string Title { get; set; }
NI08     public StringBuilder Author { get; set; }
NI09     public string Content { get; set; }
NI10     public DateTime PubDate { get; set; }
NI11     public Uri Link { get; set; }
NI12     public string Summary { get; set; }
NI13   }
NI14 }
```

NewsSource.es

```
NS01 using System;
NS02 using System.Collections.Generic;
NS03 namespace NewsReader.Code
NS04 {
NS05   public class NewsSource
NS06   {
NS07     public string Title { get; set; }
NS08     public string Description { get; set; }
NS09     public DateTime PublicationDate { get; set; }
NS10     public string Image { get; set; }
NS11     private List<NewsItem> _items = new List<NewsItem>();
NS12     public List<NewsItem> Items
NS13     {
NS14       get
NS15       {
NS16         return this._items;
NS17       }
NS18     }
NS19   }
NS20 }
```

FcedRetriever.es

```
FR01 using System;
FR02 using System.Collections.Generic;
FR03 using System.Collections.ObjectModel;
FR04 using System.Threading.Tasks;
FR05 using Windows.Web.Syndication;
FR06 namespace NewsReader.Code
FR07 {
FR08   public class FeedRetriever
FR09   {
FR10     private ObservableCollection<NewsSource> _news = new ObservableCollection<NewsSource>();
FR11     public ObservableCollection<NewsSource> News
FR12     {
FR13       get { return this._news; }
FR14     }
FR15     public async Task GetNewsSources(List<string> addresses)
FR16     {
FR17       ...
FR18     }
FR19     private async Task<NewsSource> GetNewsSourceAsync(string address)
FR20     {
FR21       NewsSource source = new NewsSource();
FR22       try
FR23       {
FR24         ...
FR25       }
FR26       catch (Exception ex)
FR27       {
FR28         throw ex;
FR29       }
FR30       return source;
FR31     }
FR32   }
FR33 }
```

SocialItem.es

```
SI01 using System;
SI02 namespace NewsReader.Code
SI03 {
SI04   public class SocialItem
SI05   {
SI06     public string ProfileImgUrl { get; set; }
SI07     public string Content { get; set; }
SI08     public DateTime PostTime { get; set; }
SI09     public Uri Link { get; set; }
SI10   }
SI11 }
```

SocialSources.cs

```
SS01 using System;
SS02 using System.Collections.Generic;
SS03 namespace NewsReader.Code
SS04 {
SS05   public class SocialSource
SS06   {
SS07     public string Name { get; set; }
SS08     public Uri RequestUri { get; set; }
SS09     public Uri CallbackUri { get; set; }
SS10     public bool isAuthenticated { get; set; }
SS11     private List<SocialItem> _items = new List<SocialItem>();
SS12     public List<SocialItem> Items
SS13     {
SS14       get
SS15       {
SS16         if (this._items == null)
SS17           this._items = new List<SocialItem>();
SS18         return this._items;
SS19       }
SS20     }
SS21   }
SS22 }
```

SocialRetriever.es

```
SR01 using System;
SR02 using System.Collections.Generic;
SR03 using System.Collections.ObjectModel;
SR04 using System.Net.Http;
SR05 using System.Threading.Tasks;
SR06 using Windows.Security.Authentication.Web;
SR07 namespace NewsReader.Code
SR08 {
SR09   public class SocialRetriever
SR10   {
SR11     private ObservableCollection<SocialSource> _social = new
SR12      ObservableCollection<SocialSource>();
SR13     public ObservableCollection<SocialSource> SocialFeeds
SR14     {
SR15       get
SR16       {
SR17         return this._social;
SR18       }
SR19     }
SR20     public async Task GetSocialSources(List<string> socialNetworks)
SR21     {
SR22       foreach (string network in socialNetworks)
SR23       {
SR24         SocialSource source = new SocialSource();
SR25         switch (network)
SR26         {
SR27           case "Contoso":
SR28             string contosoUrl = "https://www.contoso.com/auth/oauth";
SR29             string clientId = "1234";
SR30             source.CallbackUri = new Uri("https://www.contoso.com/auth/login_success.html");
SR31             source.RequestUri = new Uri(
```

