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 : **70-485J**

Title : Advanced Windows Store App Development using C#

Version : DEMO

トピック 1、シナリオマージーの旅

バックグラウンド
あなたはマージーの旅行での販売およびマーケティングのチームのために Windows ストアのメディア共有アプリを開発している。アプリは、チームメンバーが、同社のクラウドベースのメディア・マネージャ・サービスから、現在、提案された製品やサービスについての文書やメディアをダウンロードすることができます。チームメンバーは、クラウドサービスに新しいコンテンツを追加し、コンテンツを印刷して共有することができるようになります。

ビジネス要件
行動:
-チームメンバーは、同社のサーバーから製品情報データシート、マーケティング資料、および製品デモのビデオクリップをダウンロードすることができなければなりません。
-チームメンバーは、バッチとして、新規および変更されたコンテンツを含む複数のファイルを選択してアップロードすることができなければなりません。
-チームメンバーは、チームメンバーのデバイスの近傍に他のデバイスにビデオクリップをストリーミングすることができなければならない。このアプリは、写真のストリーミングをサポートしていません。
-アプリは、チームメンバーが、一時停止して再起動するか、アップロードやファイルのダウンロードをキャンセルできるようにする必要があります。アプリは、進捗状況と、これらの操作の完了状態の両方を報告しなければならない。また、アップロードおよびダウンロード操作に関する結果を返す必要があります。

ユーザ・インタフェース:
-アプリはフォトビューアを含める必要があります。写真は、フォトビューアウィンドウに追加または削除されると、それらは、視野の内外にアニメートする必要があります。残りの写真は、写真が削除されたときに作成され、空のスペースを埋めるために移動する必要があります。フォトビューアは、セマンティックズームをサポートしている必要があります。
-アプリは、デバイスのロック画面に情報を表示する必要があります。情報はテキストベースのアラートや保留中のファイルのダウンロードの数を示す値を含める必要があります。

技術要件
行動:
-同社は、ビデオプロセッサという名前の既存のコンポーネントを持っています。このコンポーネントは、ビデオクリップを圧縮し、ビデオクリップは、メディアマネージャサービスにアップロードされる前に他の処理を実行する。コンポーネントは、マネージコードで書かれていた。ビデオプロセッサーのコンポーネントは、HTML5 と JavaScript で開発された Windows ストアのアプリで使用されます。アプリは、パラメータとして文字列とブール値を受け入れる ProcessVideo（）メソッドのオーバーロードを呼び出すことができなければなりません。
-チームメンバーは、ダウンロードするには、ビデオクリップを選択すると、アプリはバックグラウンドタスクとしてファイルをダウンロードする必要があります。ダウンロードが開始された後、アプリは、アプリが中断されている場合でも、サーバーへのネットワーク接続を維持する必要があります。

ユーザ・インタフェース:
-アプリは、カスタム・フォトビューア・コントロールを含める必要があります。制御は頻繁に更新され、

アプリケーションの残りの部分とは別個に配置することができる。フォトビューアコントロールは、テンプレートとスタイルをサポートする必要があります。

ユーザ・インタフェース：
-アプリは、カスタム・フォトビューア・コントロールを含める必要があります。制御は頻繁に更新され、アプリケーションの残りの部分とは別個に配置することができる。フォトビューアコントロールは、テンプレートとスタイルをサポートする必要があります。
-アプリは、ルートレイアウトコントロールとして Grid コントロールを使用する必要があります。フォトビューアは、グリッドの 2 行目に配置する必要があります。
-アプリは、スナップモードのときにこのアプリの外観は変更する必要があります。ルートレイアウトグリッドの最初の行は、高さを変更してはならない。二行目は、利用可能なすべてのスペースを埋める必要があります。
-利用可能なビデオクリップがダウンロード VideoList 名前付き拡張リストビューコントロールクラスに表示されなければなりません。
- DownloadedVideoList のためのテンプレートがすでに定義されています。
-新しいビデオクリップが DownloadVideoQ メソッドが完了したときに DownloadedVideoList に追加する必要があります。
- DownloadedVideoList で新しいビデオクリップ項目は、チームメンバーに警告するために定期的に変更を彩るする必要があります。

適用構造
次のようにアプリケーション・ファイルの関連する部分である。（コード・セグメント内の行番号は参考情報として含まれており、それらが属する特定のファイルを表す二つ文字の接頭辞を含む。）

