Windows® Phone

Managing Application Page Layout

Session 4.3

Topics

- Landscape and Portrait orientation
- The OrientationChanged event
- Using Containers to manage layout
- The StackPanel element

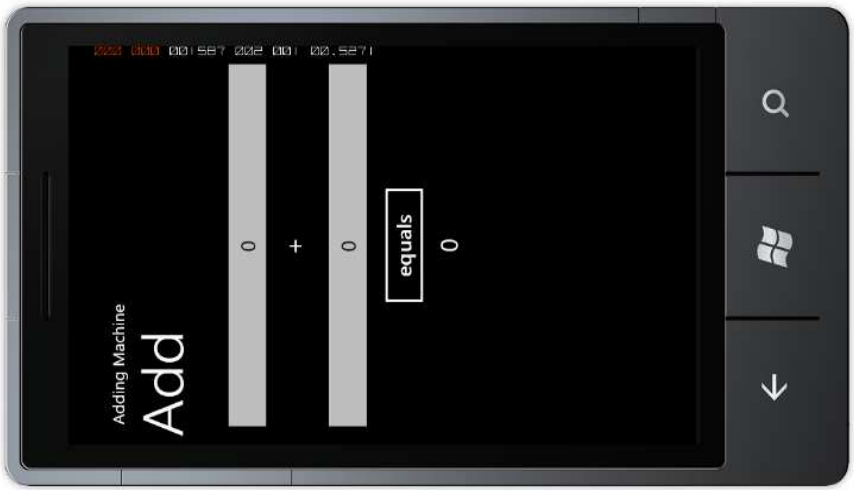
Windows Phone Orientation

- Unlike a desktop device, users will can use a Windows Phone in either orientation
  - Portrait orientation – stood up
  - Landscape orientation – laid on side
- Some applications work best in particular orientations
- We might want to show off, and make an application that works in either

Orientation Warning

- Handling multiple orientations is hard work
- It is at least twice as difficult as one orientation
  - You need to design the screen twice
  - You then have to add the orientation change management
- Many applications only work in one orientation
  - Including the Windows Phone Start Screen
- So consider this issue carefully

Landscape and Portrait Programs



- At the moment the adding machine handles a change of orientation quite badly
  - It does nothing
- The application has been configured to work only in "portrait" mode

Landscape and Portrait Selection

```
SupportedOrientations="Portrait" Orientation="Portrait"
```

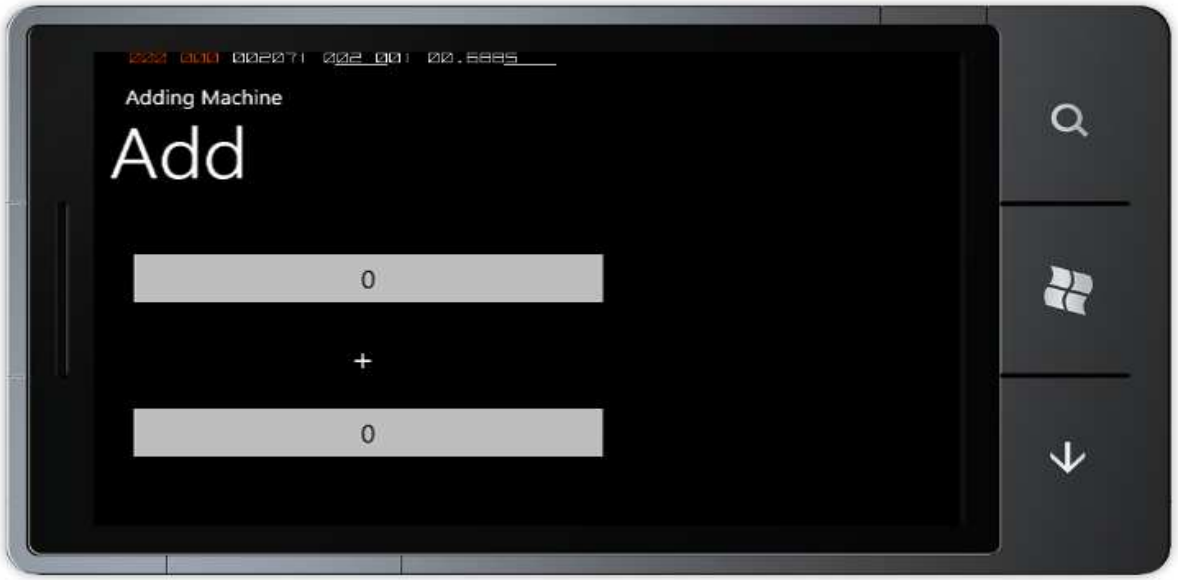
- The selection of orientation types for a Windows Phone page is done in the XAML file for that page
- This is the default setting
- The phone supports portrait orientation and is initially set to portrait

