Contents

[Prerequisites 2](#_Toc491009863)

[Recommended Windows Programs and Features 2](#_Toc491009864)

[Show real filenames 2](#_Toc491009865)

[Install an Archiving Program 3](#_Toc491009866)

[Install a Text Editor Program 4](#_Toc491009867)

[Blocked Files 4](#_Toc491009868)

[Setting Up a Project Folder 4](#_Toc491009869)

[Setting Up Visual Studio 4](#_Toc491009870)

[Install Visual Studio 5](#_Toc491009871)

[Apply Class Settings (Optional) 6](#_Toc491009872)

[Setting Up GitHub 7](#_Toc491009873)

[Creating a Profile 7](#_Toc491009874)

[Creating a Class Repository 7](#_Toc491009875)

[Managing Repositories 7](#_Toc491009876)

[Cloning a Repository 8](#_Toc491009877)

[Cloning from Visual Studio 8](#_Toc491009878)

[Cloning from Command Line 8](#_Toc491009879)

[Making Changes 8](#_Toc491009880)

[Commit Changes 8](#_Toc491009881)

[Committing from Visual Studio 8](#_Toc491009882)

[Committing from Command Line 9](#_Toc491009883)

[Pushing Changes 9](#_Toc491009884)

[Pushing from Visual Studio 9](#_Toc491009885)

[Pushing from Command Line 9](#_Toc491009886)

[Pushing from Github 9](#_Toc491009887)

[Getting Changes 9](#_Toc491009888)

[Pulling from Visual Studio 9](#_Toc491009889)

[Pulling from Command Line 10](#_Toc491009890)

Setting up your machine for this class is not difficult but will require some time. This document discusses how to set up your machine for this course and provides some suggestions for simplifying the work you will do in this class. The screenshots are from Windows 10 and Visual Studio 2015. Your screen may vary.

# Prerequisites

Before you begin note the following prerequisites for your environment.

1. Windows 7 SP1 or higher or Windows Server 2008 R2 SP1 or higher
2. 1.8 Ghz or faster processor (dual core or higher strongly recommended)
3. 2 GB memory (4 GB or more strongly recommended)
4. 10+ GB hard drive space
5. Fast internet connection for downloading software
6. Additional requirements may be found [here](https://www.visualstudio.com/downloads/). Note that older operating systems may require additional Windows updates to be installed.

Special note: Mac, Linux and Android are not supported operating systems. These platforms are not supported and no assistance can be given if you choose to use them. Visual Studio Code and Visual Studio for Mac currently do not support the necessary project types used by the course. You may be able to them however most of the tools discussed below will not work and no support can be given for its use. You are responsible for ensuring any labs completed using other version of Visual Studio work correctly in VS 2017 as well.

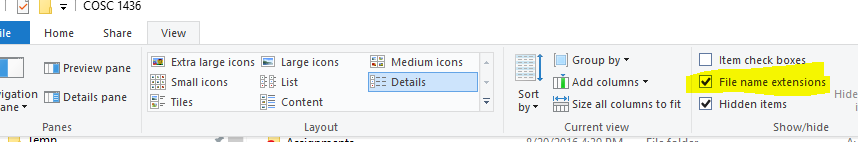
If you are unable to meet the above requirements then you will need to use the Computer Lab available on campus.

# Recommended Windows Programs and Features

Windows is designed for a novice user. Programming is considered advanced user tasks and therefore the default Windows settings and programs are not ideal. The following are recommended changes you make to your computer to simplify your programming efforts.

### Show real filenames

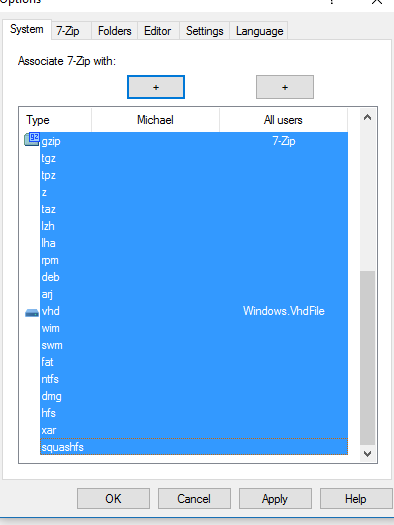
Out of the box Windows does not show you the real name of files. Most files have a file extension that help identify the type of file but users are generally not concerned with this. We will be and therefore showing the full filename is important.

