As we agreed on 7/31/18, this is a lab that I could use a replacement for the real lab13 (the wifi lab for the TI)

**How to setup windows 10 IoT your Raspberry Pi 2 or 3**

Side Note: These are the recommended Sd cards by Microsoft:

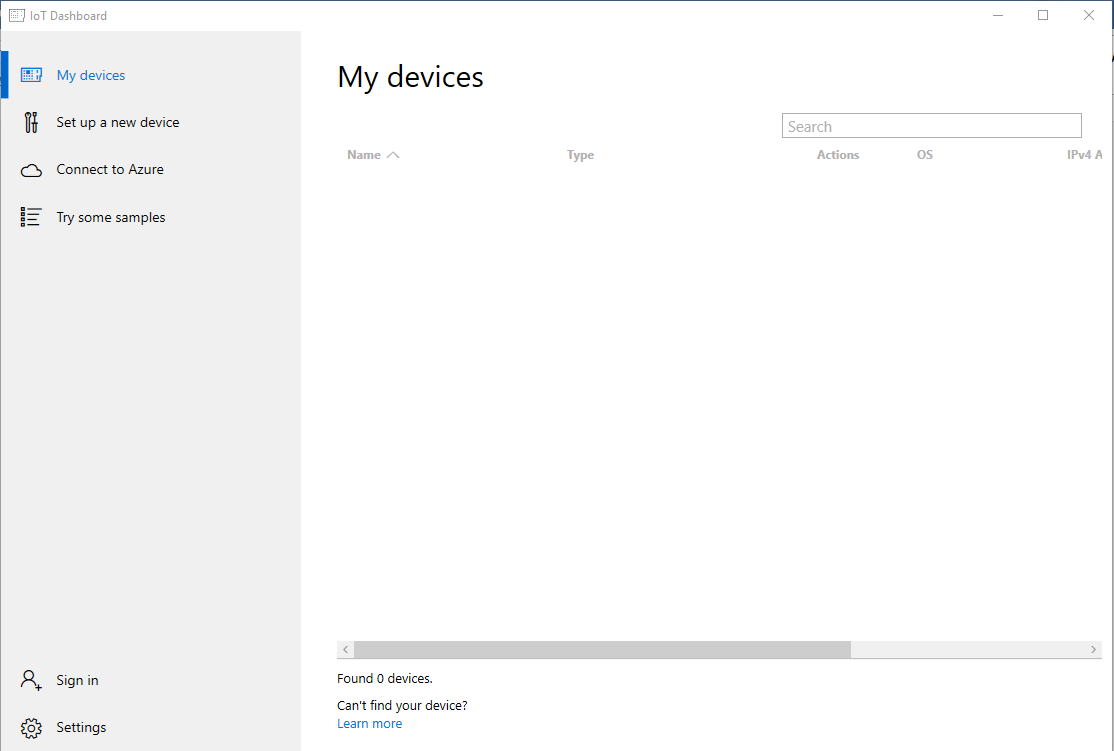
<https://smile.amazon.com/gp/product/B073K14CVB/ref=oh_aui_detailpage_o01_s00?ie=UTF8&psc=1>

**Part 1 – Setting up your Pi**

Go to this URL:

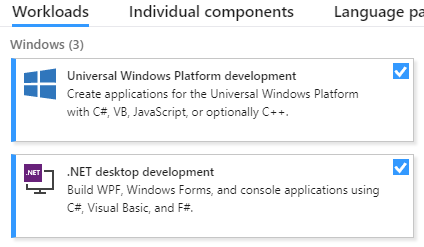
<https://docs.microsoft.com/en-us/windows/iot-core/tutorials/quickstarter/devicesetup>

Watch the video on how to setup your RPi with Windows 10.

