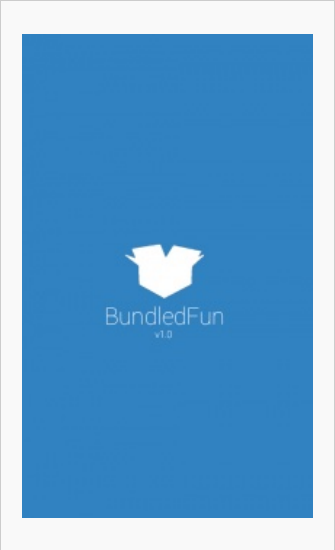
How to create a Video/Audio/Image and Text based Quiz Game for Windows Phone

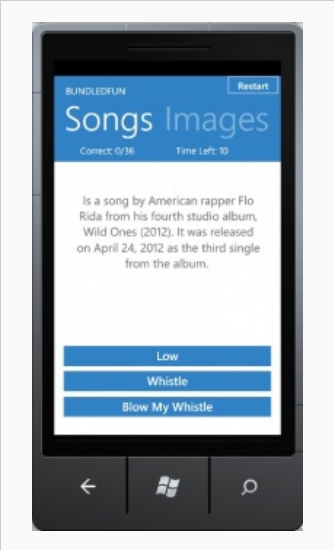
This article will help you to create a Video/Audio/Image and Text based Quiz Game for Windows Phone

Introduction

We will build a very nice, quick and unique quiz game like this below.



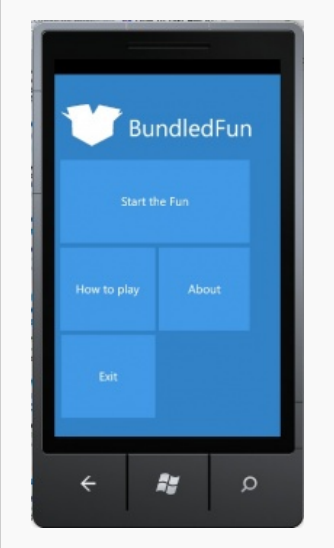
Splash Screen Page



Songs Page

Trailers Page

Images Page



Menu Page

How To Page

(c) Copyright Nokia 2013. All rights reserved.

About Page

Prerequisite

- You should be a developer already.
- Knows how to program
- Have understanding for OOP (Object Oriented Programming)
- Basic understanding of XAML (or XML), C# and Windows Phone

Overview

Implementation

1. Create an empty Windows Phone Project. Launch **Visual Studio -> click on File->New Project**. Select **Windows Phone Pivot App (Visual C# Template)**. Add Name and Location of the project then click **OK** to create the project.
   Now we have blank application.
2. Open '*MainPage.xaml* and rename your application

```
<controls:Pivot Title="YOUR APPLICATION NAME">
```

3. Delete all `PivotItem` (as listed below) and replace with the content in http://pastebin.com/s544iYvE - MainPage.xaml

```
<!--Pivot item one-->
            <controls:PivotItem Header="first">
                <!--Double line list with text wrapping-->
                <ListBox x:Name="FirstListBox" Margin="0,0,-12,0" ItemsSource="{Binding Items}">
                    <ListBox.ItemTemplate>
                        <DataTemplate>
                          <StackPanel Margin="0,0,0,17" Width="432" Height="78">
                              <TextBlock Text="{Binding LineOne}" TextWrapping="Wrap" Style="{StaticResource PhoneTextExtraLa
                              <TextBlock Text="{Binding LineTwo}" TextWrapping="Wrap" Margin="12,-6,12,0" Style="{StaticResou
```


