Layout

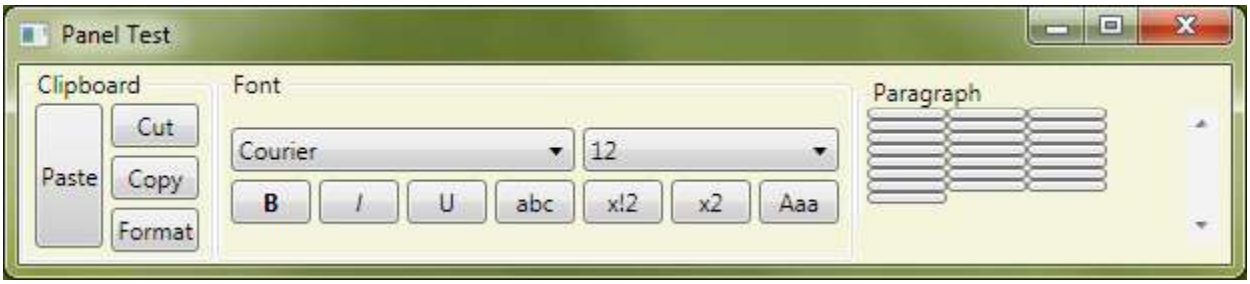
Estimated time for completion: 45 minutes

Goals:

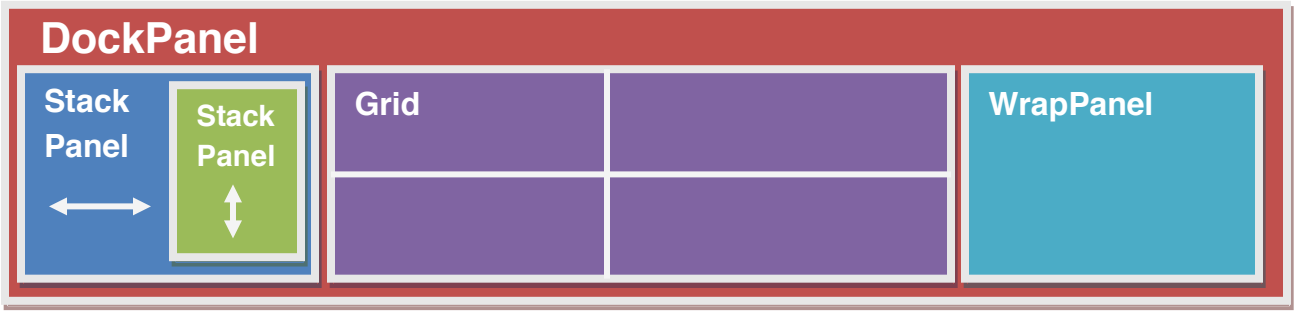
- Utilize the common layout panels to compose a complex user interface.
- Use the Control properties to position controls within the layout panels.

Overview:

In this lab, you'll use the different panel types to build a toolbar from scratch. The goal is to end up with a display which looks like the following picture:



We will not be creating an application here – simply some XAML markup to display the above picture. If you are adventurous, you can try to create the above screen on your own without detailed instructions. It uses Buttons and Combo boxes for all the UI elements. The above can be created a variety of ways and you are encouraged to try on your own, but for a starter, consider the following layout:



Notice how panels are contained within other panels to achieve the effect we want? This is very common in WPF. Play around with properties like **HorizontalAlignment** and **Margin** to get controls to line up and have proper spacing. The surrounding elements (Clipboard, Font and Paragraph) are `GroupBox` controls that are wrapping the panels.

Part 1 – Using StackPanel

In this part, you'll build the first chunk of the UI using a set of stack panels.