1. Insert your SD card into your Computer
2. Download the “Windows 10 IoT Core Dashboard” from the URL above. Or, a direct link: <https://developer.microsoft.com/en-us/windows/iot/Downloads>
3. Run the application to get to the following screen:
4. Click Set up a new Device option
5. Set:
   1. Device Type = Broadcomm (Raspberry Pi 2 & 3)
   2. OS Build = Windows 10 IoT Core
   3. Drive = (your sd card, if using the ones above it should show up as 14GB. Make sure you have the correct SD card selected)
   4. Leave the Device name as minwinpc
   5. password can be whatever you want, “password” is acceptable
   6. Optionally, you can select which network for the Pi to connect to on startup with the “Wi-Fi Network Connection”)
6. Accept the software license terms, and click “Download and install”. Now wait until the Sd card is ready.
7. When done, insert into your SD card into the Pi and verify that it powers up. If it worked, you should see the standard windows boot icon with the spinning wheel.
   1. If you get a “Windows failed to start” message, it is likely that you are trying to use an SD card that is not recommended.
8. When booted, follow the steps in the welcome screen.
9. When done, you are greeted with a start screen that says your device type, name, and network information.
10. Go to settings:
    1. Under basic, make sure that the device has the correct time zone
    2. Under wifi (if you are using it), make sure that it is on and you are connected to the same network as your laptop.
11. Reboot the device and verify that the settings stuck.
12. The Device is now setup. Show this to your instructor.

**Part 2 – Setting Up Visual studio**

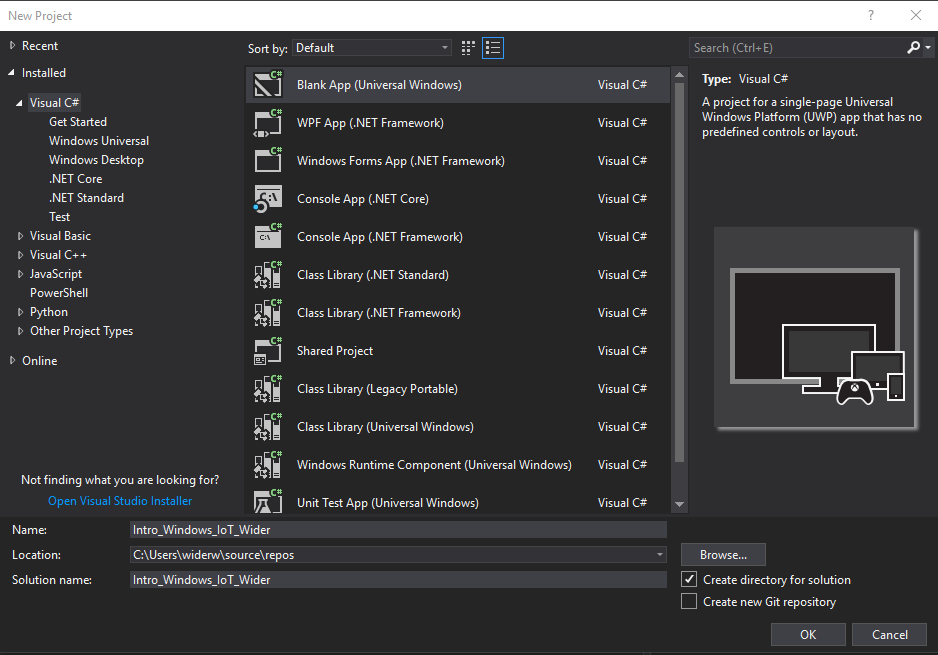
1. Go to this URL to download Visual Studio: <https://visualstudio.microsoft.com/downloads/>
2. Upon component selection:
   1. Make sure you are looking from the “Workloads” tab.
   2. Select “Universal Windows Platform development”, for the components you will need to develop on the Pi.
   3. Optionally, you can also select “.NET desktop development” to develop C# .NET applications via Winfoms and WPF for the desktop.



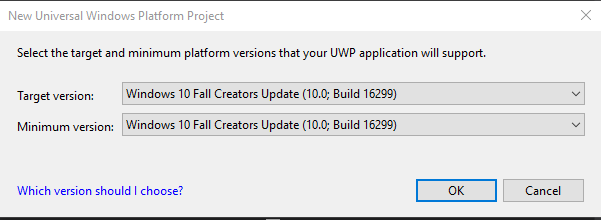
1. Let the program install from this point.
2. When done, it is recommended to reboot your PC

