Surface Dial

Lab 1. Adding Surface Dial support for Ink Controls

The Surface Dial is designed to be used by the customers non-dominant hand. This leaves the dominant hand available to work on the details or the focal point of the task at hand.

A common scenario for Dial usage is where the customer is drawing with a stylus in their dominant hand and using the Surface Dial to control aspects of the ink input with their non-dominant hand.

In this lab you will learn how to easily add Surface Dial support to any application you build that supports Digital Ink input.

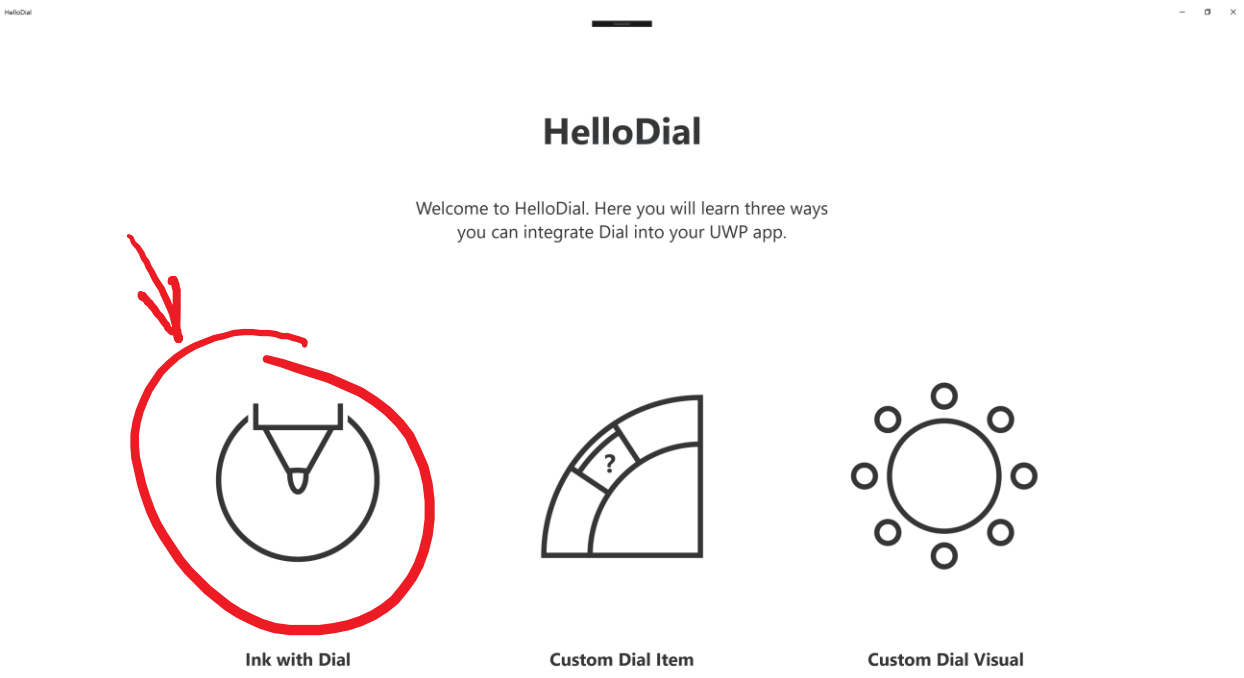
This lab assumes you have a working knowledge of C# and the Visual Studio IDE. To test the code in this lab you will need a device that has an active digitizer stylus (such as a Surface Pro or Surface Book) and a Surface Dial.

Ensure:

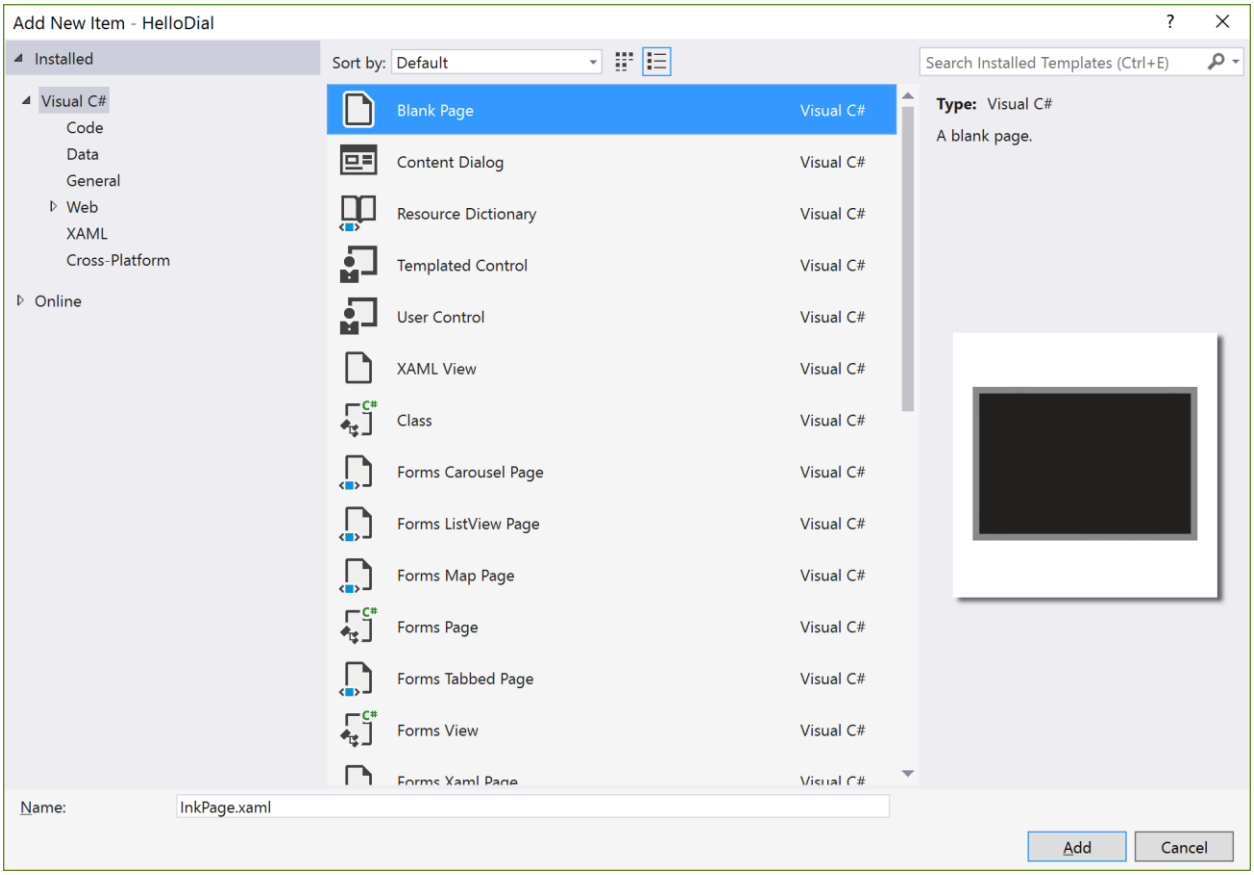
- The user account you are using is an administrator
- In settings>update & security>for developers: set to 'developer mode'
- In Visual Studio set the destination target to x86 or x64 (default may be ARM)
- If another user account on your machine has created this demo it will need to be uninstalled from that user account via the start menu

Note: This lab has been built on a Surface Studio running a resolution of 4500 x 3000. You might need to change the layout on the Pages to work on your display.

1. Open the existing application:
    a. Find the folder named **1. Start HelloDial**
    b. In this folder open the **HelloDial.sln** solution file in **Visual Studio 2015 or Visual Studio 2017**
2. Build and Run the application (F5) and you will see the start screen. In this hands on lab you will build the first scenario ‘Ink with Dial’



3. Tap on the ‘Ink with Dial’ option. It will produce a blank page. We are now going to create an app that demonstrates using the Dial and inking at the same time.
4. In **Solution Explorer** right click on the **Pages** folder and select **Add | New Item…** from the popup menu.
5. In the Add New Item dialog select Blank Page and name it InkPage.xaml



6. In the **InkPage.xaml** file name the Grid **MainGrid** and add a **Button** to go back to the previous page, and add a **TextBlock** to describe the contents of the Page.

```
<Grid x:Name="MainGrid"
    Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">
    <Button Background="Transparent"  Click="BackButton_Click"
        FontFamily="Segoe MDL2 Assets" Content="&#xE72B;" VerticalAlignment="Top"
        HorizontalAlignment="Left" Margin="5"/>
    <TextBlock FontFamily="Segoe UI" FontSize="66.5" FontWeight="Bold"
         Foreground="#353535"
         Margin="0,150,0,0" TextAlignment="Center" Text="Ink with Dial" />
</Grid>
```

7. Open the **InkPage.xaml.cs** code file. This is the **InkPage class**.

8. Immediately under the 'public sealed partial class InkPage: Page' add two member fields:

```
private InkCanvas inkCanvas = null;
private InkToolbar inkToolbar = null;
```