Steps:

1. Create a new WPF project solution with Visual Studio.
2. In the main window (`Window1.xaml`), replace the `Grid` with a `DockPanel`.
3. Inside this `DockPanel`, add a `StackPanel`.
    a. Set its `Orientation` to "Horizontal".
    b. Set its `VerticalAlignment` to "Top".
    c. Finally, set its `Height` to "100" and its `Background` to "Beige" - so that you can see it.

```
<DockPanel>
   <StackPanel Orientation="Horizontal" DockPanel.Dock="Top"
               Background="Beige" VerticalAlignment="Top" Height="100">
   </StackPanel>
</DockPanel>
```

4. Inside the stack panel, add three group boxes and set their `Header` to "Clipboard", "Font", and "Paragraph".

```
<DockPanel>
   <StackPanel Orientation="Horizontal" DockPanel.Dock="Top"
               Background="Beige" VerticalAlignment="Top" Height="100">
      <GroupBox Header="Clipboard" />
      <GroupBox Header="Font" />
      <GroupBox Header="Paragraph" />
  </StackPanel>
</DockPanel>
```

5. In the first `GroupBox`, add a `StackPanel` with 4 buttons with the text "Paste", "Cut", "Copy" and "Format".

```
<GroupBox Header="Clipboard">
   <StackPanel>
      <Button>Paste</Button>
      <Button>Cut</Button>
      <Button>Copy</Button>
      <Button>Format</Button>
   </StackPanel>
</GroupBox>
```

6. Notice that the buttons are not arranged the way we want. In particular, it does not fit in the height. What we need is to align the last three buttons next to the first button.

7. Introduce a new child `StackPanel` after the "Paste" button and set the Orientation to get the right layout. Move the last three buttons inside this new panel.

```
<GroupBox Header="Clipboard">
   <StackPanel Orientation="Horizontal">
      <Button>Paste</Button>
      <StackPanel>
         <Button>Cut</Button>
         <Button>Copy</Button>
         <Button>Format</Button>
      </StackPanel>
   </StackPanel>
</GroupBox>
```

8. Finally, stretch the "Paste" `Button` and set a `Margin` for each button so that it looks prettier – the example here will use a margin of "2" on all edges, but feel free to experiment. At this point, you can remove the `Height` on the parent `StackPanel` since the `StackPanel` will adjust to its content automatically.

```
<GroupBox Header="Clipboard">
   <StackPanel Orientation="Horizontal">
      <Button Margin="2,2,2,2" VerticalAlignment="Stretch">Paste</Button>
      <StackPanel>
         <Button Margin="2,2">Cut</Button>
         <Button Margin="2">Copy</Button>
         <Button Margin="2">Format</Button>
      </StackPanel>
   </StackPanel>
</GroupBox>
```

Solution

```
<DockPanel>
   <StackPanel Orientation="Horizontal" DockPanel.Dock="Top"
               Background="Beige" VerticalAlignment="Top">
      <GroupBox Header="Clipboard">
         <StackPanel Orientation="Horizontal">
           <Button Margin="2" VerticalAlignment="Stretch">Paste</Button>
           <StackPanel>
              <Button Margin="2">Cut</Button>
              <Button Margin="2">Copy</Button>
              <Button Margin="2">Format</Button>
           </StackPanel>
         </StackPanel>
      </GroupBox>
      <GroupBox Header="Font" />
      <GroupBox Header="Paragraph" />
```

```
   </StackPanel>
</DockPanel>
```

Part 2 – Using Grid

In this part, you'll implement the second GroupBox. Although you could implement it using StackPanels, you'll use a Grid instead.

Steps:

1. In the second `GroupBox`, add a `Grid`. Inside the grid, add a `ComboBox` with font items, a `ComboBox` for font sizes and 7 other buttons.
    a. Use your imagination for the content of these buttons.

```
<GroupBox Header="Font">
   <Grid>
      <ComboBox>
         <ComboBoxItem>Arial</ComboBoxItem>
         <ComboBoxItem>Courier</ComboBoxItem>
         <ComboBoxItem>Times New Roman</ComboBoxItem>
      </ComboBox>
      <ComboBox>
         <ComboBoxItem>10</ComboBoxItem>
         <ComboBoxItem>12</ComboBoxItem>
         <ComboBoxItem>14</ComboBoxItem>
      </ComboBox>
      <Button FontWeight="Bold" Grid.Column="0" Grid.Row="1">B</Button>
      <Button FontStyle="Italic" Grid.Column="1" Grid.Row="1">I</Button>
      <Button>U</Button>
      <Button>abc</Button>
      <Button>x!2</Button>
      <Button>x2</Button>
      <Button>Aaa</Button>
   </Grid>
</GroupBox>
```