**Part 3 – Preparing your first Windows IoT App for deployment**

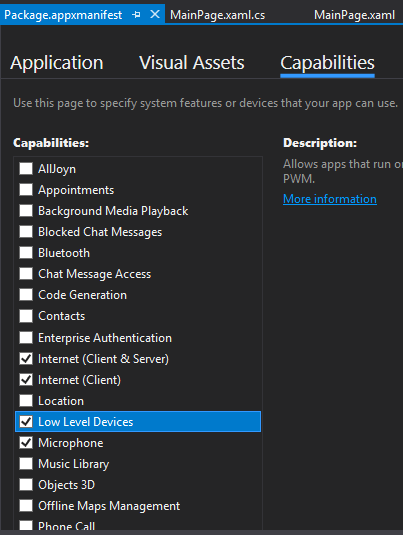
1. When opening visual studio, you will need to login to your Microsoft account. You may be able to continue not logged in, or login via your Microsoft account from Wentworth
2. Click File->New->Project
   1. Select on the left view tree-> Visual C#.
   2. In the middle column, select “Black App (Universal Windows)”
   3. The name should be “Intro\_Windows\_IoT\_Wider, but replace with your last name”



1. When it asks the target and minimum version, it is recommended to use the latest version that you can, in the case of this example, the current windows is the 2017 Fall Creators Update



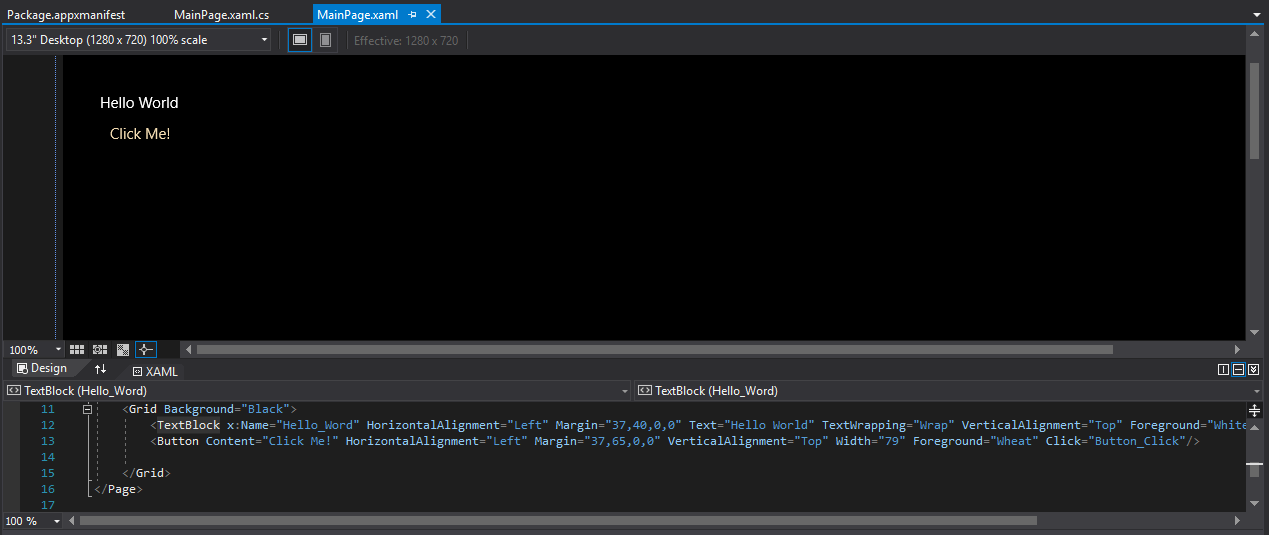
1. On the right, “Solution Explorer”, open the Package.appxmanifest file to declare the required IoT capabilities of your application.
   1. Go to the capabilities tab
   2. Select “low level devices” (you may select the others in the picture if you wish, they are not required)



1. Open the MainPage.xaml file. This file servs tow purposes:
   1. The UI designer, using the WPF/XML style engine
   2. The Code-Behind, stored in the \*.cs file

https://puu.sh/B6kMz/0ebd8988c6.png

* 1. Your designer should now look similar to this:



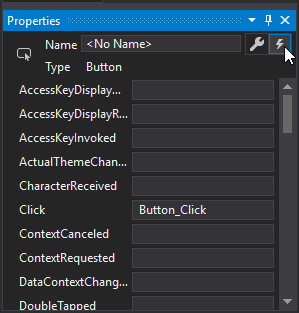
(It shows the UI up top, with the XAML code below.

