**Pre-Lab 0: Setup your working environment**

**Lab Objectives**

The aims of this pre-lab session are:

1. To get familiar with Git and GitHub, a distributed version control software
2. To get familiar with VScode, a powerful code editor.

**Background**

Have you ever been in the situation where you wanted to modify existing code to test a new idea, but did not want to mess up the workspace at the same time? Or you would like to cooperate with other programmers but do not want to resolve code conflicts manually. The combination of Git and GitHub is a great solution for those situations. Git itself can record changes over time without overwriting previous change and GitHub is the collaboration platform built on top of Git, taking care of pull requests, comments, reviews, integrated tests, and so much more.

**Terminology**

Visual Studio Code (VSCode) is a code editor to develop, run and debug code.

Git is the tool that lets you create a local repository (on your PC) and manage versions of your files.

Github is an online service that will host your Git repositories (in the cloud).

**Task 0 – Get your GitHub account ready**

Open <https://github.com/> in your favorite browser and create your own account if you haven’t done so. It is recommended to use your ITSC account because you can use it to apply for the GitHub Student Developer Pack(<https://education.github.com/pack> ) later.

**Task 1 – Download Git and install it locally**

Before proceeding to the following instructions. Please check if Git is already installed. you can go in the command line (on Windows), or in the terminal (on Mac and Linux) and type this command.

git –-version

If it is installed already, you should see something similar to the figure below and you can skip this section. Otherwise, please continue.



To install Git, follow the instruction described in <https://github.com/git-guides/install-git> .Please pay attention to your platform. For Windows user, see <https://github.com/git-guides/install-git#install-git-on-windows>. For Mac user, see <https://github.com/git-guides/install-git#install-git-on-mac>. To test if Git is installed successfully, please repeat the same procedures mentioned above.

**Task 2 – Use Git together with GitHub**

Here is a really good tutorial for you to get started with Git and GitHub. You can have a basic understanding of them within one hour by finishing <https://lab.github.com/githubtraining/introduction-to-github>

Graphical user interface, text, application, email

Description automatically generated

Attention: Please click the additional Options and choose either the command line interface (CLI) or Visual Studio Code. If you are using UNIX system (Mac and Linux), it is recommended to choose CLI. Because being familiar with shell and shell commands are really important for your future study as a software engineer. If you are using Windows and new to command line, you can start from Git Extension in VScode.

**Task 3 – Setup Arduino environment in VScode**

VScode is a powerful, a cross platform source code editor with powerful intellisense, code completion and debugging support as well as rich extensions can be installed from marketplace. Whenever you want to add some functionalities to it, do a quick search in Marketplace first.

1. Install Arduino IDE(https://www.arduino.cc/en/software). It provides the necessary toolchain for compiling and uploading Arduino code.

* Windows: Use Windows Installer version. Do not install from the App Store.
* macOS: Drag and drop the extracted Arduino.app into /Applications folder.
* Ubuntu: Unzip it into folder such as $HOME/Downloads/arduino-1.8.8

1. Install Visual Studio Code. Please follow the instruction in <https://code.visualstudio.com/> and install the latest version of VScode.
2. Launch VSCode, look for Arduino in the extension marketplace and install it. This extension provides enhanced experiences for developing on Arduino platform.

Graphical user interface, text

Description automatically generated

1. Configure VS Code with Arduino settings.

Graphical user interface, text, application

Description automatically generatedIn Visual Studio Code, click File > Preferences > Settings (on macOS, Code > Preferences > Settings). Then click the Open Settings (JSON) icon in the upper-right corner of the Settings page.

Add following lines to configure Arduino depending on your platform:

* Windows:

"arduino.path": "C:\\Program Files (x86)[\\Arduino](file://Arduino)"

* macOS:

"arduino.path": "/Applications"

* Ubuntu:

"arduino.path": "/home/{username}/Downloads/arduino-1.8.8"

**Task 4 – Write your first Arduino program**

In this section, we are going to implement an echo service in Arduino Nano.

1. Setup Arduino System

You have to change a few system settings before starting the program. First create a folder and open it in the VScode. Then create a .ino file with the same name as the parent folder. After this step, you should have a file structure like this:

<name>\ //workspace directory in VScode

<name>.ino

Once you have opened up an \ .ino file, VS Code reconfigures in an Arduino mode, and gives access to special functionality in its bottom blue margin, as shown in the image below.



Text

Description automatically generatedClick on <Select Programmer> and select AVR ISP (Arduino AVR Board).

