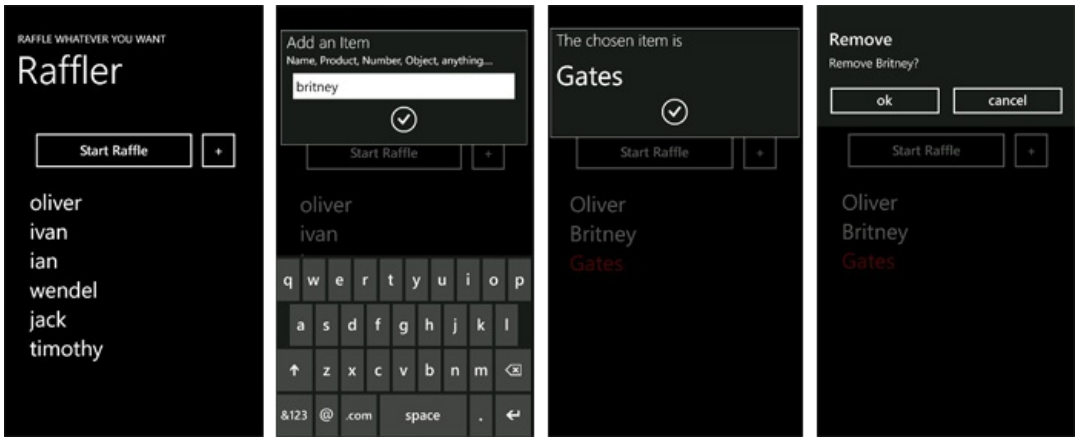
Create a random selection app for Windows Phone

This article shows how to create a UI for adding user-entered items to ListBox and then randomly selecting a "winner". This might be useful, for example, for choosing the winner of a raffle. The example also shows how to use the (third party) InputPrompt to get user input, how to remove items from the `ListBox`, and how to select a random number.

Introduction

The "Raffling" application is shown below. The user enters items by pressing the **+** button, and can remove items by selecting them from the list. When all items are entered the **Start Raffle** button can be pressed to randomly select one of the items.



The following sections show how each part of the application are constructed.

Implementation overview

The main screen of the application is shown below. In addition to the normal Silverlight application infrastructure like page and application title, we have two buttons for adding items and starting the random selection, a `ListBox` for displaying the items.

Items are added using the InputPrompt from the Coding4Fun Tools (which is easier to use than creating your own "popup" user input component.)

(C) Copyright Nokia 2014. All rights reserved.

The random item is selected using the C# `Random` object.

Create The Project

1. Create an empty Windows Phone Project.
   - Launch **Visual Studio -> click on File->New Project**.
   - Select **Windows Phone Application (Visual C# Template)**.
   - Add Name and Location of the project, then click **OK** to create the project.

Using Coding4Fun tools

In this application we are going to use a very cool third party component to get user inputs called: `InputPrompt`. You can download it here at : Coding4Fun.Phone.Toolkit and save and extract it on any location you want. There is more information on this component here: InputPrompt in depth

1. Now we need to add "Coding4Fun.Phone.Controls.dll assembly" as Reference to our Project for us to be able to use it.

   On the Solution Explorer Right Click on **References** > **Add Reference** > **Browse** > and locate **Coding4Fun.Phone.Controls.dll** inside the extracted folder
2. Then Add this Using Statement above `using Coding4Fun.Phone.Controls;`
3. And you are now ready to use the `InputPrompt` component

Building and Designing the UI

On your **MainPage.xaml** You may refer to the illustration above.

1. Draw the Button for triggering the Raffle event. Design and name it whatever you want (I named it: `btnRaffle`).
2. Draw also a Button for Adding Items to be raffled. Design and Name it whatever you want (I named it: `btnAddItem`).
3. (Last) Draw a ListBox for the added items to be stored. Design and Name it whatever you want (I named it: `listItems`).

Now that our UI ingredients are done, we can now start putting some action on them.

Adding Actions to our UIs

Next, we'll learn how to add items to the `ListBox` programatically.

On your **MainPage.xaml** Double Click on the **Add Item Button** to add an `EventHandler` and **MainPage.cs** will be opened it should look something like this below

```
//  btnAddItem Click Event Handler
private void btnAddItem_Click(object sender, RoutedEventArgs e)
{

}
```

Getting user input using the `InputPrompt` component

When the user clicks the **Add Item** button, we want our app to give them a place to input their desired items. This is where the `InputPrompt` component is useful.