Allowing multiple orientations

```
SupportedOrientations="PortraitOrLandscape“
Orientation="Portrait"
```

- With this configuration the page can be used in both orientations
- The initial orientation of the page is landscape
- If the user tips the phone to the other orientation the program will try to draw the page in that orientation
- This might not end well

Rotating the Adding Machine



- The program works fine if the phone is rotated
- Unfortunately the equals button and the result are no longer visible

Silverlight Element positions

- The Sliverlight system uses coordinates to express the position of items
  - The display we designed places each item in the correct place for a Landscape display
- The Silverlight system will not complain if an application tries to draw something off the screen
- But this will not work very well when the display changes orientation

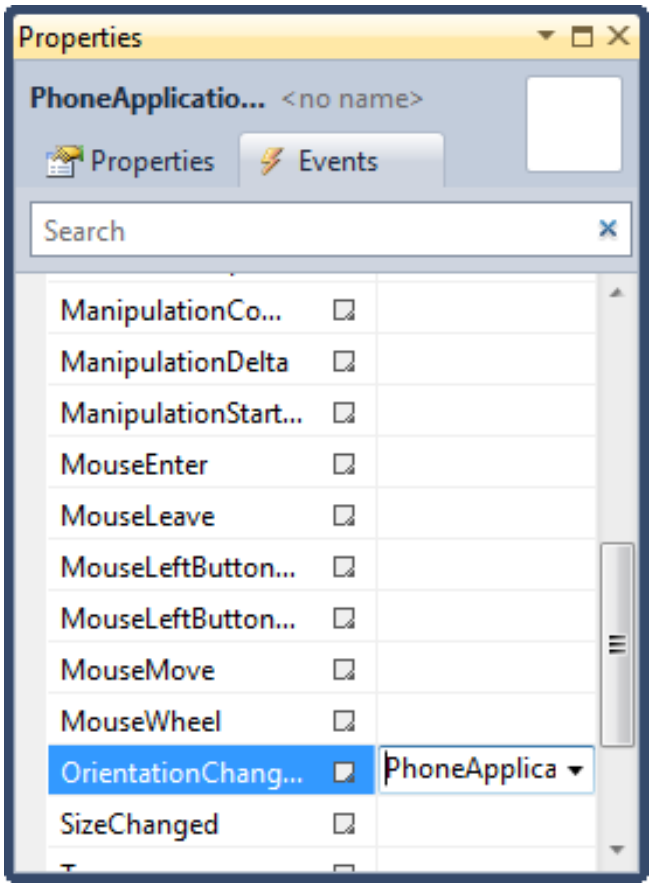
Allowing multiple orientations

```
<TextBox Height="72" HorizontalAlignment="Left"
Margin="8,19,0,0“ Name="firstNumberTextBox" Text="0"
VerticalAlignment="Top" Width="460"
TextAlignment="Center" />
<Button Content="equals" Height="72"
HorizontalAlignment="Left" Margin="158,275,0,0"
Name="equalsButton" VerticalAlignment="Top" Width="160"
Click="equalsButton_Click" />
```

- When we position elements using the designer Visual Studio adjusts Margin values to position them on the display
- These margin values need to be adjusted when the orientation changes