App.xaml.cs

```
AP01 cts= newCancellationTokenSource();
AP02 private List<DownloadOperation>MyPendingDownloads;
AP03 privateasyncTaskHandleMyPendingDownloads(DownloadOperationdownload,
boolstart)
AP04 {
AP05   MyPendingDownloads.Add(download);
AP06   Progress<DownloadOperation> progressCallback = new
Progress<DownloadOperation>(DownloadProgress);
AP07   if(start)
AP08   {
AP09     awaitdownload.StartAsync().AsTask(cts.Token, progressCallback);
AP10   }
AP11   else
AP12   {
AP13     awaitdownload.AttachAsync().AsTask(cts.Token, progressCallback);
AP14   }
AP15 }
AP16 privateasyncvoidUploadContent()
AP17 {
AP18   FileOpenPickerpicker = newFileOpenPicker();
AP19
AP20   List<BackgroundTransferContentPart> uploadGrp = new
List<BackgroundTransferContentPart>();
AP21   for(intfileNum = 0; fileNum < files.Count; fileNum ++)
AP22   {
AP23     BackgroundTransferContentPartuploadItem= new
BackgroundTransferContentPart("File"+ fileNum,
          files[fileNum].Name);
AP24     uploadItem.SetFile(files[fileNum]);
AP25     uploadGrp.Add(uploadItem);
AP26   }
AP27   BackgroundUploaderuploader = newBackgroundUploader();
AP28
AP29   awaitHandleUploadAsync(upload, true);
AP30 }
```

VideoProcessor.cs

```
IP01 publicclassVideoProcessor
IP02 {
IP03
IP04   publicVideoProcessor(intvideoID)
IP05   {
IP06     ...
IP07   }
IP08
IP09   publicVideoProcessor(stringvideoName)
IP10   {
IP11     ...
IP12   }
IP13
IP14
IP15   publicvoidProcessVideo(stringvideoName, stringvideoType)
IP16   {
IP17     ...
IP18   }
IP19
IP20   publicvoidProcessVideo(stringvideoName, boolcompressFile)
IP21   {
IP22     ...
IP23   }
IP24 }
```

MainPage.xaml

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

MainPage.xaml.cs

```
MC01 private PlayToManagerptMgr = PlayToManager.GetForCurrentView();
MC02
MC03 protectedoverridevoidOnNavigatedTo(NavigationEventArgse)
MC04 {
MC05
MC06
MC07 }
MC08 privatevoidSourceRequestHandler(PlayToManagersender,
      PlayToSourceRequestedEventArgse)
MC09 {
MC10
MC11    e.SourceRequest.SetSource(mediaElement.PlayToSource);
MC12 }
MC13 publicvoidStartNewVideoAnimation()
MC14 {
MC15    NewVideoStoryboard.Begin();
MC16 }
MC17 publicvoidDownloadVideo(stringvideoName)
MC18 {
MC19    ...
MC20    videoList.Items.Add(videoName);
MC21 }
```

1.HOTSPOT

あなたがダウンロードおよびアップロードに関するビジネス要件を満たす必要があります。

あなたはどのようにアプリ設定する必要がありますか? （答え領域の各ドロップダウンリストから適切なオプションを選択し、応答します。）

Configure the Application UI settings in Package.appxmanifest

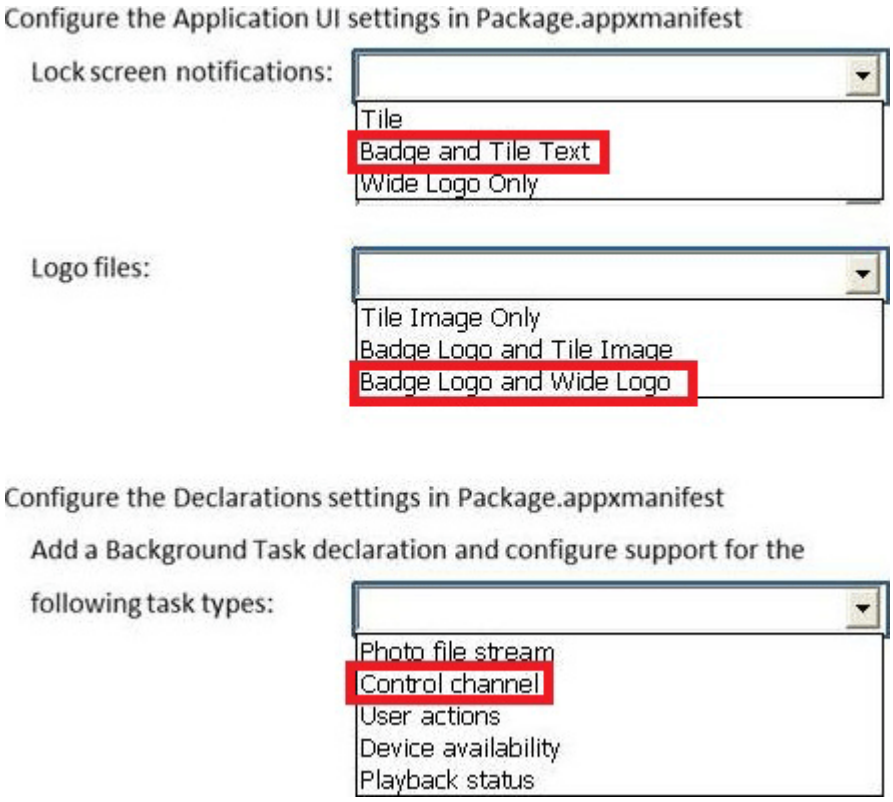
Lock screen notifications: [ ]
- Tile
- Badge and Tile Text
- Wide Logo Only

Logo files: [ ]
- Tile Image Only
- Badge Logo and Tile Image
- Badge Logo and Wide Logo

Configure the Declarations settings in Package.appxmanifest

Add a Background Task declaration and configure support for the following task types: [ ]
- Photo file stream
- Control channel
- User actions
- Device availability
- Playback status

Answer:



2. あなたは、ビデオクリップを表示するビジネス要件を実装する必要があります。
どのコードセグメントは、MainPage.xaml ファイルで使用する必要がありますか?