```
SR31               source.RequestUri = new Uri(
SR32               string.Format("{0}?client_id={1}&redirect_uri={2}&response_type=token",
SR33                 contosoUrl,
SR34                 clientId,
SR35                 source.CallbackUri),
SR36                 UriKind.RelativeOrAbsolute);
SR37               source.Name = "Contoso Social";
SR38               WebAuthenticationResult authenticationResult = await
SR39                WebAuthenticationBroker.AuthenticateAsync(
SR40
SR41                 source.CallbackUri
SR42               );
SR43               switch (authenticationResult.ResponseStatus)
SR44               {
SR45                 case WebAuthenticationStatus.Success:
SR46                   source.isAuthenticated = true;
SR47                   this._social.Add(source);
SR48                   break;
SR49                 case WebAuthenticationStatus.ErrorHttp:
SR50                   throw new Exception("Error occurred while authenticating");
SR51                   break;
SR52                 case WebAuthenticationStatus.UserCancel:
SR53                   source.isAuthenticated = false;
SR54                   break;
SR55               }
SR56               break;
SR57             case "Litware Inc.":
SR58
SR59               break;
SR60             case "Northwind":
SR61               ...
SR62               break;
SR63           }
SR64       }
SR65     }
SR66   }
SR67 }
```

Question No : 17 - (Topic 6)

You need to make available the content that is provided by the design team.

Which markup segment should you use?

A.
```
<ListView x:Name="lvSocial" ItemsSource="{Binding SocialFeeds}">
  <ListView.ItemTemplate>
    <DataTemplate Name="{StaticResource newTemplate}" />
  </ListView.ItemTemplate>
</ListView>
```

B.
```
<ListView x:Name="lvSocial" ItemTemplate="{StaticResource newTemplate}"
 ItemsSource="{Binding News}" />
```

C.
```
<ListView x:Name="lvSocial" ItemTemplate="{StaticResource newTemplate}"
 ItemsSource="{Binding SocialFeeds}" />
```

D.
```
<ListView x:Name="lvSocial" ItemsSource="{Binding SocialFeeds}">
  <ListView.ItemTemplate Name="{StaticResource newTemplate}" />
</ListView>
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: B

Question No : 18 - (Topic 6)

You need to create a custom template for a Listview control that will be located on a page that has the NewsSource object bound to the DataContext property.

Which code segment should you use?

A.
```
<ListView x:Name="lvNews" ItemsSource="{Binding NewsFeeds}">
  <ListView.ItemTemplate>
    <StackPanel>
      <TextBlock Text="{Binding Name}" FontSize="24" Margin="5,0,0,0" />
    </StackPanel>
  </ListView.ItemTemplate>
</ListView>
```

B.
```
<ListView x:Name="lvNews" ItemsSource="{Binding Items}">
  <ListView.ItemTemplate>
    <StackPanel>
      <TextBlock Text="{Binding Item.Content}" FontSize="24" Margin="5,0,0,0" />
    </StackPanel>
  </ListView.ItemTemplate>
</ListView>
```

C.
```
<ListView x:Name="lvNews" ItemsSource="{Binding Items}">
  <ListView.ItemTemplate>
    <DataTemplate>
      <StackPanel>
        <TextBlock Text="{Binding Content}" FontSize="24" Margin="5,0,0,0" />
      </StackPanel>
    </DataTemplate>
    </ListView.ItemTemplate>
  </ListView>
```

D.
```
<ListView x:Name="lvNews" ItemsSource="{Binding NewsFeeds}">
  <ListView.ItemTemplate>
    <DataTemplate>
      <StackPanel>
        <TextBlock Text="{Binding Name}" FontSize="24" Margin="5,0,0,0" />
      </StackPanel>
    </DataTemplate>
  </ListView.ItemTemplate>
</ListView>
```

A. Option A
B. Option B
C. Option C
D. Option D

Answer: C

Question No : 19 - (Topic 6)

You need to authenticate to a social networking site.

Which code segments should you insert at line SR40? (Each correct answer presents part of the solution. Choose all that apply.)

A. WebAuthenticationOptions.SilentMode,
B. source.RequestUri,
C. source.RequestUri.SecureString,
D. WebAuthenticationOptions.None,

Answer: A,C

Explanation: A (not D): Scenario: Retrieve user data from the social network services by using the authentication credentials.

*

C (not B): Scenario: When SSL is available, use SSL to retrieve data from social network providers.

Question No : 20 - (Topic 6)

You need to implement storage for the user preferences.

Which storage solutions can you use to meet the specification for the app? (Each correct answer presents a complete solution. Choose all that apply.)