9. Add a handler for the **BackButton_Click**

```
private void BackButton_Click(object sender, RoutedEventArgs e)
{
```

```
        Frame.GoBack();
    }
```

10. Create a new method to add an **InkCanvas** to the Page

```
    private void AddInkCanvas()
    {
        inkCanvas = new InkCanvas();
        inkCanvas.IsHitTestVisible = false;
        MainGrid.Children.Add(inkCanvas);
    }
```

11. Add a method to add an **InkToolbar** to the Page

```
    private void AddInkToolbar()
    {
        inkToolbar = new InkToolbar();
        inkToolbar.TargetInkCanvas = inkCanvas;
        inkToolbar.HorizontalAlignment = HorizontalAlignment.Center;
        inkToolbar.VerticalAlignment = VerticalAlignment.Bottom;
        inkToolbar.Margin = new Thickness(0, 0, 0, 105);
        MainGrid.Children.Add(inkToolbar);
    }
```

12. Override the **OnNavigatedTo** method to call the **AddInkCanvas** and **AddInkToolbar** methods

```
    protected override void OnNavigatedTo(NavigationEventArgs e)
    {
        base.OnNavigatedTo(e);
        AddInkCanvas();
        AddInkToolbar();
    }
```

13. Override the **OnNavigatingFrom** method

```
    protected override void OnNavigatingFrom(NavigatingCancelEventArgs e)
    {
        base.OnNavigatingFrom(e);
        MainGrid.Children.Remove(inkToolbar);
        MainGrid.Children.Remove(inkCanvas);
        //release the reference
        inkToolbar = null;
        inkCanvas = null;
    }
```

*In this method remove the InkToolbar and InkCanvas from the Grid and release the reference to them. This removes the InkToolbar and InkCanvas from the view and therefore will modify the **RadialController** menu for this view. You will learn more about the **RadialController** in future labs.*

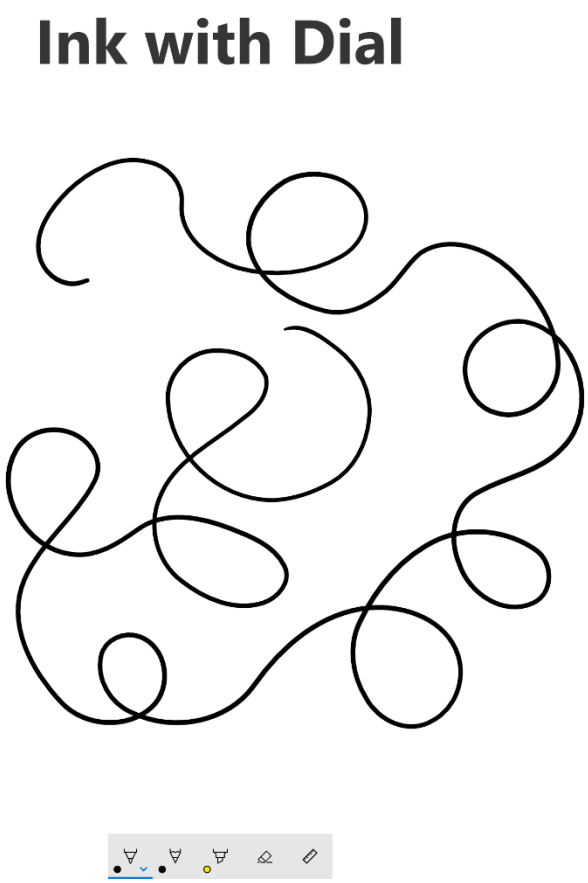
14. Open the **MainPage.xaml.cs** file and add code to navigate to the **InkPage** when the **InkBorder** is Tapped

```
using HelloDial.Pages;

. . .

private void InkBorder_Tapped(object sender, TappedRoutedEventArgs e)
{
    Frame.Navigate(typeof(InkPage));
}
```

15. Build and Run the application (F5). You can now tap on the *Ink With Dial* menu item and start drawing with a stylus and using the Surface Dial to manipulate the feature of the ink you are drawing.



When you select to go back to the MainPage, you will notice that the menu presented by the Surface Dial no longer contains the Ink controls. This is because the InkToolbar is no longer part of the view.

16. In this Lab you have learned that adding an InkCanvas and InkToolbar to a Page will automatically add Ink controls to the menu presented by Surface Dial. You have also learned that you need to remove the InkToolbar from the View (even if it is not visible) in order to remove the Ink menu items from the RadialController (the menu presented by the Surface Dial)