GSL Technical Setup Guide

This year, we'll be working with Android, Python, and web development. This guide teaches you how to install the tools necessary to develop for these platforms/languages. This guide also allows you to get set up on version control (which is used for collaboration) using Git.

Please complete all setup and installation on the computer and user account you wish to use for development.

Android

To get ready to program in Android, download Android Studio here: https://developer.android.com/studio.

Android Studio is the ESSENTIAL Android development tool. It has everything you need to develop for Android built in. From Android studio, you can test an emulator (virtual phone) or run on your Android phone. Android Studio also contains the Android SDK (software development kit), which is necessary for producing apps.

Python

There are many distributions of Python out there. The most common is cpython, which is the standard Python distribution. For this program, we'll be using **Anaconda**. Anaconda allows you to easily install new libraries and packages. It also provides a way for managing installed packages and libraries. In addition, Anaconda provides a graphical user interface (GUI) for accessing Python resources and creating environments (environments are isolated setups of python - you can think of them as offering the benefits of having separate user accounts or machines with Python installations and packages without actually needing to buy new machines or make new accounts).

You can install Anaconda here: https://www.anaconda.com/distribution/#download-section. You should download the Python 3.7 version.

You can install PyCharm if you'd like. PyCharm is an IDE that will make development easier, and it's made by the same people who made Android Studio.

We'll be using Python for server-side and back end development. In addition, Python is a good general-purpose language, meaning it's good for many different types of development. It's also good for quick coding.

Web Development

You'll be doing web development for your project. Regardless of the project type, you'll be making at the very least a web page to showcase your project and explain what it does. Some of you may be making web applications as well.

Web development can use a variety of tools. To make it easy to get packages and libraries necessary for web development, we'll install npm. npm is node package manager, and it helps manages packages for node/JavaScript, which is a web development language. It also manages other useful web applications.

In addition, we'll be using node, which is software for running JavaScript outside of the browser.

Node and npm come together (npm, after all, is the package manager for node).

You can get npm and node here: https://nodejs.org/en/

Text Editor

Having a text editor is extremely useful. It allows you to do general programming and write in a variety of languages. Text editors are often (and should be, at least for programming purposes) customizable.

There are many text editors out there, and choice of text editor varies from programmer to programmer.

We recommend the following:

VS Code: https://code.visualstudio.com/
Sublime Text: https://www.sublimetext.com/
Atom is also a good choice (https://atom.io/)

Git

Git is a version control tool for software. This means that it allows you to manage all the different versions of software that you create. Git also allows teams to collaborate. Since you'll be working in teams, this is really important.

You can get Git here: https://git-scm.com/. For Windows users, Git comes with a nice bash command line that will be extremely helpful in programming. Mac users and Linux users already have a bash command line.

Git can be a bit confusing when you first get started, ESPECIALLY if you're not familiar with bash. If you'd like to get familiar with bash, you can use this reference guide here: https://www.gnu.org/software/bash/manual/bash.html

If you'd like to get familiar with using Git on bash, you can use the reference guide here: https://git-scm.com/docs.

It's important to note that most programs and commands you can run in bash have **man** (manual) pages that explain how to use it. Type **man [program/command name]** to access.

ALTERNATIVELY (AND I HIGHLY RECOMMEND THIS FOR THIS PROGRAM), you can use a graphical git tool. I recommend using GitKraken. You can install GitKraken here: https://www.gitkraken.com/. (Install the Git Client, not Glo Boards.) GitHub also has GitHub desktop. You don't need both GitKraken and GitHub desktop. You should pick the one you feel more comfortable with. You do not have to have the same type as your team, although it can be helpful.

If you're on Windows, I recommend installing both Git and a Git graphical tool just so you have the command line. **If you do this, though, you should pick ONE tool to use for git use. I don't recommend switch back and forth as this could get messy.** If you're on macOS or Linux, I recommend not installing both.

You should also sign up for a GitHub account. GitHub is an online service that hosts Git repositories online so multiple people on a team can easily access them. If you want your repository to be accessible over the internet, you need to find a way to host them online. Typically using a service is the best way to go. GitHub is an easy service that hosts repositories and is widely used in the developer community. In addition, you can make your code public should you so desire (open source software is really important - you should think about the pros and cons of having open source software). Also, GitHub can be a place to showcase your code to outside parties, such as job recruiters.

You can sign up for a GitHub here: https://github.com/