A. TheWindows.Storage.StorageItemType object
B. TheWindows.Storage.ApplicationData.Current.RoamingSettings object
C. Windows Azure
D. The await Windows.Storage.ApplicationData.Current.LocalFolder method

Answer: A,D

Explanation: * Scenario: The app will run locally on the user's device. User preferences will be available locally.

Topic 7, Scenario 7

Background

You are developing a Windows Store media sharing app for the sates and marketing team at Margie's Travel. The app will allow team members to download documents and media about current and proposed products and services from the company's cloud-based media manager service. Team members will be able to add new content to the cloud service and to print and share content.

Business Requirements

Behavior:

- Team members must be able to download product information data sheets, marketing materials, and product demonstration video clips from the company's server.
- Team members must be able to select and upload multiple files that contain new and modified content as a batch.
- Team members must be able to stream video clips to other devices in the vicinity of the team member's device. The app will not support the streaming of photographs.
- The app must allow team members to pause, restart, or cancel uploads and downloads of files. The app must report both the progress and completion status of these operations. It must also return results about upload and download operations.

User Interface:

- The app must include a photo viewer. When photos are added or deleted in the photo viewer window, they must animate in and out of the field of view. Remaining photos must move to fill the empty space created when photos are deleted. The photo viewer must support semantic zoom.
- The app must display information on the lock screen of the device. The information must include text-based alerts and a value indicating the number of pending file downloads.

Technical Requirements

Behavior:

- The company has an existing component named VideoProcessor. This component compresses video clips and performs other processing before the video clips are uploaded to the media manager service. The component was written with managed code. The VideoProcessor component will also be used by Windows Store apps developed in HTML5 and JavaScript. The apps must be able to call the overload of the ProcessVideoO method that accepts a string and a Boolean value as parameters.
- When a team member selects a video clip to download, the app must download the file as a background task. After a download has started, the app should maintain the network connection to the server even when the app is suspended.

User Interface:

- The app must include a custom photo viewer control. The control will be updated frequently and may be deployed separately from the rest of the app. The photo viewer control must support templates and styles.
- The app must use a Grid control as the root layout control. The photo viewer must be placed in the second row of the grid.
- The appearance of the app must change when the app is not in full screen mode. The first row of the root layout grid must not change height- The second row must fill all available space.
- Available video clips must be displayed in an extended ListView control class named DownloadedVideoList
- The template for the DownloadedVideoList is already defined.
- New video clips should be added to DownloadedVideoList when the

DownloadVideo() method completes.

- New video clip items in the DownloadedVideoList should color change periodically to alert the team member.

Application Structure

Relevant portions of the app files are as follows. (Line numbers in the code segments are included for reference only and include a two-character prefix that denotes the specific file to which they belong.)

App.xaml.cs

```
AP01 cts = new CancellationTokenSource();
AP02 private List<DownloadOperation> MyPendingDownloads;
AP03 private async Task HandleMyPendingDownloads(DownloadOperation download, bool start)
AP04 {
AP05   MyPendingDownloads.Add(download);
AP06   Progress<DownloadOperation> progressCallback = new Progress<DownloadOperation>(DownloadProgress);
AP07   if (start)
AP08   {
AP09     await download.StartAsync().AsTask(cts.Token, progressCallback);
AP10   }
AP11   else
AP12   {
AP13     await download.AttachAsync().AsTask(cts.Token, progressCallback);
AP14   }
AP15 }
AP16 private async void UploadContent()
AP17 {
AP18   FileOpenPicker picker = new FileOpenPicker();
AP19
AP20   List<BackgroundTransferContentPart> uploadGrp = new List<BackgroundTransferContentPart>();
AP21   for (int fileNum = 0; fileNum < files.Count; fileNum ++)
AP22   {
AP23     BackgroundTransferContentPart uploadItem = new BackgroundTransferContentPart("File" + fileNum,
          files[fileNum].Name);
AP24     uploadItem.SetFile(files[fileNum]);
AP25     uploadGrp.Add(uploadItem);
AP26   }
AP27   BackgroundUploader uploader = new BackgroundUploader();
AP28
AP29   await HandleUploadAsync(upload, true);
AP30 }
```