* 1. Notice that when you drag components from the toolbox (on the left) like a label, it updates the XAML code.
  2. Copy and Paste the following code inside the “Grid” node:

<TextBlock x:Name="Hello\_Word" HorizontalAlignment="Left" Margin="37,40,0,0" Text="Hello World" TextWrapping="Wrap" VerticalAlignment="Top"/>

<Button Content="Click Me!" HorizontalAlignment="Left" Margin="37,65,0,0" VerticalAlignment="Top" Width="79"/>

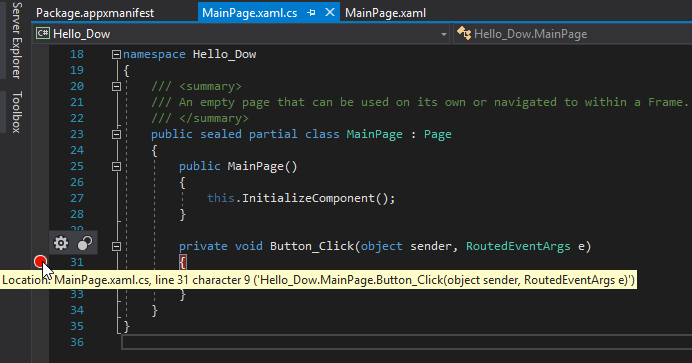
* 1. You will now see a similar result to the picture above, but the colors will be different. For this, it does not matter.
  2. The Node element is the type of UI component that it is (like TextBlock). The properties of the UI element are saved as XAML attributes. Any “child” UI components are saved as nodes within that node. Like the parent node “Grid” for example.
  3. Click on the button, and go to the event subscription manager area (the lightning bolt) and double click on the click event box. You will be brought to a new event method created in the\*.cs file.



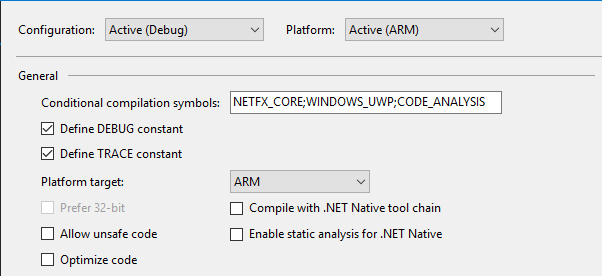
* 1. Add the following code to the method

Hello\_Word.Text = "Hello Dow!";

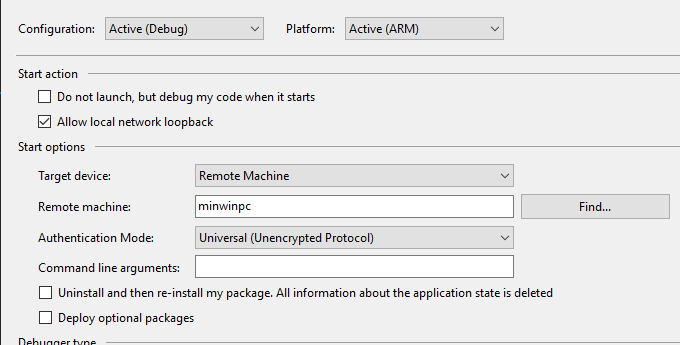
* 1. The above code will change the text of the first text block on a button click.
  2. Place a debug breakpoint right before the statement (at the bracket), so you will see the step by step GUI debugger.



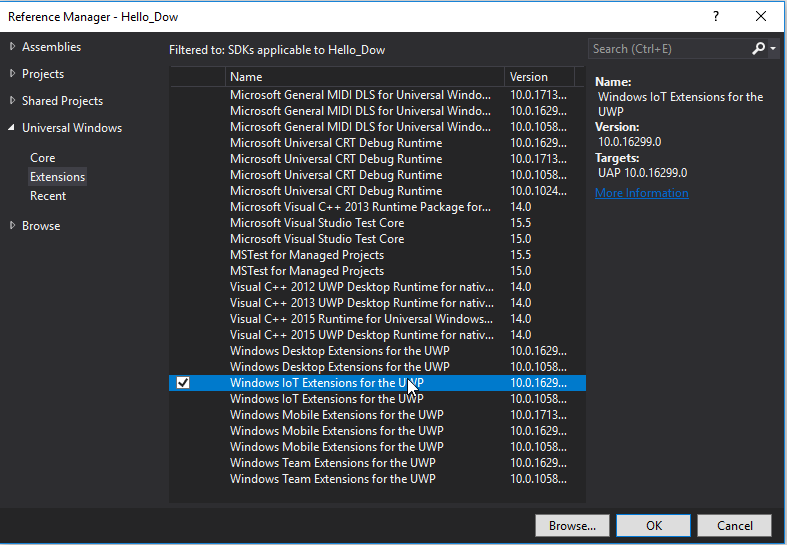
1. Configure the project Properties to run on the Rpi:
   1. Click project->Project properties menu option, and go to Build tab
      1. Make sure the Configuration is Debug
      2. Make sure the Platform is ARM
      3. Make sure the Platform Target is ARM



