

CST0006 – Computer Programming Foundations

Introduction to LAB00

In today's lab we will be preparing our laptops and getting all the resources we will be using in this course set up. For some of you this may take up the full two hours. For others it may only take about thirty minutes. It's important that you complete each step, as we will be using all of these components to do the rest of our labs for the whole semester.

Installing the IDEs

For this class we are going to use the Thonny IDE for the Python programming language and Code::Blocks for the C programming language. Both of these IDE are simplistic and powerful. They will get you used to the concepts of an IDE and managing the code in software development projects.

Install Thonny:

Download Thonny for Windows from here: <https://bitbucket.org/plas/thonny/downloads>

Chose the latest **thonny-*<#.##>.exe*** release, download and install.

If you are using Fedora, the command to install Thonny is:

```
dnf install thonny -y
```

Once installed, run Thonny from your applications menu. Thonny will then prepare the virtual environment that it needs (wait for this step to finish).

Install Code::Blocks:

Download Code::Blocks for Windows from here: <http://www.codeblocks.org/downloads/binaries>

Chose the latest **codeblocks-*<##.##>mingw-setup.exe*** release, download and install.

If you are using Fedora, the command to install Code::Blocks is:

```
dnf install codeblocks -y
```

Install Git

Download Git for Windows: <https://git-scm.com/download/win>

The latest Windows release should automatically start, download and install.

In Fedora, the command to install Git is:

```
dnf install git -y
```

Generate an SSH Key Pair

Using SSH key authentication is far more secure than using a standard password, so we'll create an RSA key-pair for our user to use.
This step is to be done on your laptop

For Windows users, please download the other file called “**CST0006_LAB00_generating_rsa_keys_in_windows.pdf**” as it will give you a visual walk-through for installing OpenSSH and all the extra steps needed to do these next couple of steps written here for Linux and Mac users.

If you are on a Linux or Mac machine, please open the windows tutorial, so that you can see what it takes to do these things in Windows.

Create an SSH key on Linux/Mac

```
ssh-keygen -t rsa
```

Press Enter to use the default names id_rsa and id_rsa.pub in /home/<username>/.ssh
Enter your passphrase and confirm it by entering it again.

Add this new key to your profile

```
ssh-add
```

Enter the passphrase and <Enter> to add the ssh key to your profile

Create an Account on Gitlab

Go here and create an account: https://gitlab.com/users/sign_in

Use your real name and a real email address. You will be using this in your future and if not, for the rest of this class. You may use your school email. You can change this at any time.

Upload your Public Key to Gitlab

Login to Gitlab.com

Click the icon in the top right corner of the Gitlab menu and choose “Settings”.

On the left-hand column, choose “SSH Keys”

Now comes the hard part...

Locate the “.ssh” folder in your home directory.

For Windows users: C:\Users\<username>\.ssh

For Linux users: /home/<username>/.ssh

For Mac users: /Users/<username>/.ssh

Within the .ssh folder there is a file called “id_rsa.pub”, open it with a text editor.

- If you are using Windows and you don’t see this file, you may need to enable file extensions so that you can see the .pub extension.
- If that file doesn’t exist at all, you didn’t follow the steps within **Generate an SSH Key Pair** and may need to do that step again.

Copy the **ENTIRE** contents of the “id_rsa.pub” file to the clipboard. (**Ctrl+a** to select all, **Ctrl+c** to copy)

Paste this into the “Key” field of the SSH Keys window in Gitlab.

The title will be automatically generated. This is a reminder for you to know which key belongs to which user.

Click on “Add Key”

Create your first Repository

Click on the “Gitlab” icon on the top left of the Gitlab menu.

Click the “New project” button

Under the “Project name” field, create a project called “2018W_CST0006_LAB00”

Under the “Project description (optional)” type a description of this lab if you want.

Under the “Visibility Level”, make sure “Private” is selected

Click on the “Create project” button

Leave this window open, we will be using these commands to download a local copy of this repository.

Download the repository locally

If you’re using Windows, open the “Git Bash” terminal, if you are using Linux/Mac open the terminal

Under your home directory, create a folder called “git” (in lowercase letters).

```
mkdir git
```

Change Directory into git

```
cd git
```

Create a folder called “2018W”, (which will hold all of your coding projects for this term).