A.
```
<DownloadedVideoList x:Name="videoList">
  <DownloadedVideoList.Resources>
    <Storyboard x:Name="NewVideoStoryboard">
      <ColorAnimation Storyboard.TargetName="NewVideoBrush"
       Storyboard.TargetProperty="Color" From="Red" To="Green"
       Duration="0:0:8" RepeatBehavior="Forever"/>
    </Storyboard>
  </DownloadedVideoList.Resources>
  <DownloadedVideoList.Background>
    <SolidColorBrush x:Name="NewVideoBrush" Color="Red"/>
  </DownloadedVideoList.Background>
</DownloadedVideoList>
```

B.
```
<DownloadedVideoList x:Name="videoList">
  <DownloadedVideoList.Resources>
    <Storyboard x:Name="NewVideoStoryboard">
      <ColorAnimation Storyboard.TargetName="NewVideoBrush"
       Storyboard.TargetProperty="Color" From="Red" To="Green"
       AutoReverse="true"/>
    </Storyboard>
  </DownloadedVideoList.Resources>
  <DownloadedVideoList.Background>
    <SolidColorBrush x:Name="NewVideoBrush" Color="Red"/>
  </DownloadedVideoList.Background>
</DownloadedVideoList>
```

C.
```
<DownloadedVideoList x:Name="videoList">
  <DownloadedVideoList.Transitions>
    <TransitionCollection>
      <EntranceThemeTransition/>
    </TransitionCollection>
  </DownloadedVideoList.Transitions>
</DownloadedVideoList>
```

D.
```
<DownloadedVideoList x:Name="videoList">
  <DownloadedVideoList.Transitions>
    <TransitionCollection>
      <AddDeleteThemeTransition/>
    </TransitionCollection>
  </DownloadedVideoList.Transitions>
</DownloadedVideoList>
```

A. Option A

B. Option B

C. Option C

D. Option D

Answer: A

3. あなたは、メディアファイル、およびその他のコンテンツのダウンロードを実装する必要があります。どのコードセグメントには App.xaml.cs ファイルに追加しますでしょうか?

A.
```
private async Task GetPendingDownloadsList()
{
  IReadOnlyList<DownloadOperation> downloads = await
   BackgroundDownloader.GetCurrentDownloadsAsync();
  if (downloads.Count > 0)
  {
    List<Task> myTasks = new List<Task>();
    for (int i=0; i < downloads.count; i++)
    {
      await HandleMyPendingDownloads(downloads[i], true);
    }
    await Task.WhenAll(myTasks);
  }
}
```

B.
```
private async Task GetPendingDownloadsList()
{
  IReadOnlyList<DownloadOperation> downloads = await
   BackgroundDownloader.GetCurrentDownloadsAsync();
  if (downloads.Count > 0)
  {
    List<Task> myTasks = new List<Task>();
    foreach (DownloadOperation download in downloads)
    {
      myTasks.Add(HandleDownloadAsync(download, false));
    }
    await Task.WhenAll(myTasks);
  }
}
```

C.
```
private GetPendingDownloadsList()
{
  IReadOnlyList<DownloadOperation> downloads = await
   BackgroundDownloader.GetCurrentDownloadsAsync();
  if (downloads.Count > 0)
  {
    List<Task> myTasks = new List<Task>();
    for (int i=0; i < downloads.count; i++)
    {
      await HandleMyPendingDownloads(downloads[i], true);
    }
    await Task.WhenAll(myTasks);
  }
}
```

D.
```
private Task GetPendingDownloadsList()
{
  IReadOnlyList<DownloadOperation> downloads =
   BackgroundDownloader.CreateDownloadAsync();
  if (downloads.Count > 0)
  {
    List<Task> myTasks = new List<Task>();
    foreach (DownloadOperation download in downloads)
    {
      myTasks.Add(HandleDownloadAsync(download, false));
    }
    Task.WhenAll(myTasks);
  }
}
```

A. Option A

B. Option B
C. Option C
D. Option D
Answer: B

4. あなたが確認する必要があり、そのメディア・マネージャ・サービスにアプリをアップロードするメディアとファイルです。
あなたはどうすればいいのでしょうか? （各正解はソリューションの一部を提供します。当てはまるものをすべて選択してください。）