2. At this point, all the items are on top of one another. In order to fix that we need to define column and row definitions on the `Grid` and assign controls individually into cells.
    a. Define 7 columns by adding the appropriate `ColumnDefinition` elements. These are defined inside the `Grid.ColumnDefinitions` element.

```
<Grid>
   <Grid.ColumnDefinitions>
      <ColumnDefinition />
      <ColumnDefinition />
      <ColumnDefinition />
```

```
      <ColumnDefinition />
      <ColumnDefinition />
      <ColumnDefinition />
      <ColumnDefinition />
   </Grid.ColumnDefinitions>
    ...
</Grid>
```

b. Define 2 rows by adding the appropriate `RowDefinition` elements. These are defined inside the `Grid.RowDefinitions` element.

```
<Grid>
   <Grid.ColumnDefinitions>
     ...
   </Grid.ColumnDefinitions>
   <Grid.RowDefinitions>
      <RowDefinition />
      <RowDefinition />
   </Grid.RowDefinitions>
    ...
</Grid>
```

c. For each control you added to the `Grid`, set the `Row` and `Column` property - these are attached properties defined by `Grid`. An example is:

```
<Button FontWeight="Bold" Grid.Column="0" Grid.Row="1">B</Button>
```

d. Finally, set a `ColumnSpan` of 4 and 3 for the two combo boxes.

```
<Grid>
   ...
   <ComboBox Grid.Column="4" Grid.ColumnSpan="4">
      <ComboBoxItem>Arial</ComboBoxItem>
      <ComboBoxItem>Courier</ComboBoxItem>
      <ComboBoxItem>Times New Roman</ComboBoxItem>
   </ComboBox>
   <ComboBox Grid.Column="4" Grid.ColumnSpan="3">
     ...
</Grid>
```

e. The full solution is presented below, here we have also broken out the underline and strike-out buttons with a full `TextBlock` using the `TextDecorations` property to implement underline and strike-out.

```
<GroupBox Header="Font">
   <Grid>
      <Grid.ColumnDefinitions>
         <ColumnDefinition />
```

```
         <ColumnDefinition />
         <ColumnDefinition />
         <ColumnDefinition />
         <ColumnDefinition />
         <ColumnDefinition />
         <ColumnDefinition />
      </Grid.ColumnDefinitions>
      <Grid.RowDefinitions>
         <RowDefinition />
         <RowDefinition />
      </Grid.RowDefinitions>
      <ComboBox Grid.ColumnSpan="4">
         <ComboBoxItem>Arial</ComboBoxItem>
         <ComboBoxItem>Courier</ComboBoxItem>
         <ComboBoxItem>Times New Roman</ComboBoxItem>
      </ComboBox>
      <ComboBox Grid.Column="4" Grid.ColumnSpan="3">
         <ComboBoxItem>10</ComboBoxItem>
         <ComboBoxItem>12</ComboBoxItem>
         <ComboBoxItem>14</ComboBoxItem>
      </ComboBox>
      <Button FontWeight="Bold" Grid.Column="0" Grid.Row="1">B</Button>
      <Button FontStyle="Italic" Grid.Column="1" Grid.Row="1">I</Button>
      <Button Grid.Column="2" Grid.Row="1">
          <TextBlock Text="U" TextDecorations="Underline" />
      </Button>
      <Button Grid.Column="3" Grid.Row="1">
          <TextBlock Text="abc" TextDecorations="Strikethrough" />
      </Button>
      <Button Grid.Column="4" Grid.Row="1">x!2</Button>
      <Button Grid.Column="5" Grid.Row="1">x2</Button>
      <Button Grid.Column="6" Grid.Row="1">Aaa</Button>
   </Grid>
</GroupBox>
```

3. By default, rows and columns are evenly distributed. Set the `Height` of the rows to "Auto" so that they adapt to their content.

```
<Grid>
   ...
   <Grid.RowDefinitions>
      <RowDefinition Height="Auto" />
      <RowDefinition Height="Auto" />
   </Grid.RowDefinitions>
    ...
</Grid>
```

