User interface design

in C#, with Windows Presentation Foundation (WPF)

# Lab 1 – Using and arranging built-in controls

## Functional Goal

Develop the initial version of a user interface for the My TV Companion application.

## Learning Goals

* Understand the Visual Studio designer
* Understand how Design and XAML are connected
* Understand the Grid control
* Understand properties

## Prerequisites

You’ll need to install Visual Studio (2012 Ultimate was used to create this lab) from the MSDN/DreamSpark service on ANGEL’s RosePortal.

## Submission Instructions

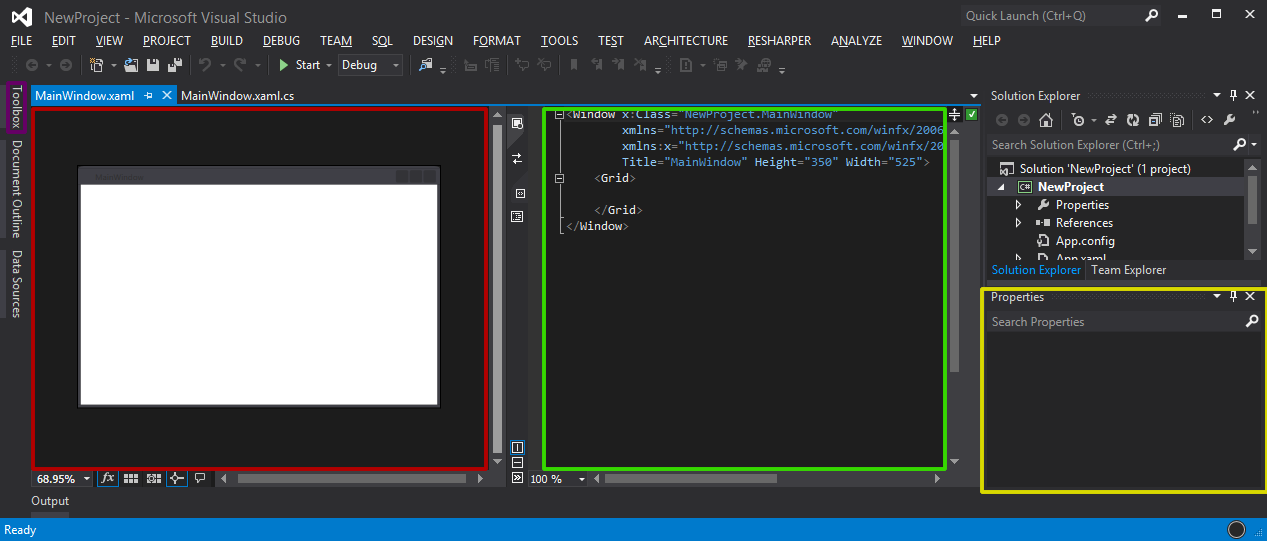
Submit answers to the **3** questions in this lab as a .pdf to the appropriate Moodle submission form.

## Get started: Creating a New Project

1. Open Visual Studio and click “New Project…”
2. In the tree on the left, expand to this path: “Installed/ Templates/ Visual C#/ Windows.
3. Select WPF Application in the middle pane.
4. Give your application the name MyTVCompanion and choose a location for your source code.
   1. Ensure that “Create directory for solution” is checked.
   2. Click OK.

## Meet your new best friend: the Visual Studio designer

When your new project is created, you should see something like the below image.



To use the designer successfully, you’ll need to use each of the following components to work most efficiently:

Toolbox: This contains all the controls currently available, given your project setup and file type. Come here to click-and-drag a new component or simply to see what’s available in WPF.

Design window: This is a semi-live version of your E**x**tensible **A**pplication **M**arkup **L**anguage (XAML). XAML is an XML-based language that describes the visual look of your applications. You can modify your layout here and watch what happens in the XAML, or simply click controls you want to find in the XAML. Later on, you’ll see the true power of the design pane when we begin data binding.

Code window: Pretty self-explanatory. Make sure you use IntelliSense (start typing or press Ctrl+Space) as much as possible – XAML should practically write itself.

Properties pane: The properties pane enumerates every option you can apply to a control. If you’re looking for functionality, or want to have Visual Studio write your binding code, this is the place to come.

Question 1: Which component of the designer would you use if you’ve added some controls to your Window, but you can’t figure out how to make them stretch all the way across? (2 points)

## Your first steps: the main My TV Companion window

Now that you’ve met the designer, let’s make a UI using several of its components. If you’re feeling ambitious, try to create this screen yourself, using the mockup for reference, before following the walkthrough below. Don’t worry about showing any actual shows in the left pane – just place a control that can show lists.