A. Insert the following line of code at line AP28.

```
IReadOnlyList<UploadOperation> upload =
await BackgroundUploader.GetCurrentUploadsAsync();
```

B. Insert the following line of code at line AP28.

```
UploadOperation upload = await uploader.CreateUpload(uri, uploadGrp);
```

C. Insert the following line of code at line AP28.

```
UploadOperation upload = await uploader.CreateUploadAsync(uri, uploadGrp);
```

D. Insert the following line of code at line AP19.

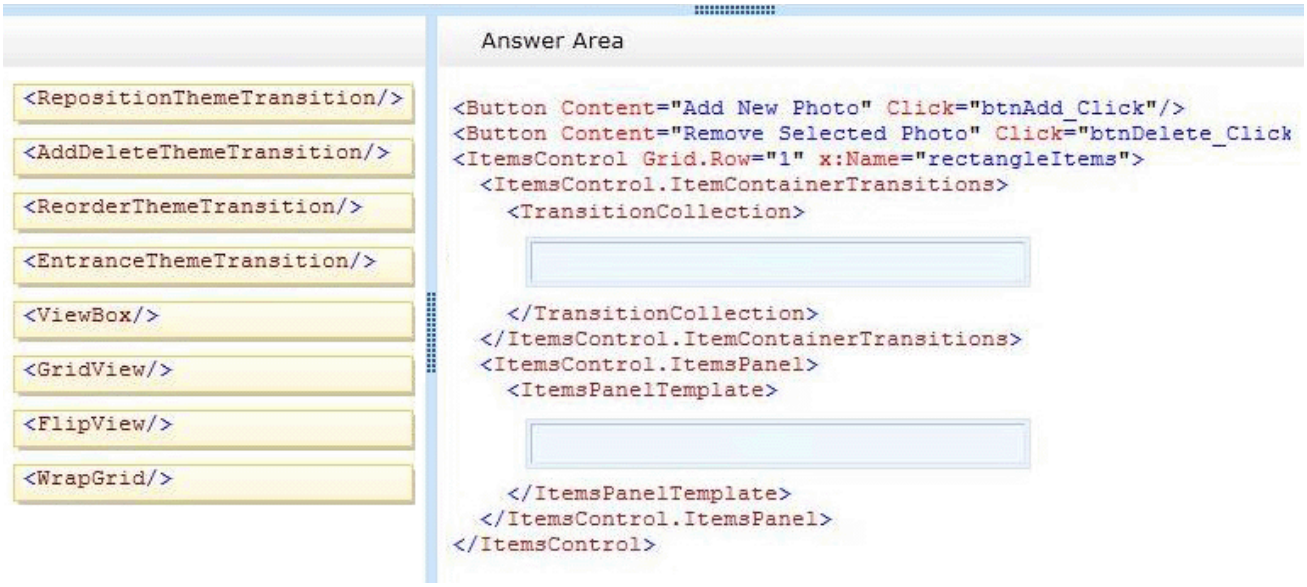
```
IReadOnlyList<StorageFile> files = await
picker.PickMultipleFilesAsync();
```

E. Insert the following line of code at line AP19.

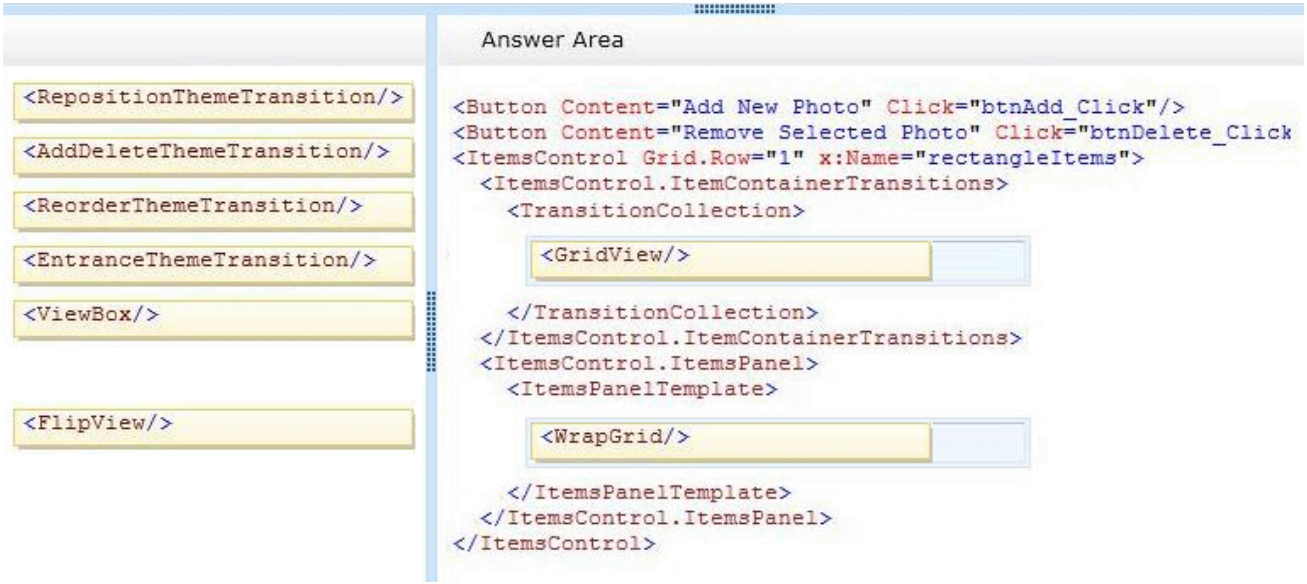
```
IReadOnlyList<StorageFile> files = await
picker.PickSingleFilesAsync();
```