4. Finally, center the grid vertically and set the `Margin` to "2" on all Controls.

Solution

```
<DockPanel>
   <StackPanel Orientation="Horizontal" DockPanel.Dock="Top"
               Background="Beige" VerticalAlignment="Top">
      <GroupBox Header="Clipboard">
         <StackPanel Orientation="Horizontal">
           <Button Margin="2" VerticalAlignment="Stretch">Paste</Button>
           <StackPanel>
              <Button Margin="2">Cut</Button>
              <Button Margin="2">Copy</Button>
              <Button Margin="2">Format</Button>
           </StackPanel>
         </StackPanel>
      </GroupBox>
      <GroupBox Header="Font">
         <Grid VerticalAlignment="Center">
            <Grid.ColumnDefinitions>
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
            </Grid.ColumnDefinitions>
            <Grid.RowDefinitions>
               <RowDefinition Height="Auto" />
               <RowDefinition Height="Auto" />
            </Grid.RowDefinitions>
            <ComboBox Margin="2" Grid.ColumnSpan="4">
               <ComboBoxItem>Arial</ComboBoxItem>
               <ComboBoxItem>Courier</ComboBoxItem>
               <ComboBoxItem>Times New Roman</ComboBoxItem>
            </ComboBox>
            <ComboBox Margin="2" Grid.Column="4" Grid.ColumnSpan="3">
               <ComboBoxItem>10</ComboBoxItem>
               <ComboBoxItem>12</ComboBoxItem>
               <ComboBoxItem>14</ComboBoxItem>
            </ComboBox>
            <Button Margin="2" FontWeight="Bold" Width="40"
                    Grid.Column="0" Grid.Row="1">B</Button>
            <Button Margin="2" Width="40" FontStyle="Italic"
                    Grid.Column="1" Grid.Row="1">I</Button>
            <Button Margin="2" Width="40" Grid.Column="2" Grid.Row="1">
               <TextBlock Text="U" TextDecorations="Underline" />
            </Button>
            <Button Margin="2" Width="40" Grid.Column="3" Grid.Row="1">
```

```
                <TextBlock Text="abc" TextDecorations="Strikethrough" />
            </Button>
            <Button Margin="2" Width="40" Grid.Column="4"
                    Grid.Row="1">x!2</Button>
            <Button Margin="2" Width="40" Grid.Column="5"
                    Grid.Row="1">x2</Button>
            <Button Margin="2" Width="40" Grid.Column="6"
                    Grid.Row="1">Aaa</Button>
         </Grid>
      </GroupBox>
      <GroupBox Header="Paragraph" />
   </StackPanel>
</DockPanel>
```

Part 3 – Using WrapPanel

In this last part, you'll implement the final GroupBox using a WrapPanel.

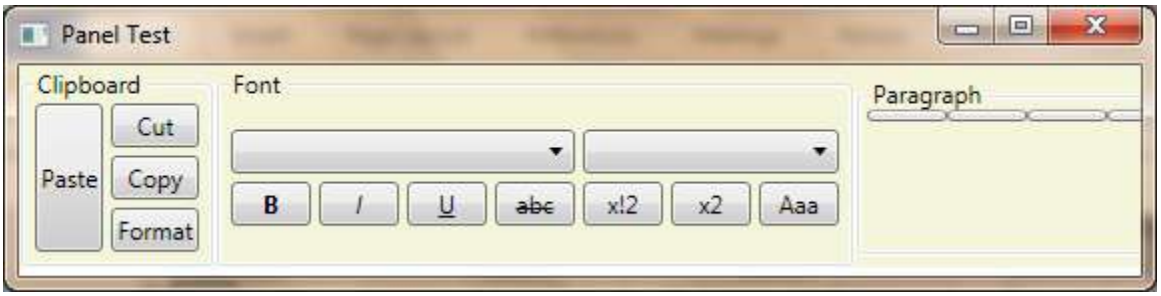
Steps:

5. In the last `GroupBox`, add a `WrapPanel` and add 20 buttons in it.
    a. Set the `Width` of each button to "40".

```
<GroupBox Header="Paragraph">
   <WrapPanel>
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
      <Button Width="40" />
   </WrapPanel>
```

```
</GroupBox>
```