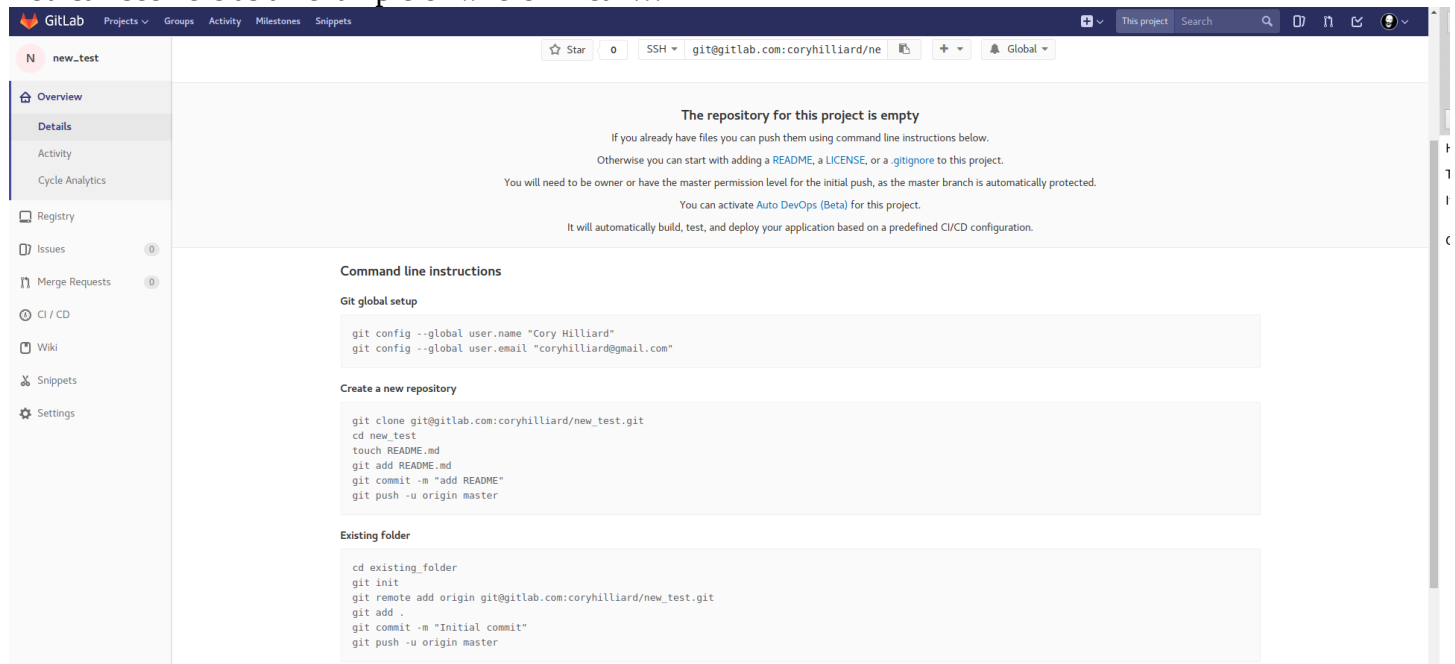
```
mkdir 2018W
```

Change Directory into 2018W

```
cd 2018W
```

The rest of the commands that you will be typing into “Git Bash” (or in the terminal if you’re not on Windows) are located in your Gitlab repository under “Git global setup” and “Create a new repository”.

You can see here as an example of where I mean...



The screenshot shows the GitLab web interface for a new repository named "new_test". The left sidebar contains navigation links: Overview, Details, Activity, Cycle Analytics, Registry, Issues (0), Merge Requests (0), CI / CD, Wiki, Snippets, and Settings. The main content area displays the repository's status and provides command-line instructions for setting up Git globally, creating a new repository, and cloning an existing folder.

The repository for this project is empty

If you already have files you can push them using command line instructions below.
Otherwise you can start with adding a [README](#), a [LICENSE](#), or a [.gitignore](#) to this project.
You will need to be owner or have the master permission level for the initial push, as the master branch is automatically protected.
You can activate [Auto DevOps \(Beta\)](#) for this project.
It will automatically build, test, and deploy your application based on a predefined CI/CD configuration.

Command line instructions

Git global setup

```
git config --global user.name "Cory Hilliard"
git config --global user.email "coryhilliard@gmail.com"
```

Create a new repository

```
git clone git@gitlab.com:coryhilliard/new_test.git
cd new_test
touch README.md
git add README.md
git commit -m "add README"
git push -u origin master
```

Existing folder

```
cd existing_folder
git init
git remote add origin git@gitlab.com:coryhilliard/new_test.git
git add .
git commit -m "Initial commit"
git push -u origin master
```