### Main window: Mockup

### Main window: Click-and-build

To get started, you’re going to build a UI simply by clicking and dragging – no code required.

1. Open the Toolbox pane on the left and drag a Button onto the Window. Position it bottom center.
2. Add a new row directly above the button by hovering over the left measurement line and clicking when the orange line appears where you want it.
3. Now add a new column in the middle of the Window by hovering over the top measurement line and clicking when the orange line appears where you want it.
4. Add a new StackPanel to the new left column by dragging it from the Toolbox.
   1. Position your StackPanel in the top-left of the new column. When you’ve dropped it, grab the bottom diagonal resize handle and expand the StackPanel to fit the column.
5. Add a new Label to the StackPanel by dragging it from the Toolbox.
6. Add a new ListView to the StackPanel by dragging it from the Toolbox.
7. Add a new Calendar to the new right column by dragging it from the Toolbox.

Now, run your application using the “Start” button. Resize the application when it appears and observe what happens.

### Main window: In code

Now, click and drag is nice, and it’s a great way to get started, but it’s rather imprecise. It also tends to favor absolute layouts, as you saw when you ran it. So, let’s build the same interface entirely by coding and specify the layouts manually. You’ll want to focus on the XAML pane for this part.

1. To get started, delete all the code within the <Grid> tag and change the <Window Height and Width to 350 and 525, respectively.
2. N.B. Don’t copy and paste the code below; start typing and let IntelliSense help you type the code yourself. Watch the designer as you add properties to see what happens.
3. Define your rows and columns.

<Grid.RowDefinitions>

<RowDefinition Height="\*" />

<RowDefinition Height="Auto" />

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*" />

<ColumnDefinition Width="\*" />

</Grid.ColumnDefinitions>

1. Add a StackPanel, Label, and ListView. Don’t worry about the Content tag – we’ll cover binding in a subsequent lab.

<StackPanel Grid.Row="0" Grid.Column="0" HorizontalAlignment="Stretch" VerticalAlignment="Stretch">

<Label HorizontalAlignment="Center" Content="{Binding Source={x:Static System:DateTime.Today}, StringFormat={}{0:MMMM d, yyyy}}" />

<ListView />

</StackPanel>

1. Add a Calendar.

<Calendar Grid.Row="0" Grid.Column="1" HorizontalAlignment="Center" VerticalAlignment="Center" />

1. Add a Button.

<Button Grid.Row="1" Grid.ColumnSpan="2">Settings</Button>

Now, run your application using the “Start” button. Resize the application when it appears – notice how your application now dynamically adjusts to the window size.

Question 2: What’s the difference between Height="Auto" and Height="\*" in RowDefinitions? Are either (or both) preferable to absolute layouts? Can you mix absolute and calculated measurements? (6 points)

## Your first steps: the My TV Companion Settings window

You’ve now seen how to arrange controls in a WPF application – it’s time to try for yourself. Create a new window (instructions are below) and create the Settings window for your application from the mockup below. Don’t worry if your implementation doesn’t look exactly like the mockup – you don’t actually need the sample shows and search results. The grader will be looking for your use of concepts from earlier in the lab (e.g. Row and ColumnDefinitions and StackPanels), not for pixel-perfect accuracy.

### Settings window: Mockup

### Settings window: Getting started

1. To create a new window, right-click on the MyTVCompanion project and select Add > Window. Call your new Window SettingsWindow.xaml.
2. To get to your new Window, you’ll need an event on the Button we made earlier. Locate it, and in the Properties panel, click the Events button ().
3. Double-click the Click event to create a new event handler. In that handler, create and show the Settings window.

private void Button\_Click\_1(object sender, RoutedEventArgs e)

{

new SettingsWindow().Show();

}

### Settings window: Your turn

Now, try to create the Settings window from the mockup. Some helpful hints:

* You can have a Grid within a Grid, a StackPanel within a StackPanel, a Grid within a StackPanel, etc. Pick the container that best matches what you’re trying to lay out and then position it appropriately in the parent container.
* If you want to see some data in a ListView, you can fake that data like this:

<ListView>

<ListView.Items>

<ListViewItem Content="Search result" />

</ListView.Items>

</ListView>

Question 3: Submit the XAML code (and a screenshot (if you’re not sure how to take a screenshot, type “Snipping Tool” in Start Search) of your Settings Window. If you are using an application that doesn’t retain Visual Studio’s text formatting on copy/paste, please take a screenshot of your code so that it remains properly formatted and colored. (12 points; 6 for all required controls, 6 for reasonable, resize-friendly layout)

## Congratulations

You’re done. Don’t forget: submit answers to the **3** questions in this lab as a .docx or .pdf to the appropriate Moodle submission form.