VideoProcessor.es

```
IP01 public class VideoProcessor
IP02 {
IP03
IP04    public VideoProcessor(int videoID)
IP05    {
IP06      ...
IP07    }
IP08
IP09    public VideoProcessor(string videoName)
IP10    {
IP11      ...
IP12    }
IP13
IP14
IP15    public void ProcessVideo(string videoName, string videoType)
IP16    {
IP17      ...
IP18    }
IP19
IP20    public void ProcessVideo(string videoName, bool compressFile)
IP21    {
IP22      ...
IP23    }
IP24 }
```

MainPage.xaml

```
MP01 <Grid x:Name="LayoutRoot">
MP02    <Grid.RowDefinitions>
MP03      <RowDefinition Height="100"/>
MP04      <RowDefinition Height="200"/>
MP05    </Grid.RowDefinitions>
MP06    <VisualStateManager.VisualStateGroups>
MP07
MP08    </VisualStateManager.VisualStateGroups>
MP09 </Grid>
```

MainPage.xaml.es

Microsoft Exams List

70-246 Dump PDF VCE	70-485 Dump PDF VCE	70-742 Dump PDF VCE	98-366 Dump PDF VCE
70-247 Dump PDF VCE	70-486 Dump PDF VCE	70-743 Dump PDF VCE	98-367 Dump PDF VCE
70-331 Dump PDF VCE	70-487 Dump PDF VCE	70-744 Dump PDF VCE	98-368 Dump PDF VCE
70-332 Dump PDF VCE	70-488 Dump PDF VCE	70-761 Dump PDF VCE	98-369 Dump PDF VCE
70-333 Dump PDF VCE	70-489 Dump PDF VCE	70-762 Dump PDF VCE	98-372 Dump PDF VCE
70-334 Dump PDF VCE	70-490 Dump PDF VCE	70-765 Dump PDF VCE	98-373 Dump PDF VCE
70-339 Dump PDF VCE	70-491 Dump PDF VCE	70-768 Dump PDF VCE	98-374 Dump PDF VCE
70-341 Dump PDF VCE	70-492 Dump PDF VCE	70-980 Dump PDF VCE	98-375 Dump PDF VCE
70-342 Dump PDF VCE	70-494 Dump PDF VCE	70-981 Dump PDF VCE	98-379 Dump PDF VCE
70-345 Dump PDF VCE	70-496 Dump PDF VCE	70-982 Dump PDF VCE	MB2-700 Dump PDF VCE
70-346 Dump PDF VCE	70-497 Dump PDF VCE	74-343 Dump PDF VCE	MB2-701 Dump PDF VCE
70-347 Dump PDF VCE	70-498 Dump PDF VCE	74-344 Dump PDF VCE	MB2-702 Dump PDF VCE
70-348 Dump PDF VCE	70-499 Dump PDF VCE	74-409 Dump PDF VCE	MB2-703 Dump PDF VCE
70-354 Dump PDF VCE	70-517 Dump PDF VCE	74-678 Dump PDF VCE	MB2-704 Dump PDF VCE
70-383 Dump PDF VCE	70-532 Dump PDF VCE	74-697 Dump PDF VCE	MB2-707 Dump PDF VCE
70-384 Dump PDF VCE	70-533 Dump PDF VCE	77-420 Dump PDF VCE	MB2-710 Dump PDF VCE
70-385 Dump PDF VCE	70-534 Dump PDF VCE	77-427 Dump PDF VCE	MB2-711 Dump PDF VCE
70-410 Dump PDF VCE	70-640 Dump PDF VCE	77-600 Dump PDF VCE	MB2-712 Dump PDF VCE
70-411 Dump PDF VCE	70-642 Dump PDF VCE	77-601 Dump PDF VCE	MB2-713 Dump PDF VCE
70-412 Dump PDF VCE	70-646 Dump PDF VCE	77-602 Dump PDF VCE	MB2-714 Dump PDF VCE
70-413 Dump PDF VCE	70-673 Dump PDF VCE	77-603 Dump PDF VCE	MB2-715 Dump PDF VCE
70-414 Dump PDF VCE	70-680 Dump PDF VCE	77-604 Dump PDF VCE	MB2-716 Dump PDF VCE
70-417 Dump PDF VCE	70-681 Dump PDF VCE	77-605 Dump PDF VCE	MB2-717 Dump PDF VCE
70-461 Dump PDF VCE	70-682 Dump PDF VCE	77-881 Dump PDF VCE	MB2-718 Dump PDF VCE
70-462 Dump PDF VCE	70-684 Dump PDF VCE	77-882 Dump PDF VCE	MB5-705 Dump PDF VCE
70-463 Dump PDF VCE	70-685 Dump PDF VCE	77-883 Dump PDF VCE	MB6-700 Dump PDF VCE
70-464 Dump PDF VCE	70-686 Dump PDF VCE	77-884 Dump PDF VCE	MB6-701 Dump PDF VCE
70-465 Dump PDF VCE	70-687 Dump PDF VCE	77-885 Dump PDF VCE	MB6-702 Dump PDF VCE
70-466 Dump PDF VCE	70-688 Dump PDF VCE	77-886 Dump PDF VCE	MB6-703 Dump PDF VCE
70-467 Dump PDF VCE	70-689 Dump PDF VCE	77-887 Dump PDF VCE	MB6-704 Dump PDF VCE
70-469 Dump PDF VCE	70-692 Dump PDF VCE	77-888 Dump PDF VCE	MB6-705 Dump PDF VCE
70-470 Dump PDF VCE	70-695 Dump PDF VCE	77-891 Dump PDF VCE	MB6-884 Dump PDF VCE
70-473 Dump PDF VCE	70-696 Dump PDF VCE	98-349 Dump PDF VCE	MB6-885 Dump PDF VCE
70-480 Dump PDF VCE	70-697 Dump PDF VCE	98-361 Dump PDF VCE	MB6-886 Dump PDF VCE
70-481 Dump PDF VCE	70-698 Dump PDF VCE	98-362 Dump PDF VCE	MB6-889 Dump PDF VCE
70-482 Dump PDF VCE	70-734 Dump PDF VCE	98-363 Dump PDF VCE	MB6-890 Dump PDF VCE
70-483 Dump PDF VCE	70-740 Dump PDF VCE	98-364 Dump PDF VCE	MB6-892 Dump PDF VCE
70-484 Dump PDF VCE	70-741 Dump PDF VCE	98-365 Dump PDF VCE	MB6-893 Dump PDF VCE