The OrientationChanged event

- An application can get notification of orientation changed events from a page
- When the phone is moved from one orientation to another the event will fire and the application can reposition display elements



Allowing multiple orientations

```
private void PhoneApplicationPage_OrientationChanged(
            object sender, OrientationChangedEventArgs e)
{
    if (e.Orientation == PageOrientation.PortraitUp) {
        setPortrait();
    }
    else {
        setLandscape();
    }
}
```

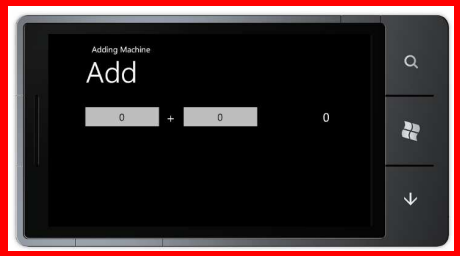
- This method runs when the orientation changes and calls the appropriate setting method

Setting Landscape Mode

```
private void setLandscape()
{
    firstNumberTextBox.Margin = new Thickness(8,19,0,0);
    firstNumberTextBox.Width = 207;
    secondNumberTextBox.Margin = new Thickness(252,19,0,0);
    secondNumberTextBox.Width = 207;
    plusTextBlock.Margin = new Thickness(221,35,0,0);
    resultTextBlock.Margin = new Thickness(538,35,0,0);
}
```

- This method configures the display for landscape mode
- The Thickness value contains four elements
  - X and Y position and border thickness (usually 0)

Demo



Demo 1: Changing Orientation

Using Containers

- Using Margins to position display elements will mean that the display will only work in one orientation
- One way to address this problem is to get the components to do the layout automatically
- Silverlight has a number of container components that can be used to lay out elements on the display
  - A container holds a number of elements

The StackPanel container

- A StackPanel contains a number of other text elements and stacks them up on the screen
- We don't have to explicitly position them, the StackPanel does this for us
- A StackPanel can arrange things across or down the screen
- We can nest StackPanel elements inside other StackPanels to create complex layouts