1. Open Windows Explorer (not Internet Explorer).
2. For Windows 10
   1. Go to the View tab
   2. Check the option for *File name extensions*
3. For older Windows versions
   1. Press the Alt key to bring up the menu for Explorer and select Tools\Options
   2. Go to the View tab
   3. Uncheck the option to *Hide file extensions for known file types*
4. All files shown in Windows Explorer will now show the (generally 3 letter) file extension after the file name (ex. Explorer.exe).

### Install an Archiving Program

Windows ships with built in support for archive (.zip) files but there are several problems with it. For one it is really slow even for small files. Secondly it shows the file as a folder which gives you the illusion of it actually being a folder in which you can double click files. But this rarely works and more likely will cause errors so replacing the standard Windows archive program with a better one is recommended.

One freely available and solid program is 7-Zip. I recommend that you install it (or another tool).

1. Download the [program](http://www.7-zip.org/). For most computers the 64-bit exe version is the correct version.
2. Run the setup program to install it. (Warning: Free programs tend to offer additional software during their installation. Read each screen carefully and only install the core program. Do not install additional software such as Yahoo Toolbar or Bing, etc.)
3. After installation start the program as an administrator (right click the icon and select *Run as Adminstrator*).
4. Go to Tools\Options.
5. Select all the files in the list (click the first item, hold down Shift and click the last item) and then click either the option to associate all files just for you or for all users. Then click Apply.  
     
   
6. Now when you are working with any .zip file it will use 7-zip. To extract a .zip file right-click and select one of the Extract menu options.

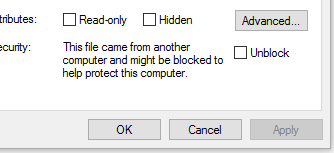
### Install a Text Editor Program

Windows ships with a couple of text editors and you likely also have Word but they are either too simple or too complex. There are many freely available text editors but I recommend Notepad++.

1. Download the [program](https://notepad-plus-plus.org/).
2. Run the setup program. (Warning: Refer to the warning earlier about setup programs) You don’t need to install anything other than the core program files but you may want to review what other options are available.
3. Now you can right-click any text file and select the option to open in Notepad++. It is recommended that you set Notepad++ as the default program for most text files.

### Blocked Files

Windows is designed to be secure by default. When downloading a file from the Internet Windows may mark the file as blocked. This is to protect you from running dangerous files. In general, trying to access a blocked file will fail. To check whether a file is blocked or not do the following.

1. Open Windows Explorer and locate the file.
2. Right click the file and select Properties.
3. On the General tab, if the file is blocked you will see a message about it being blocked and an option to unblock it.  
     