A. Option A
B. Option B
C. Option C
D. Option D
E. Option E
Answer: B, D

5.DRAG DROP
あなたは、ビジネス要件を満たすためにフォトビューア・ページを実装する必要があります。
どのようにコードセグメントを完了する必要がありますか? （答えるために、解答エリアの正しい位置や場所に適切なソースまたはソースをドラッグします。）



Answer:



6. あなたは、ビデオクリップのサムネイル画像を表示するカスタムコントロール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用する必要がありますか?

A.
```
public sealed class DownloadedVideoList: FlipView
{
  public DownloadedVideoList()
  {
    this.DefaultStyleKey = typeof(ListView);
  }
}
```

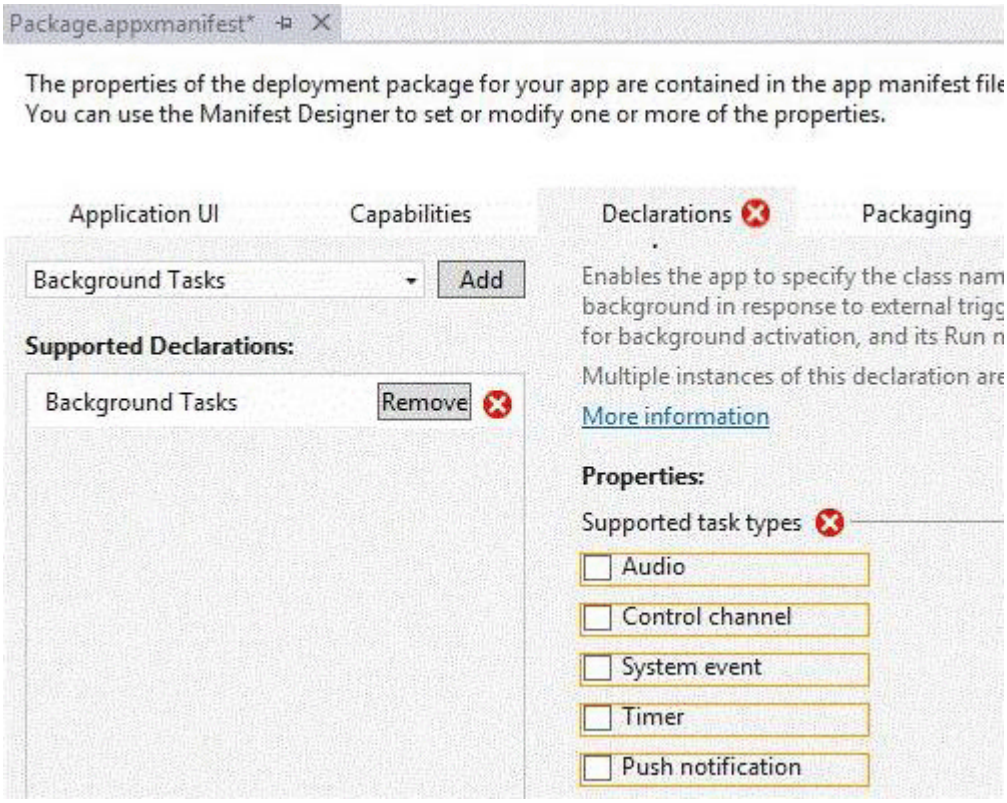
B.
```
public sealed class DownloadedVideoList: FlipView
{
  public DownloadedVideoList()
  {
    this.DefaultStyleKey = typeof(DownloadedVideoList);
  }
}
```

C.
```
public sealed class DownloadedVideoList: ListView
{
  public DownloadedVideoList()
  {
    this.DefaultStyleKey = typeof(DownloadedVideoList);
  }
}
```