(c) Copyright Nokia 2013. All rights reserved.

```
                        </StackPanel>
                    </DataTemplate>
                </ListBox.ItemTemplate>
            </ListBox>
        </controls:PivotItem>

        <!--Pivot item two-->
        <controls:PivotItem Header="second">
            <!--Triple line list no text wrapping-->
                <ListBox x:Name="SecondListBox" Margin="0,0,-12,0" ItemsSource="{Binding Items}">
                    <ListBox.ItemTemplate>
                        <DataTemplate>
                            <StackPanel Margin="0,0,0,17">
                                <TextBlock Text="{Binding LineOne}" TextWrapping="NoWrap" Margin="12,0,0,0" Style="{Stati
                                <TextBlock Text="{Binding LineThree}" TextWrapping="NoWrap" Margin="12,-6,0,0" Style="{St
                            </StackPanel>
                        </DataTemplate>
                    </ListBox.ItemTemplate>
                </ListBox>
        </controls:PivotItem>
```

Now we already have a nice looking UI, *But you can't run it yet because we still don't have event handlers for the buttons. If you try to run, it will give you errors so better follow next step first.*

4. Let's create our Questions class: This class will be an object for every question to hold every piece of data.
   - Right click on your Application Name on the Solution Explorer
   - Click '*Add New Item*. Then select Class and name your new class "Question".
5. Open the new **Question.cs** and paste in the code below:

```
public class Question
    {
        public String text { get; set; }
        public ImageSource image { get; set; }
        public String trailer { get; set; }
        public String song { get; set; }
        public String a { get; set; }
        public String b { get; set; }
        public String c { get; set; }
        public String answer { get; set; }
        public int timer { get; set; }
    }
```

Now our class is ready for use.

6. Now open up your **MainPage.cs** replace and paste the code in the following file: http://pastebin.com/g7kFYDfx - MainPage.cs
7. Create the required folders in your solution.
   - Trailers
   - Songs
   - Images
8. Then fill your folders with media using these templates for images

```
question = null;
question = new Question();
question.image = new BitmapImage(new Uri("Images/xxx.png", UriKind.Relative));
question.text = "xxx";
question.a = "xxx";
question.b = "xxx";
question.c = "xxx";
question.answer = "xxx";
question.timer = 30;
questions_Images.Add(question);
```

   - for trailers / videos

```
question = null;
question = new Question();
question.trailer = "xxx";
question.text = "xxx";
question.a = "xxx";
question.b = "xxx";
question.c = "xxx";
question.answer = "xxx";
question.timer = 30;
questions_Trailers.Add(question);
```

   - for audio

```
question = null;
question = new Question();
question.song = "xxx";
question.text = "xxx";
question.a = "xxx";
question.b = "xxx";
question.c = "xxx";
question.answer = "xxx";
question.timer = 60;
questions_Songs.Add(question);
```

9. Congratulations our app is already finished. You can also run it now! and we can already start filling up more questions

For the whole Source Code you may download it here: http://dl.dropbox.com/u/12267912/BundledFun%20Source.zip

(c) Copyright Nokia 2013. All rights reserved.

Codes Explanations

These variables will keep track of the last media the player was on. Since Windows Phone only supports 1 MediaElement on 1 page we need to use only 1 MediaElement and as we change Pivot Pages we need to save the last media we were last on.

```
        private String lastTrailers = "";
        private String lastSongs = "";
```

These variables will be needed to determine if the certain category (e.g. Trailers) is already finished or no more questions left.

```
        private Boolean isFinished = false;
```

'These variables will be needed to determine if the certain category (e.g. Trailers) has already been started for the first time

```
        private Boolean hasStarted = false;
```

'These variables will be inceremented 1 everytime the player answers a correct question on a certain category

```
        private int corrected = 0;
```

'These variables will be inceremented depending on the time elapsed for each question everytime the player answers a (only) correct question on a certain category

```
        private int totalTimeElapsed = 0;

 'These variables will be used to track the latest seconds elapsed'
 <code csharp>
        private int seconds = 0;
```

'These variables will be initialized only once the application has been loaded, this will get the total questions for each category

```
        private int totalQuestions= 0;
```

'These variables will store all the questions in every category

```
        private List<Question> questions = new List<Question>();
```

'These variables will store the current question the player is on

```
        private Question currentQuestion = new Question();
```

'These variables are DispatcherTimer/Timer Objects to be used for our Timer to show how many seconds left on a certain question

```
        private System.Windows.Threading.DispatcherTimer timer;
```