   
4. Click the option to unblock the file.

### Setting Up a Project Folder

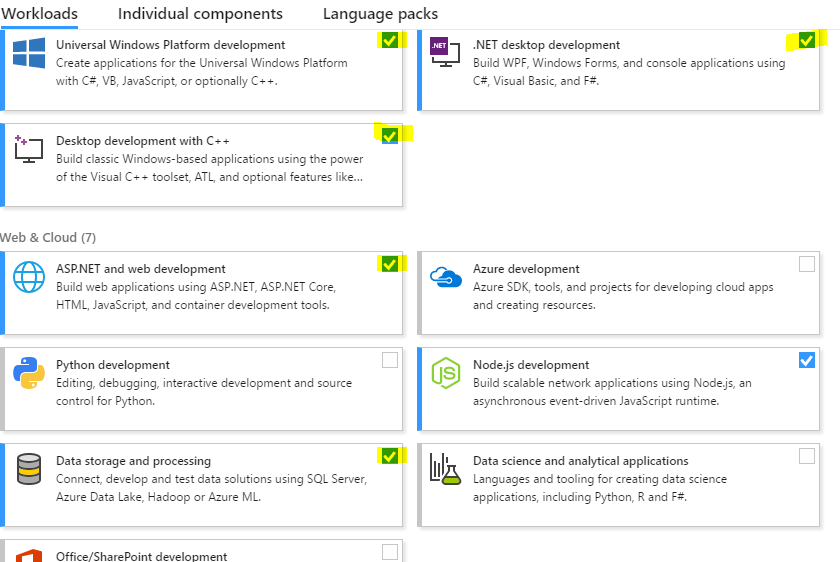
Most programs in Windows limit how long file paths can be. Visual Studio can easily test those limits. To prevent this from happening and to make it easier to find your labs it is recommended that you create a new folder in your Documents folder for labs.

1. Go to Windows Explorer.
2. Go to the Documents folder.
3. Right-click and select New\Folder.
4. Name the folder ITSE1430 or similar. Remember this path as you will need it later. The easiest way to do this is to click in the address bar (to the right of any text) and select the entire path.

# Setting Up Visual Studio

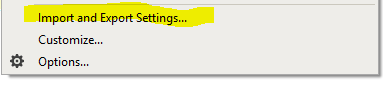
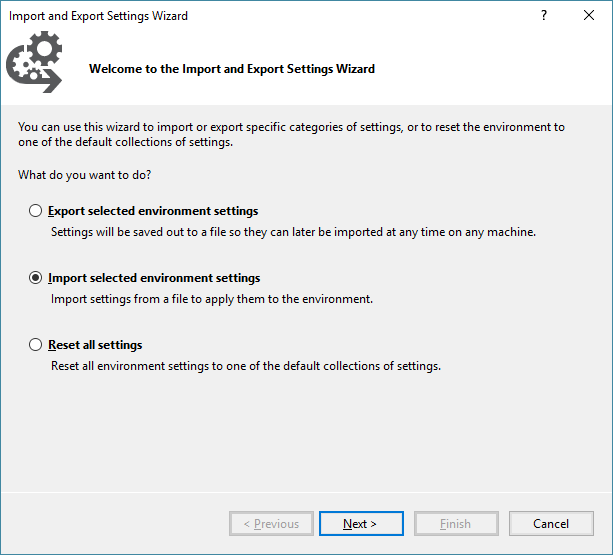
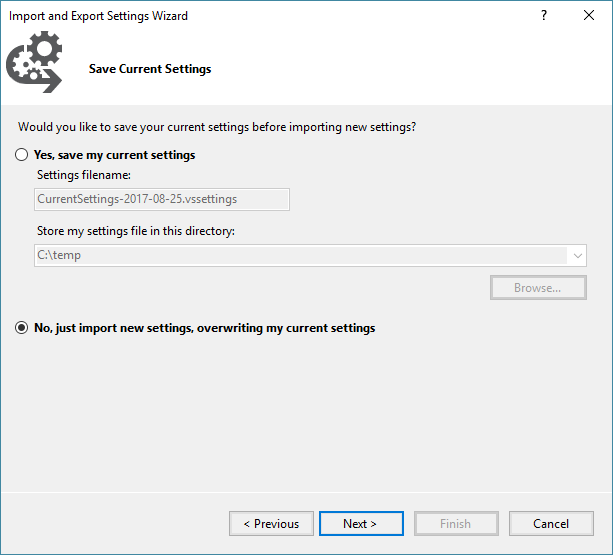
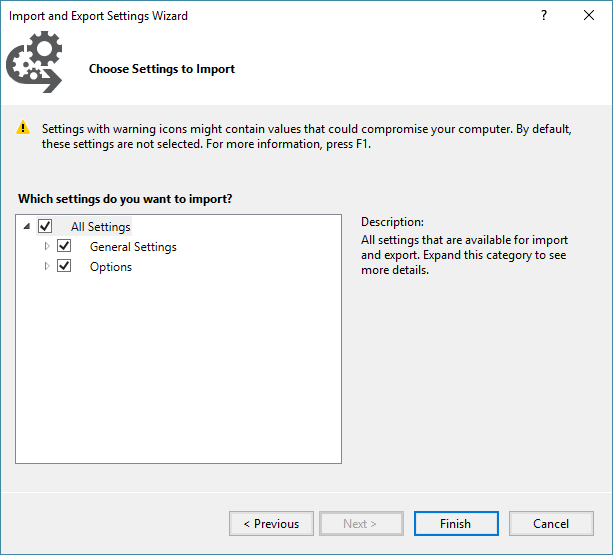
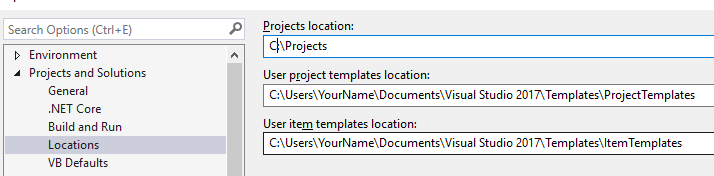
Setting up Visual Studio will take a while and involves many steps.