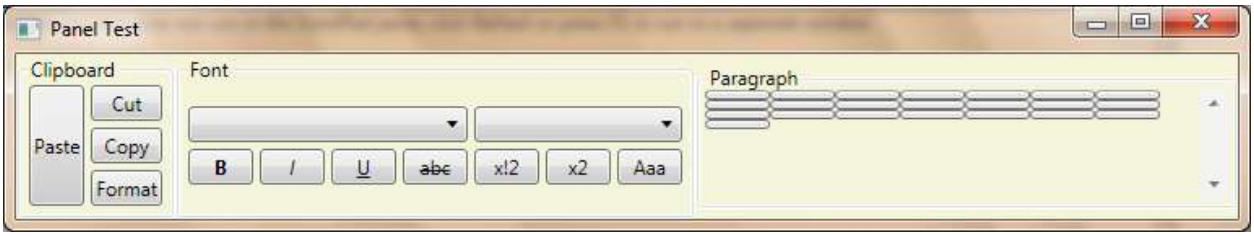
6. Compile and play with resizing the application window.
    a. Notice what happens when you resize to a small width – the `WrapPanel` cannot do its job because the containing `StackPanel` expands beyond the containing window.



7. Replace the outermost `StackPanel` with a `DockPanel`. You will need to remove the `Orientation` property as it does not exist on the `DockPanel`.
    a. The default docking is to the Left, but you can set the `DockPanel.Dock` property onto each of the `GroupBoxes` if you want to make it explicit.
    b. Due to the default behavior of `DockPanel`, it acts like a `StackPanel` configured for `Horizontal` orientation when you don't specify otherwise.
8. Verify that the buttons now flow from one line to another as you resize the window.
9. Finally, place the `WrapPanel` inside a `ScrollViewer` so that you can scroll it when necessary.

Solution

The final application looks like:




```
<Window Title="Panel Test"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml">

   <DockPanel Background="Beige"
              VerticalAlignment="Top">
      <GroupBox Header="Clipboard">
```

```
         <StackPanel Orientation="Horizontal">
           <Button Margin="2" VerticalAlignment="Stretch">Paste</Button>
           <StackPanel>
              <Button Margin="2">Cut</Button>
              <Button Margin="2">Copy</Button>
              <Button Margin="2">Format</Button>
           </StackPanel>
         </StackPanel>
      </GroupBox>
      <GroupBox Header="Font">
         <Grid VerticalAlignment="Center">
            <Grid.ColumnDefinitions>
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
               <ColumnDefinition />
            </Grid.ColumnDefinitions>
            <Grid.RowDefinitions>
               <RowDefinition Height="Auto" />
               <RowDefinition Height="Auto" />
            </Grid.RowDefinitions>
            <ComboBox Margin="2" Grid.ColumnSpan="4">
               <ComboBoxItem>Arial</ComboBoxItem>
               <ComboBoxItem>Courier</ComboBoxItem>
               <ComboBoxItem>Times New Roman</ComboBoxItem>
            </ComboBox>
            <ComboBox Margin="2" Grid.Column="4" Grid.ColumnSpan="3">
               <ComboBoxItem>10</ComboBoxItem>
               <ComboBoxItem>12</ComboBoxItem>
               <ComboBoxItem>14</ComboBoxItem>
            </ComboBox>
            <Button Margin="2" FontWeight="Bold" Width="40"
                    Grid.Column="0" Grid.Row="1">B</Button>
            <Button Margin="2" Width="40" FontStyle="Italic"
                    Grid.Column="1" Grid.Row="1">I</Button>
            <Button Margin="2" Width="40" Grid.Column="2" Grid.Row="1">
               <TextBlock Text="U" TextDecorations="Underline" />
            </Button>
            <Button Margin="2" Width="40" Grid.Column="3" Grid.Row="1">
               <TextBlock Text="abc" TextDecorations="Strikethrough" />
            </Button>
            <Button Margin="2" Width="40"
                    Grid.Column="4" Grid.Row="1">x!2</Button>
            <Button Margin="2" Width="40"
                    Grid.Column="5" Grid.Row="1">x2</Button>
            <Button Margin="2" Width="40"
```



```
                    Grid.Column="6" Grid.Row="1">Aaa</Button>
         </Grid>
      </GroupBox>
      <GroupBox MaxHeight="90" Header="Paragraph">
         <ScrollViewer>
         <WrapPanel>
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
            <Button Width="40" />
         </WrapPanel>
         </ScrollViewer>
      </GroupBox>
   </DockPanel>
</Window>
```