This is the initialization part of the app, store all questions to their respective variables

```
        public PivotPage()
        {
            InitializeComponent();
            this.initialize_Images();
            this.initialize_Trailers();
            this.initialize_Songs();
            this.startImages();
        }
```

This is the initialization part of every questions, this is where the adding of questions to the List and initialization of objects like Timers and etc.

```
        private void initialize()
        {
            Question question = null;
            question = null;
            question = new Question();
            question.image = new BitmapImage(new Uri("Images/adobe.png", UriKind.Relative));
            question.text = "Is an American multinational computer software company founded in 1982 and headquartered in San Jose, C
            question.a = "Gimp";
            question.b = "Adobe Systems";
            question.c = "Microsoft";
            question.answer = "Adobe Systems";
            question.timer = 30;
            questions.Add(question);
            totalQuestions = questions.Count;
            txtCorrect.Text = "Correct: " + corrected + "/" + totalQuestions;
            timer = new System.Windows.Threading.DispatcherTimer();
            timer.Interval = new TimeSpan(0, 0, 0, 0, 1000);
            timer.Tick += new EventHandler(timer_Tick);
        }
```

This is the function to start a certain category'

```
        private void start()
        {
```


(c) Copyright Nokia 2013. All rights reserved.

```
        txtTimer.Text = "Time Left: " + currentQuestion.timer;
        timer.Start();
        this.nextQuestion();
        hasStarted= true;
    }
```

This is the function to stop a certain category'

```
    private void stop()
    {
        seconds = 0;
        timer.Stop();
        isFinished = true;
        btnA.IsEnabled = false;
        btnB.IsEnabled = false;
        btnC.IsEnabled = false;
        pivot.SelectedIndex = 1;
        if (this.isEverythingFinished())
        {
           MessageBox.show("Everything Finished");
        }
    }
```

This is function to shuffle/randomize the list of Questions

```
    private List<E> ShuffleList<E>(List<E> inputList)
    {
        List<E> randomList = new List<E>();
        Random r = new Random();
        int randomIndex = 0;
        while (inputList.Count > 0)
        {
            randomIndex = r.Next(0, inputList.Count); //Choose a random object in the list
            randomList.Add(inputList[randomIndex]); //add it to the new, random list
            inputList.RemoveAt(randomIndex); //remove to avoid duplicates
        }
        return randomList; //return the new random list
    }
```

This is function checks if all categories has been finished (or no more questions left for everthing)

```
    private Boolean isEverythingFinished()
    {
        if (isImagesFinished && isTrailersFinished && isSongsFinished)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
```

This event handling function for the Pivot is triggered everytime we swipe/change page

```
    private void pivot_SelectionChanged(object sender, SelectionChangedEventArgs e)
    {
        if (pivot.SelectedIndex == 0) // Images
        {
            if (!hasStarted_Images)
            {
                this.startImages();
            }
            else if(!isImagesFinished)
            {
                this.resumeImages();
                this.pauseTrailers();
                this.pauseSongs();
            }
        }
        else if (pivot.SelectedIndex == 1)// Trailers
        {
            if (!hasStarted_Trailers)
            {
               this.startTrailers();
            }
            else if (!isTrailersFinished)
            {
                this.pauseImages();
                this.resumeTrailers();
                this.pauseSongs();
            }
        }
        else if (pivot.SelectedIndex == 2)// Songs
        {
            if (!hasStarted_Songs)
            {
                this.startSongs();
            }
            else if (!isSongsFinished)
            {
                this.pauseImages();
                this.pauseTrailers();
                this.resumeSongs();
            }
        }
    }
```

This pauses the timer

```
    private void pause()
    {
        timer.Stop();
    }
```


(c) Copyright Nokia 2013. All rights reserved.