Cisco Exams List

010-151 Dump PDF VCE	350-018 Dump PDF VCE	642-737 Dump PDF VCE	650-667 Dump PDF VCE
100-105 Dump PDF VCE	352-001 Dump PDF VCE	642-742 Dump PDF VCE	650-669 Dump PDF VCE
200-001 Dump PDF VCE	400-051 Dump PDF VCE	642-883 Dump PDF VCE	650-752 Dump PDF VCE
200-105 Dump PDF VCE	400-101 Dump PDF VCE	642-885 Dump PDF VCE	650-756 Dump PDF VCE
200-120 Dump PDF VCE	400-151 Dump PDF VCE	642-887 Dump PDF VCE	650-968 Dump PDF VCE
200-125 Dump PDF VCE	400-201 Dump PDF VCE	642-889 Dump PDF VCE	700-001 Dump PDF VCE
200-150 Dump PDF VCE	400-251 Dump PDF VCE	642-980 Dump PDF VCE	700-037 Dump PDF VCE
200-155 Dump PDF VCE	400-351 Dump PDF VCE	642-996 Dump PDF VCE	700-038 Dump PDF VCE
200-310 Dump PDF VCE	500-006 Dump PDF VCE	642-997 Dump PDF VCE	700-039 Dump PDF VCE
200-355 Dump PDF VCE	500-007 Dump PDF VCE	642-998 Dump PDF VCE	700-101 Dump PDF VCE
200-401 Dump PDF VCE	500-051 Dump PDF VCE	642-999 Dump PDF VCE	700-104 Dump PDF VCE
200-601 Dump PDF VCE	500-052 Dump PDF VCE	644-066 Dump PDF VCE	700-201 Dump PDF VCE
210-060 Dump PDF VCE	500-170 Dump PDF VCE	644-068 Dump PDF VCE	700-205 Dump PDF VCE
210-065 Dump PDF VCE	500-201 Dump PDF VCE	644-906 Dump PDF VCE	700-260 Dump PDF VCE
210-250 Dump PDF VCE	500-202 Dump PDF VCE	646-048 Dump PDF VCE	700-270 Dump PDF VCE
210-255 Dump PDF VCE	500-254 Dump PDF VCE	646-365 Dump PDF VCE	700-280 Dump PDF VCE
210-260 Dump PDF VCE	500-258 Dump PDF VCE	646-580 Dump PDF VCE	700-281 Dump PDF VCE
210-451 Dump PDF VCE	500-260 Dump PDF VCE	646-671 Dump PDF VCE	700-295 Dump PDF VCE
210-455 Dump PDF VCE	500-265 Dump PDF VCE	646-985 Dump PDF VCE	700-501 Dump PDF VCE
300-070 Dump PDF VCE	500-275 Dump PDF VCE	648-232 Dump PDF VCE	700-505 Dump PDF VCE
300-075 Dump PDF VCE	500-280 Dump PDF VCE	648-238 Dump PDF VCE	700-601 Dump PDF VCE
300-080 Dump PDF VCE	500-285 Dump PDF VCE	648-244 Dump PDF VCE	700-602 Dump PDF VCE
300-085 Dump PDF VCE	500-290 Dump PDF VCE	648-247 Dump PDF VCE	700-603 Dump PDF VCE
300-101 Dump PDF VCE	500-801 Dump PDF VCE	648-375 Dump PDF VCE	700-701 Dump PDF VCE