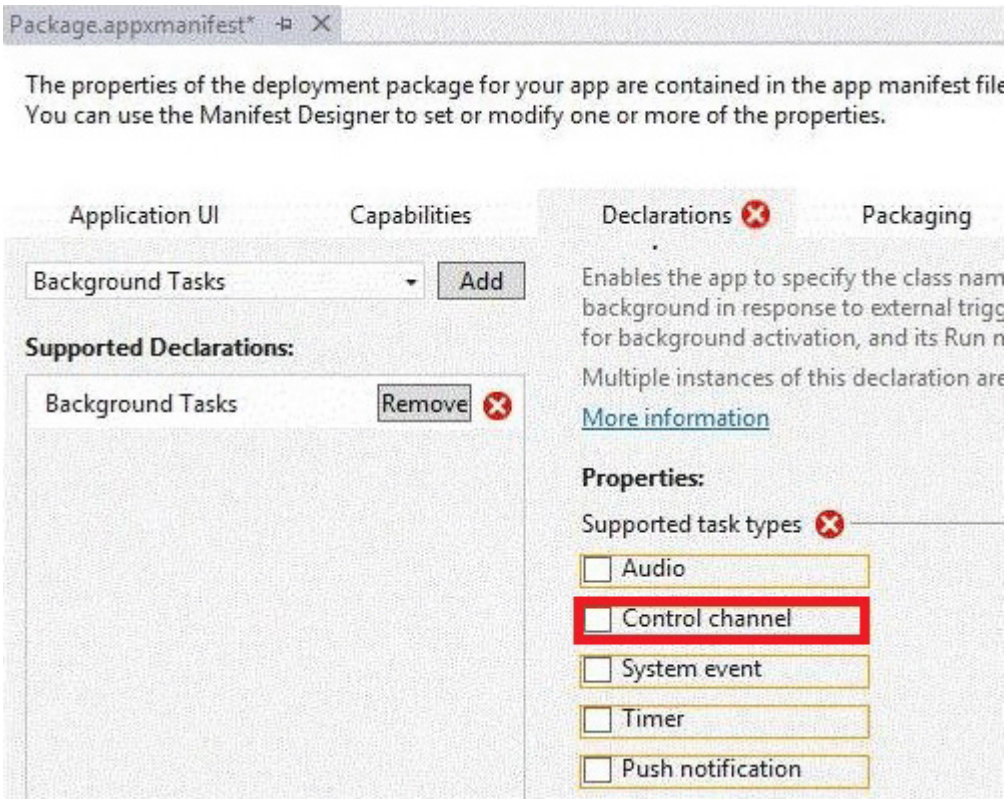
D.
```
public sealed class DownloadedVideoList: ListView
{
  public DownloadedVideoList()
  {
    this.DefaultStyleKey = typeof(ListView);
  }
}
```

A. Option A
B. Option B
C. Option C
D. Option D
Answer: C

7.HOTSPOT
あなたは、ファイルのダウンロードの要求をサポートするために、アプリマニフェストを構成する必要があります。
あなたはどのタスクの種類のプロパティを指定する必要がありますか? （解答エリアに適切なプロパティを選択し、応答します。）



Answer:



8. あなたはメディアの再生のための要件を実装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？（各正解はソリューションの一部を提供します。当てはまるも

のをすべて選択してください。)
A. ライン MC02 での次のコード行を追加します。
private void ShowPlayTo()
{
Windows.Media.PlayTo.PlayToManager.ShowPlayToUI();
}
B. ライン MC06 での次のコード行を追加します。
ptMgr.DefauitSourceSelection = false;
C. ライン MC10 での次のコード行を追加します。
ptMgr.PlayRequested += SourceRequestHandler;
D. ライン MC05 での次のコード行を追加します。
ptMgr.SourceRequested += SourceRequestHandler;
Answer: B, D

9. あなたは、メインページの動作のための要件を実装する必要があります。
どのコードセグメントは、あなたがライン MP07 に挿入する必要がありますか?

A.
```
<VisualStateGroupx:Name="ApplicationViewStates">
  <VisualStatex:Name="Snapped">
    <Storyboard>
      <ObjectAnimationUsingKeyFrames Storyboard.TargetName="LayoutRoot"
       Storyboard.TargetProperty="(Grid.RowDefinitions)[1].Height">
        <DiscreteObjectKeyFrameKeyTime="0"Value="Auto"/>
      </ObjectAnimationUsingKeyFrames>
    </Storyboard>
  </VisualState>
</VisualStateGroup>
```