To set the serial port that VS Code will communicate with your Arduino Nano we need to open up the Command Palette (Cntrl+Shift+p / Cmd+P) and then type “Arduino Select Serial Port”. You should then select which USB port your Nano is connected to.

Background pattern

Description automatically generated

Graphical user interface, application

Description automatically generatedTo set the board type, Click on <Select Board Type> and select Arduino Nano.

1. Code a serial echo service

Arduino boards have built in support for serial communication on pins 0 and 1, and they are bridged to USB port by a USB-TTL chip internally. This means that you can access the Arduino's serial port from your computer by simply connecting a USB cable.

Echo service is to send anything received on the RX back to TX. That means whatever you send to the Nano board will appear in the received window.

Here are all the functions you need for this task. Please search the official Arduino documentation(<https://www.arduino.cc/reference/en/> ) to see what features they offer.

Serial.begin()

Serial.available()

Serial.write()

Serial.read()

1. Verify and Upload Your Code

A screen shot of a computer

Description automatically generated with low confidenceVS Code in Arduino mode provides a couple of convenient icons for you to click for code verification and uploading to your connected Arduino board. These are available in the upper right-hand corner. Either operation will result in a console window appearing at the bottom of VS Code informing you of the outcome.

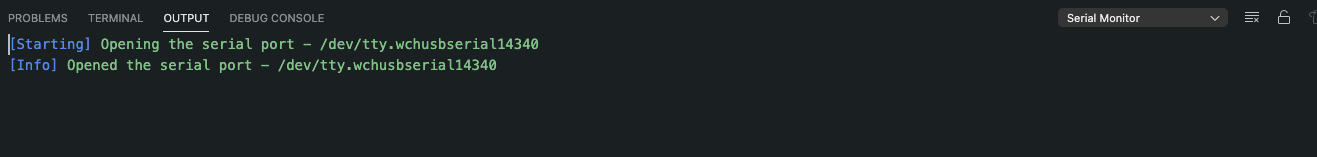
1. To test your program:

Make sure the Nano board is connected to your PC.

Click on the plug symbol on the blue line to open the Serial Monitor

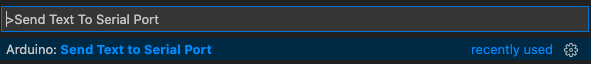
If everything is working properly, you should see something similar to





Open up the Command Palette (Cntrl+Shift+p / Cmd+P) and type:

>Send Text To Serial Port



Press Enter and type whatever you want to send to your board.

A picture containing text

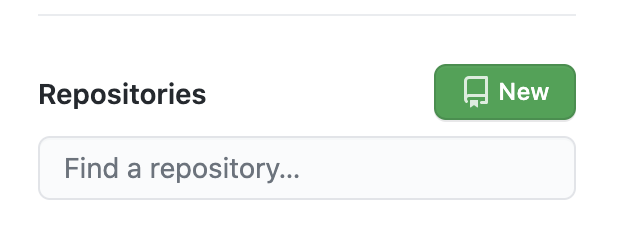
Description automatically generatedYou should see the same message appear in the Serial Monitor.

**Task 5 – Commit your program in Git and upload it to GitHub**

After you finish the program, you have to commit your changes and push them to the remote. By doing so, you can keep a record of the current version and it is easy for you to rollback.

1. Create a new repository in GitHub

Go to [https://github.com/](https://github.com/H) and create a new repository via GitHub Web GUI.



Graphical user interface, text, application, email

Description automatically generatedGive your repository a memorable name and leave the rest as default. Procced to the next step.

1. Follow the instruction listed below and upload the local repo to the remote



Replace it with your own URL

**Task 6 – Create a new branch and add a new feature**

1. Create and checkout to a branch named feature-LED
2. Add LED control to the existing code.

In this section, you are required to add LED control feature based on the echo service you write before.

Besides the basic functionalities, you are required to check the content sent to the board.

Pseudo code

If ‘1’ is received,

turn on the on-board LED.

else If ‘0’ is received,

turn off the on-board LED.

else

Echo the received message

Here are some the functions you may need for this task. Please search the official Arduino documentation(<https://www.arduino.cc/reference/en/> ) to see what features they offer.

Serial.begin()

Serial.available()

Serial.write()

Serial.read()

digitalRead()

digitalWrite()

pinMode()

You can also look through Arduino built-in examples(https://www.arduino.cc/en/Tutorial/BuiltInExamples/Blink) before starting this task

1. Commit your changes and push the new branch to the remote.