300-115 Dump PDF VCE	600-199 Dump PDF VCE	648-385 Dump PDF VCE	700-702 Dump PDF VCE
300-135 Dump PDF VCE	600-210 Dump PDF VCE	650-032 Dump PDF VCE	700-703 Dump PDF VCE
300-160 Dump PDF VCE	600-211 Dump PDF VCE	650-042 Dump PDF VCE	700-801 Dump PDF VCE
300-165 Dump PDF VCE	600-212 Dump PDF VCE	650-059 Dump PDF VCE	700-802 Dump PDF VCE
300-180 Dump PDF VCE	600-455 Dump PDF VCE	650-082 Dump PDF VCE	700-803 Dump PDF VCE
300-206 Dump PDF VCE	600-460 Dump PDF VCE	650-127 Dump PDF VCE	810-403 Dump PDF VCE
300-207 Dump PDF VCE	600-501 Dump PDF VCE	650-128 Dump PDF VCE	820-424 Dump PDF VCE
300-208 Dump PDF VCE	600-502 Dump PDF VCE	650-148 Dump PDF VCE	840-425 Dump PDF VCE
300-209 Dump PDF VCE	600-503 Dump PDF VCE	650-159 Dump PDF VCE	
300-210 Dump PDF VCE	600-504 Dump PDF VCE	650-281 Dump PDF VCE	
300-320 Dump PDF VCE	640-692 Dump PDF VCE	650-393 Dump PDF VCE	
300-360 Dump PDF VCE	640-875 Dump PDF VCE	650-472 Dump PDF VCE	
300-365 Dump PDF VCE	640-878 Dump PDF VCE	650-474 Dump PDF VCE	
300-370 Dump PDF VCE	640-911 Dump PDF VCE	650-575 Dump PDF VCE	
300-375 Dump PDF VCE	640-916 Dump PDF VCE	650-621 Dump PDF VCE	
300-465 Dump PDF VCE	642-035 Dump PDF VCE	650-663 Dump PDF VCE	
300-470 Dump PDF VCE	642-732 Dump PDF VCE	650-665 Dump PDF VCE	
300-475 Dump PDF VCE	642-747 Dump PDF VCE	650-754 Dump PDF VCE	

HOT EXAMS

Cisco

100-105 Dumps VCE PDF
200-105 Dumps VCE PDF
300-101 Dumps VCE PDF
300-115 Dumps VCE PDF
300-135 Dumps VCE PDF
300-320 Dumps VCE PDF
400-101 Dumps VCE PDF
640-911 Dumps VCE PDF
640-916 Dumps VCE PDF

Microsoft

70-410 Dumps VCE PDF
70-411 Dumps VCE PDF
70-412 Dumps VCE PDF
70-413 Dumps VCE PDF
70-414 Dumps VCE PDF
70-417 Dumps VCE PDF
70-461 Dumps VCE PDF
70-462 Dumps VCE PDF
70-463 Dumps VCE PDF
70-464 Dumps VCE PDF
70-465 Dumps VCE PDF
70-480 Dumps VCE PDF
70-483 Dumps VCE PDF
70-486 Dumps VCE PDF
70-487 Dumps VCE PDF

CompTIA

220-901 Dumps VCE PDF
220-902 Dumps VCE PDF
N10-006 Dumps VCE PDF
SY0-401 Dumps VCE PDF