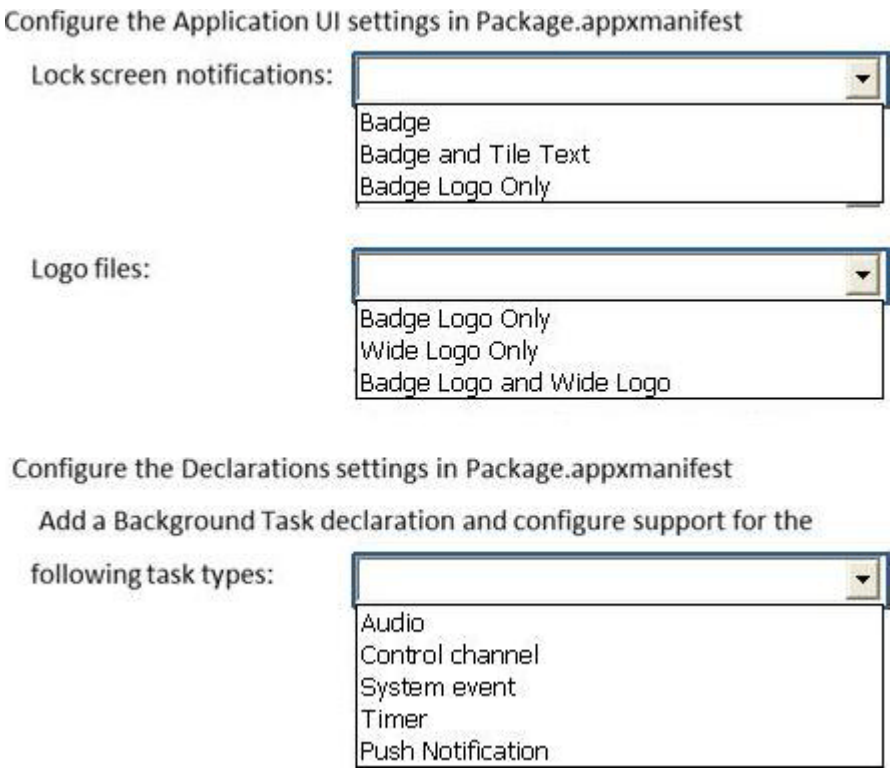
B.
```
<VisualStateGroup x:Name="ApplicationViewStates">
  <VisualState x:Name="Filled">
    <Storyboard>
      <ObjectAnimationUsingKeyFrames Storyboard.TargetProperty="LayoutRoot">
        <DiscreteObjectKeyFrame KeyTime="0" Value="*"/>
      </ObjectAnimationUsingKeyFrames>
    </Storyboard>
  </VisualState>
</VisualStateGroup>
```

C.
```
<VisualStateGroupx:Name="ApplicationViewStates">
  <VisualStatex:Name="FullScreenLandscape">
    <Storyboard>
      <ObjectAnimationUsingKeyFrames
       Storyboard.TargetProperty="LayoutRoot">
        <DiscreteObjectKeyFrameKeyTime="0"Value="Auto"/>
      </ObjectAnimationUsingKeyFrames>
    </Storyboard>
  </VisualState>
</VisualStateGroup>
```

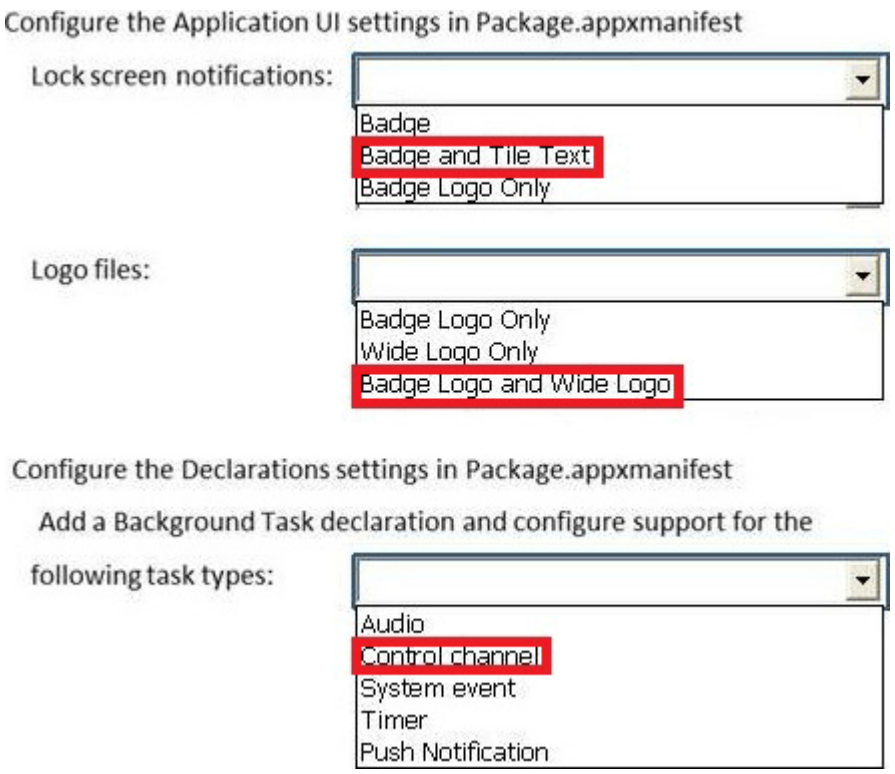
D.
```
<VisualStateGroup x:Name="ApplicationViewStates">
  <VisualState x:Name="FullScreenPortrait">
    <Storyboard>
      <ObjectAnimationUsingKeyFrames Storyboard.TargetName="LayoutRoot"
       Storyboard.TargetProperty="(Grid.RowDefinitions).Height">
        <DiscreteObjectKeyFrame KeyTime="0" Value="*"/>
      </ObjectAnimationUsingKeyFrames>
    </Storyboard>
  </VisualState>
</VisualStateGroup>
```