### Install Visual Studio

1. Download Visual Studio
   1. Community Edition is freely available. A link is available in the Links section of MyTCC.
   2. Alternatively you can download the Professional edition of VS through Imagine when it is available. There is no difference between the 2 editions.
2. Install Visual Studio
   1. Run the setup program.
   2. Select the following Workloads.
      * .NET desktop development
      * ASP.NET and web development
      * Data storage and processing
   3. Other workloads may be installed if desired.
   4. Finish the installation. This will take a while as most files will need to be downloaded.
3. Run Visual Studio for the first time.
   1. Start Visual Studio. This will take a while the first time.
   2. When prompted for a profile choose Visual C# for this course.
   3. Once Visual Studio has started then everything should be working correctly.
4. Ensure the latest update is installed (this should occur naturally during the installation above).
   1. After starting Visual Studio it will look for updates. A flag in the top right corner will indicate if there are updates available.  
        
      
   2. Clicking the flag will open the Notifications window, select each update for installation.
   3. For most extensions, closing Visual Studio will start the VSIX installer to install them.
   4. Some extensions may instead start the Windows Installer. The installer will require that Visual Studio be closed first.
   5. For Visual Studio updates the Visual Studio Installer will run. Usually the installer will need to update itself first. Before the installer can update Visual Studio you will need to close it.

### Apply Class Settings (Optional)

For this class I am providing a set of default settings for Visual Studio to save you from having to configure everything.

1. Go to the Tools folder in Github (link provided in MyTCC).
2. Download the vssettings file.
3. Start Visual Studio.
4. Click the *Tools\Import and Export Settings* menu item.  
     
   
5. Select the *Import selected environment settings* option and click *Next*.  
     
   
6. Select the *No, just import new settings* option and click *Next*.  
     
   
7. Using the Browse button locate the vssettings file you downloaded from GitHub. Then click *Next*.
8. Ensure that all check boxes are checked and click *Finish*.  
     
   
9. Visual Studio should now be set up to simplify your programming.
10. To save you time later you may want to have Visual Studio automatically save to the project folder you created earlier.
    1. Go to Tools\Options in the menu.
    2. Go to Projects and Solutions\Location.
    3. In the *Projects location* field put the path to the labs folder you created earlier (the path you copied from Windows Explorer).  
         