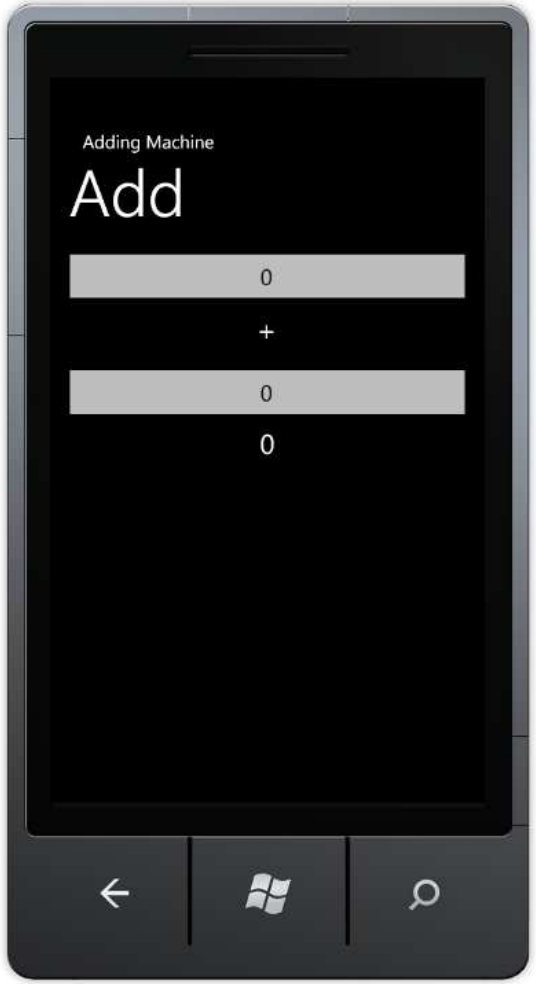
Creating a StackPanel

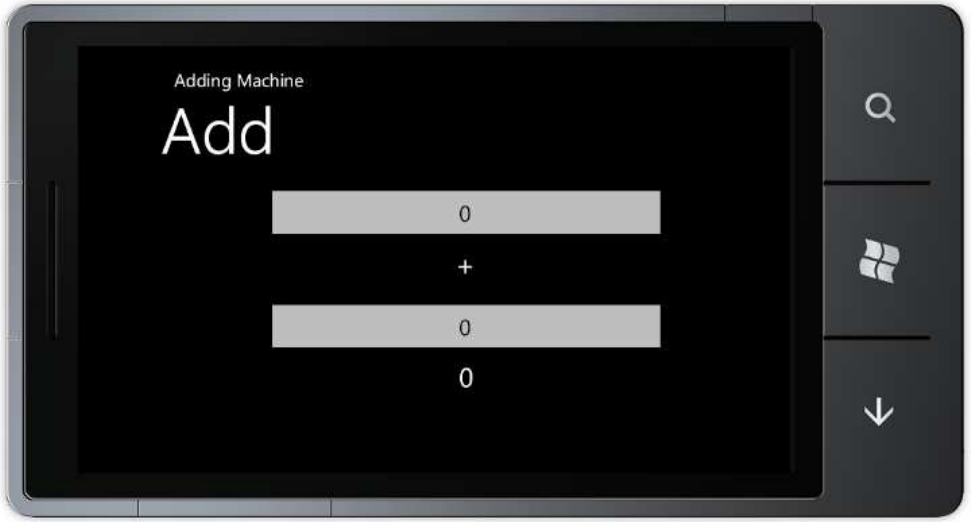
```
<StackPanel>
    <TextBox InputScope="Digits" Height="72"
             HorizontalAlignment="Center" ... />
    <TextBlock Height="56" HorizontalAlignment="Center"
       Name="plusTextBlock"
       Text="+" ... />
    <TextBox InputScope="Digits" Height="72“
       HorizontalAlignment="Center"
       Name="secondNumberTextBox" ... />
    <TextBlock Height="46" HorizontalAlignment="Center"
       Name="resultTextBlock" ... />
</StackPanel>
```

- The StackPanel will display the elements in the order they were added to the panel

Stack Panel in Action

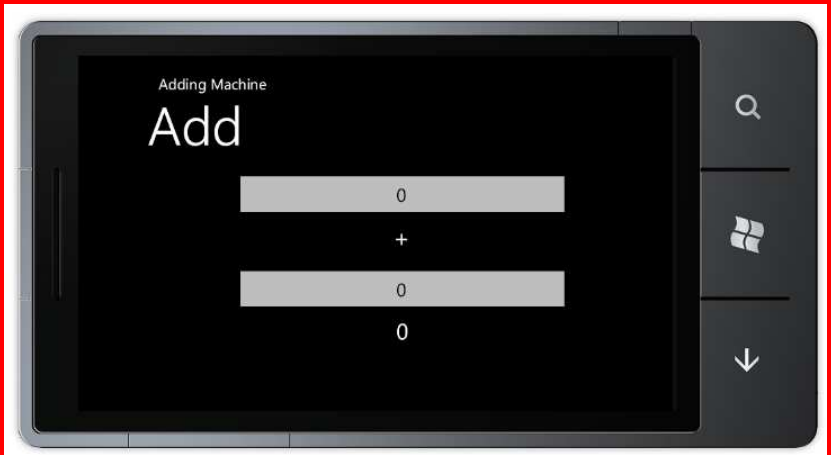
- The StackPanel performs the layout automatically
- The elements are displayed down the screen
- If the orientation of the device changes the StackPanel will lay out the controls again





- Because the elements are centred inside the StackPanel the landscape layout works quite well

Demo



Demo 2: Stack Panel

Review

- Windows Phone applications can run in landscape or portrait mode, or both
- You can set the allowed modes for each page
- Applications can bind to an event which fires when the phone orientation is changed
- Using container objects such as StackPanel can simplify the layout of pages and allow them to respond automatically to orientation changes