A. Option A
B. Option B
C. Option C
D. Option D
Answer: A

10.HOTSPOT
あなたがダウンロードおよびアップロードに関するビジネス要件を満たす必要があります。
あなたはどのようにアプリ設定する必要がありますか？（答え領域の各ドロップダウンリストから適切なオプションを選択し、応答します。）



Answer:



11. あなたは、ビデオプロセッサコンポーネントは Windows ストアのアプリで使用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？（各正解はソリューションの一部を提供します。当てはまるものをすべて選択してください。）

A. IP19 行目に次の属性を追加します。
[Windows.Foundation.Metadata.DefaultOverload()]
B. 次のコード行と行 IP01 を交換してください。
静的クラスビデオプロセッサー
C. 私の U ラインを交換してください
次のコード行で P09。
PublicVideoProcessor(string videoName, int ID)
D. IP14 行目に次の属性を追加します。
[Windows.Foundation.Metadata.DefaultOverload()]
E. 次のコード行と行 IP01 を交換してください。
公共シールクラスのビデオプロセッサー
Answer: A, C, E

12. あなたは、ファイルのアップロードとダウンロードについての情報を提供するためのビジネス要件を実装する必要があります。
どのコードセグメントは VideoProcessor.es クラスで使用する必要がありますか?

A.
```
public static IAsyncOperationWithProgress<TResult, TProgress>
Run<TResult,
TProgress>(
 Func<CancellationToken, IProgress<TProgress>, Task<TResult>>
 taskProvider)
{
  ...
}
```

B.
```
public static IAsyncActionWithProgress<TProgress> Run<TProgress>(
 Func<CancellationToken, IProgress<TProgress>,Task> taskProvider)
{
  ...
}
```

C.
```
public interface IAsyncOperation<TResult> : IAsyncInfo
{
  AsyncOperationCompletedHandler<TResult> Completed { get; set; }
  TResult GetResults();
}
```

D.
```
public interface IAsyncActionWithProgress<TProgress> : IAsyncInfo
{
  AsyncActionWithProgressCompletedHandler<TProgress>Completed{get;set;}
  AsyncActionProgressHandler<TProgress> Progress { get; set; }
  void GetResults();
}
```

E.
```
public static IAsyncOperation<TResult> Run<TResult>(
 Func<CancellationToken,  Task<TResult>> taskProvider)
{
  ...
}
```

A. Option A
B. Option B
C. Option C
D. Option D
Answer: A

13. あなたはメディアをストリーミングするための要件を実装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? （各正解はソリューションの一部を提供します。当てはまるものをすべて選択してください。）
A. ビデオライブラリへのアクセスを有効にします。
B. メディアがストリーミングされている間、アプリケーションがフォアグラウンドに留まることを確認してください。
C. 写真ライブラリへのアクセスを有効にします。
D. SourceRequested イベントを登録します。
E. ミュージックライブラリへのアクセスを有効にします。
F. PlayRequested イベントを登録します。
Answer: A, D
説明：
シナリオから：
チームメンバーは、チームメンバーのデバイスの近傍に他のデバイスにビデオクリップをストリーミングすることができなければならない。このアプリは、写真のストリーミングをサポートしていません。
D: あなたは縮小してプレイを実装することにより、オーディオやビデオのアプリケーション内だけでなく、画像をストリーミングするためにプレーを使用することができます。アプリケーション内で収縮させるプレーを実装 sourceRequested のイベントに登録する。
注意：
sourceRequested イベントを登録するには、getForCurrentView メソッドを呼び出して、現在の PlayToManager への参照を取得します。その後 sourceRequested イベントにイベントハンドラを関連付けること PlayToManager に addEventHandler を呼び出すことができます。イベントハンドラでは、PlayToSourceRequestedEventArgs の SetSource メソッドにアプリケーションからのメディア要素を渡す次の例に示すように、イベントハンドラに渡されるオブジェクト。

```
//縮めるためにプレー
private Windows.Media.PlayTo.PlayToManager ptm =
Windows.Media.PlayTo.PlayToManager.GetForCurrentView();
protected override void OnNavigatedTo(NavigationEventArgs e)
{
    ptm.SourceRequested += sourceRequestHandler;
}
private void sourceRequestHandler(
Etc.
```

14. あなたは、フォトビューアの動作要件を実装する必要があります。
それはあなたが作成する必要がコントロールですか?
A. 2 SemanticZoom のコントロールと 1 ListView コントロールを作成します。
B. 1 SemanticZoom のコントロールと 1 ListView コントロールを作成します。

C. 1 ScrollViewer コントロール、1 SemanticZoom の制御、および 1 の GridView コントロールを作成します。
D. 2 の GridView コントロールと 1 SemanticZoom のコントロールを作成します。
Answer: D