Now add the following code snippet inside the function block (I'll explain every line of code in the comment lines)

```
// create an instance of the InputPrompt Object
InputPrompt input = new InputPrompt();

// add the completed event handler
input.Completed += new EventHandler<PopUpEventArgs<string,
PopUpResult>>(input_Completed);

// set the title of the InputPrompt
```


(C) Copyright Nokia 2014. All rights reserved.

```
input.Title = "Add an Item";

// set the message of the InputPrompt
input.Message = "Name, Product, Number, Object, anything....";

// define the input scope NameValue as Text (for general)
input.InputScope = new InputScope { Names = { new InputScopeName() { NameValue =
InputScopeNameValue.Text} } };

// show the InputPrompt
input.Show();
```

Adding items to the `ListBox`

When you write the above code, it will give you an error something like "Error : The name 'input_Completed' does not exist in the current context" This is because we don't have any `input_Completed` handler for our `InputPrompt` yet. So we'll create it right now.

Paste this anywhere (make sure it's inside your class's scope).

```
//  InputPrompt Completed / Add Item Event Handler
void input_Completed(object sender, PopUpEventArgs<string, PopUpResult> e)
{
    // check if the user inputted atleast 1 character
    if (e.Result.Length > 0)
        {
        // add the inputted item
        listItems.Items.Add(e.Result);
        }
    else // if not, tell the user
        {
        MessageBox.Show(Please enter an item with atleast 1 character.");
        }
}
```

Removing Items in the `ListBox`

Go to your **MainPage.xaml** and click your `ListBox`, and on the Properties Section, go to Events > and look for the "Tap" Event, then double click it to generate the Handler code. It should look something like this:

```
//  ListItems Tap Event Handler
private void listItems_Tap(object sender, GestureEventArgs e)
{

}
```

Paste the following code inside the block

```
//  get the MessageBoxResult: if they clicked "Ok" then execute, if they clicked
"Cancel" do nothing
if (MessageBox.Show("Remove " + listItems.SelectedItem + "?", "Remove",
MessageBoxButton.OKCancel) == MessageBoxResult.OK)
{
    // Remove the selected item
    listItems.Items.Remove(listItems.SelectedItem);
```


(C) Copyright Nokia 2014. All rights reserved.

```
}
```

Now if we run it, we are already able to remove items.

Getting the random item in the `ListBox`

This is where we'll learn how to get a random item in the list.

On your **MainPage.xaml** double click on your Raffle / Raffling Button. The code should look something like this:

```
//  btnRaffle Click Event Handler
private void btnRaffle_Click(object sender, RoutedEventArgs e)
{

}
```

- Now, we need to show to the user the randomly chosen item right? Okay, Fine, Copy the below code, paste it, and run

```
//   if there are atleast 2 items on the list
if (listItems.Items.Count > 1)
{
    //  create a Random object Instance
    Random random = new Random();

    //  generate a random number from 0 up to the size of the list ( we did this because
we don't want to get an index that is not on the list)
    int randomNumber = random.Next(0, listItems.Items.Count);

    //  set the selected item by index corresponding to the generated random number
    listItems.SelectedIndex = randomNumber;

    //  show the chosen/selected item (you can say"the winner is / or anything you
want")
    MessageBox.Show("The chosen item is " + listItems.SelectedItem.ToString());
    }
else // if there are no items / only 1 item in the list tell the user to add atleast two
items
    {
    MessageBox.Show("You have no items to raffle \n\n Tap the + to add an item);
}
```

Summary

Feel free to copy this application, tweak it, expand it, improve it, share it with the whole world.

Version Hint

Windows Phone: [[Category:Windows Phone]]
[[Category:Windows Phone 7.5]]
[[Category:Windows Phone 8]]

Nokia Asha: [[Category:Nokia Asha]]
[[Category:Nokia Asha Platform 1.0]]

Series 40: [[Category:Series 40]]
[[Category:Series 40 1st Edition]] [[Category:Series 40 2nd Edition]]
[[Category:Series 40 3rd Edition (initial release)]] [[Category:Series 40 3rd Edition FP1]] [[Category:Series 40 3rd Edition FP2]]
[[Category:Series 40 5th Edition (initial release)]] [[Category:Series 40 5th Edition FP1]]

(C) Copyright Nokia 2014. All rights reserved.

[[Category:Series 40 6th Edition (initial release)]] [[Category:Series 40 6th Edition FP1]] [[Category:Series 40 Developer Platform 1.0]] [[Category:Series 40 Developer Platform 1.1]] [[Category:Series 40 Developer Platform 2.0]]

Symbian: [[Category:Symbian]]
[[Category:S60 1st Edition]] [[Category:S60 2nd Edition (initial release)]] [[Category:S60 2nd Edition FP1]] [[Category:S60 2nd Edition FP2]] [[Category:S60 2nd Edition FP3]]
[[Category:S60 3rd Edition (initial release)]] [[Category:S60 3rd Edition FP1]] [[Category:S60 3rd Edition FP2]]
[[Category:S60 5th Edition]]
[[Category:Symbian^3]] [[Category:Symbian Anna]] [[Category:Nokia Belle]]

(C) Copyright Nokia 2014. All rights reserved.