This resumes the timer

```
        private void resume()
        {
            timer.Start();
        }
```

This is the event for the buttons to check if they clicked the correct answer

```
        private void btn_Click(object sender, RoutedEventArgs e)
        {
            this.compare(btn.Content.ToString());
        }
```

This function is called when it should load the next question

```
        private void nextQuestion()
        {
            questions = ShuffleList<Question>(questions);
            currentQuestion= questions.ElementAt<Question>(0);
            txtQuestion.Text = currentQuestion.text;
            txtTimer.Text = "Time Left: " + currentQuestion.timer;
            Images.Source = currentQuestion.image; // for image
            media.Source = new Uri(file, UriKind.Relative); // for trailers and audios
            media.Play(); // for trailers and audios
            btnA.Content = currentQuestion.a;
            btnB.Content = currentQuestion.b;
            btnC.Content = currentQuestion.c;
        }
```

This function compares the clicked answer to the correct answer

```
        private void compare(String answer)
        {
            timer.Stop();
            if (answer.Equals(currentQuestion.answer))
            {
                corrected++;
                totalTimeElapsed += (60 - Convert.ToInt16(txtTimerTrailers.Text.Substring(10)));
                txtCorrect.Text = "Correct: " + corrected + "/" + totalQuestions;
                //show a correct message
            }
            else
            {
                //show a message
            }
        }
```

This is the event of the timer, called every tick of a second'

```
        void timer_Tick(object sender, EventArgs e)
        {
            seconds++;
            int timeLeft = currentQuestion.timer - seconds;
            if (timeLeft > 0)
            {
                txtTimer.Text = "Time Left: " + timeLeft;
            }
            else
            {
                seconds = 0;
                if(questions.Count > 0)
                {
                    questions.RemoveAt(0);
                    if (questions.Count > 0)
                    {
                        this.nextQuestion();
                    }
                    else
                    {
                        this.stop();
                    }
                }
                else{
                    this.stop);
                }
            }
        }
```

This is the function to restart the game, Basically what we're just doin here is resetting all the values to their default

```
 private void restart()
        {
            lastTrailers = "";
            lastSongs = "";
            isImagesFinished = false;
            isTrailersFinished = false;
            isSongsFinished = false;
            hasStarted_Images = false;
            hasStarted_Trailers = false;
            hasStarted_Songs = false;
            corrected_Images = 0;
            corrected_Trailers = 0;
            corrected_Songs = 0;
```


(cc) Copyright Nokia 2013. All rights reserved.

```
            totalTimeElapsed_Images = 0;
            totalTimeElapsed_Trailers = 0;
            totalTimeElapsed_Songs = 0;
            seconds_Images = 0;
            seconds_Trailers = 0;
            seconds_Songs = 0;
            totalQuestions_Images = 0;
            totalQuestions_Trailers = 0;
            totalQuestions_Songs = 0;
            txtCorrectImages.Text = "";
            txtCorrectSongs.Text = "";
            txtCorrectTrailers.Text = "";
            txtQuestionImages.Text = "";
            txtQuestionSongs.Text = "";
            txtQuestionTrailers.Text = "";
            txtTimerImages.Text = "";
            txtTimerSongs.Text = "";
            txtTimerTrailers.Text = "";
            btnAImages.IsEnabled = true;
            btnBImages.IsEnabled = true;
            btnCImages.IsEnabled = true;
            btnATrailers.IsEnabled = true;
            btnBTrailers.IsEnabled = true;
            btnCTrailers.IsEnabled = true;
            btnASongs.IsEnabled = true;
            btnBSongs.IsEnabled = true;
            btnCSongs.IsEnabled = true;
            media.Stop();
            timer_Images.Stop();
            timer_Trailers.Stop();
            timer_Songs.Stop();
            questions_Images = new List<Question>();
            currentQuestion_Images = new Question();
            questions_Trailers = new List<Question>();
            currentQuestion_Trailers = new Question();
            questions_Songs = new List<Question>();
            currentQuestion_Songs = new Question();
            this.initialize_Images();
            this.initialize_Trailers();
            this.initialize_Songs();
            pivot.SelectedIndex = 0;
            this.startImages();
        }
```


(c) Copyright Nokia 2013. All rights reserved.

(c) Copyright Nokia 2013. All rights reserved.