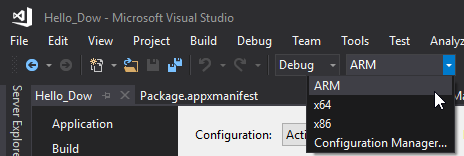
* 1. Go to Debug tab
     1. Make sure Configuration is Debug and Platform is ARM
     2. Under start options, set Target device as Remote Machine option
     3. Under remote Machine, put “minwinpc”
     4. Set authentication mode as “Universal (Unencrypted Protocol)”



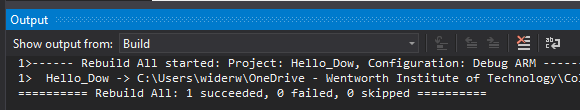
1. Add the Windows IoT extensions to the project
   1. Right-Click the Refrneces in the Solution explorer, and navigate to Universal Windows->Extensions
   2. Add the “Windows IoT Extensions for the UWP”. If there are two, select the later one.



1. Below the main window menu bar, make sure the configuration is still selected to be debug, and the platform is still ARM



1. Click Save all (icon of two floppy disks), and Build->Build Project. It should output something like this:

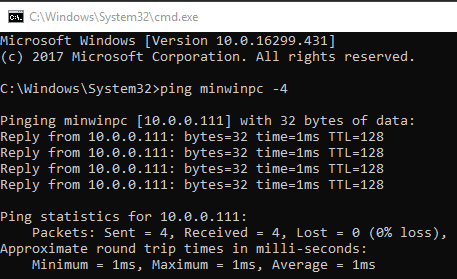


(Yours should say Build All instead of Rebuild All)

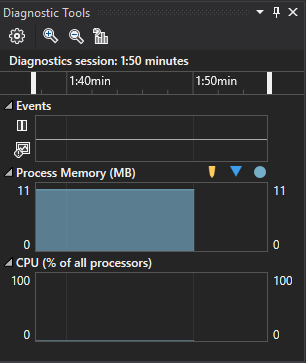
1. The application is ready for deployment. Show to your instructor that all project configurations in all tabs (as well as the top one) are correct, the capabilities are declared in the manifest file, the code with the breakpoint has been made, and that you have added the latest Windows IoT extensions.

**Part 4 – Deploying the app and using the breakpoint**

1. Before you deploy the app, it is recommended that you restart visual studio and verify connection with the device via the ping command.
   1. Open CMD.exe and type “ping minwinpc -4” without quotes. A reply should come back 4 times. If it says anything else, the device is not on the network.



1. Either click the “Remote Machine” button with the green play button, or Build->Deploy Solution to start deployment.
   1. If you get errors about the device failing to connect, you should try restarting (in order):
      1. Visual Studio
      2. The Pi
      3. Your PC
   2. It will take a while to deploy. The deployment process is rather slow, and since this is the first deployment, it will need to install additional frameworks on the device.
2. On a successful deployment, your Pi should be running the application on the display, and you should be able to see diagnostic information from the Diagnostic tools pane.
   1. If the pane says something about needing to restart the device, then do so. This is because it just installed the missing frameworks and needs a reboot.



1. When you click the button, the device will hit the breakpoint and you can step through the code line by line.
   1. Use the Buttons on the same line as the configuration manager to step into, over, and resume the application.

https://puu.sh/B6qJM/ba1ee28dcb.png

1. The screen should now say the new text in the text box.
2. Demo to your instructor the application running, the diagnostic tools working, and the breakpoint being hit.

**What to hand in**

* **Screenshot of visual studio running in debug mode with the application deployed**
* **MainPage.xaml**
* **MainPage.xaml.cs**

**Submit all materials to bb on a single word document**