       
    4. Click *OK* to save the changes.

### Apply Standard Styling Rules (Optional)

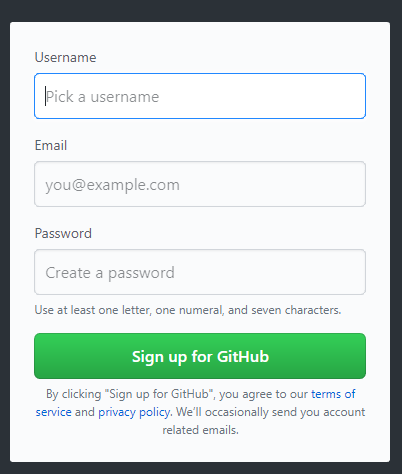
In C# it is important to follow the standard styling rules that other programmers, and the framework itself, follow. To simplify this and help you identify inconsistencies in your code there is a configuration file that you can apply that Visual Studio will use. This configuration file is set up to enforce most of the standard styling guidelines that C# uses. It also provides some suggestions on using some of the newer language features. To use this configuration file, do the following.

1. Go to the Tools folder in Github (link provided in MyTCC).
2. Download the *.editorconfig* file.
3. Copy this file to the root of your project folder you created earlier.
4. Visual Studio will now use the rules in this file for any project contained under it.
5. While writing code the editor will display suggestions, warnings and errors related to the code you are writing and any code that does not follow the settings in the configuration file. None of these warnings or errors will prevent your code from compiling.

# Setting Up GitHub

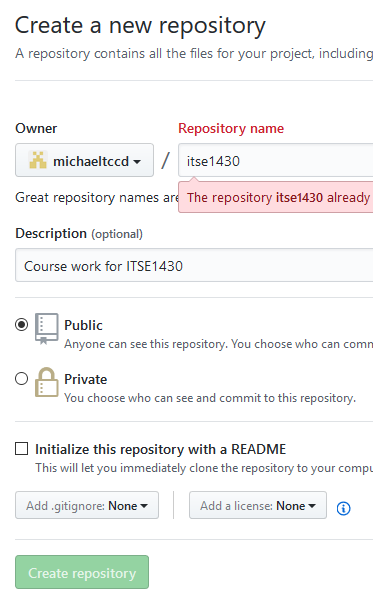
## Creating a Profile

If you do not yet have a Github account associated with your MyTccd email then do the following.

1. Go to Github (<https://github.com>).
2. Enter a username (ex. Mynametccd). Note that the user name is generally part of the URL you will share so use a generic user name.
3. Enter your MyTccd email address.
4. Enter a password.
5. Click Sign up for GitHub.  
     
   

## Creating a Class Repository

In order to store and share code you will need to create a repository. A repository is where you will store your lab assignments. You may also use it to store your class work if desired. Bear in mind that everything in the repository is visible to others.

1. In the top right corner, you will see a plus sign next to your profile.   
     
   
2. Clicking the plus will give you the option of creating a new repository.  
     
   
3. For the repository name enter ITSE1430.
4. Ensure the repository is public so that the repository can be seen by others. It is free.
5. Click the create button to create the repository.

## Managing Repositories

Git is a file-based system. Git will automatically add, update and remove files based upon changes in the file system. To have Git update you need to give it commands. A full discussion of git is beyond this document but the general flow is that you will clone a repository to your local drive. You will then make changes to the files and folders in the directory. Once you have made the changes you want you will commit them with a message. At this point, everything is happening locally on your machine. No other user or machine will be able to see these changes. Finally, you will push the changes to Git which will persist them to the repository and allow others (or yourself) get the changes. For this process, you can use either a command line Git tool (<https://git-scm.com/download/win> ) or Visual Studio.

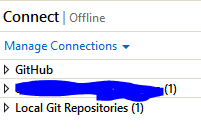
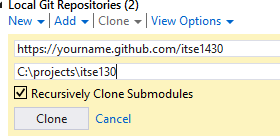
NOTE: You can determine if you are in a repository directory by looking for the hidden .git folder. This folder is always present and represents the local git repository. Never modify this folder or its contents.

NOTE: Git works on files, not folders. If you have empty folders then they are not committed. If you have a folder with no files in it then it will be ignored.

## Cloning a Repository

In order to work with a repository locally you need to clone it. Cloning a repository will set up the structure needed by git and will download the repository to your machine. This will generally only need to be done once per repository per machine. If you ever wipe out the directory structure you will need to repeat this process.

### Cloning from Visual Studio

1. Open Visual Studio and go to *Team Explorer*.
2. Click on *Projects \ Manage Connections*.  
     
   
3. You should see a *Local Git Repositor*ies list with options to New \ Add \ Clone repositories.
4. At this point you may be prompted to sign into Git.
5. Click the Clone option and enter the URL to the repository you are cloning and the local path you will be storing it in. Each repository must be in its own folder path.  
     
   
6. Clicking Clone will download the repository.

### Cloning from Command Line

1. Open a command prompt with access to Git
2. Navigate to the root folder where you want the repository to be placed (e.g. C:\Projects). Note that the repository will be placed in a subfolder of this folder.
3. Type the following command: git clone <url to repo>.
4. Git will clone the repository into the appropriate subfolder.

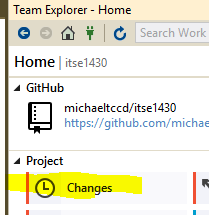
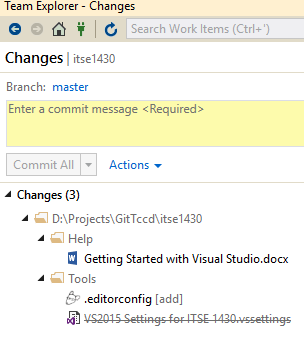
## Making Changes

You can now make changes to the files either using any command line tool or program or using Visual Studio. Git will determine what has changed later.

## Commit Changes

When you have made the changes are want to make them permanent you need to commit them. Committing changes in Git is equivalent to making a backup of the changes. Commits are not pushed to Git nor are they accessible on other machines even by the same user. They are committed to the local repository only. If you make changes that you do not want to push to the server then you can uncommit them if needed.

### Committing from Visual Studio

1. Under Team Explorer go to Changes.  
     
   
2. Visual Studio will list the changes that it has detected. Ensure these are the changes you want.  
     
   
3. For each change you can optionally undo the change, compare with the unmodified version and other actions to ensure the changes are what you want.
4. Once you’ve confirmed the changes you need to commit them to the staging repository so they can be pushed to Git in the future.
5. Enter a commit message describing why the change was made and then click Commit All.
6. The files in the Changes group will then be removed as they have been committed locally.

### Committing from Command Line

1. Using the command line, ensure you are in the repository directory.
2. Type the following command: git commit -a -m “message”.
3. Git will commit the changes to the local repository.

## Pushing Changes

Git is a distributed system. Any number of commits can be done before pushing them to the server. Once changes are pushed to the server they are visible to everyone and are accessible anywhere. Reminder that changes must be committed before they can be pushed.

### Pushing from Visual Studio

1. Under Team Explorer go to Sync.
2. Under Outgoing Commits you can see all the commits that occurred since the last push.
3. For each commit you can view the file(s) impacted by the commit.
4. To push the changes to Git click the Push command.
5. The changes should be pushed to Git and now accessible elsewhere.

If, for some reason, you decide that you do not want a commit to be pushed you can view the commit details and then select the Revert option to undo the commit.

You can optionally do the commit and push in one operation while committing by selecting the drop down next to the Commit button and selecting Commit and Push instead. That command does both the commit and push in one step.

### Pushing from Command Line

1. Using the command line, ensure you are in the repository directory and that you’ve committed any desired changes.
2. Type the following command: git push

### Pushing from Github

Using a web browser you can also push files to Git directly. This combines the commit and push commands into one. In general this should only be done when adding new files. You must provide a message for the commit.

## Getting Changes

If you are working on a repository with other people or even with yourself on multiple machines then you will need to periodically get changes back to your local machine. You can do that using pull.

### Pulling from Visual Studio

1. Under Team Explorer go to Sync.
2. Under Incoming Commits click Fetch to download changes that aren’t on the machine.

### Pulling from Command Line

1. Using the command line, ensure you are in the repository directory.
2. Type the following command: git pull