# Department of Computing

**CS312: Advanced Programming**

# Lab 1: Software Installation

**CLO3: Installation of different software that are used to develop applications**

**Date: September 06, 2018**

# Time: Thursday (10:00 - 12:50 & 14:00 – 16:50)

# Instructor: Abid Rauf

**Lab Engineer: Ayesha Asif**

# Lab 1: Software Installation

## Introduction

In this lab we will go through the steps for installation of different software/frameworks like JDK, Eclipse, Python, Kotlin, Visual Studio and some requirements for their installation, in order to develop application in these frameworks.

## Objectives

Installation of different frameworks and understanding how to use these tools.

## Tools/Software Requirement

None

**Description**

Each student must, individually install the complete application on their own systems. Students must upload their completion task report on LMS to qualify for evaluation. The following reasons can lead to losing marks

* Any exceptions or errors in software or missing files that leading to non-execution of running software / framework.
* Failure to upload the report on LMS.
* Failure to explain the installation steps or submission, during viva.

**Lab Task 1:**

Following are the installation steps for JDK:

##### **Step 0: Un-Install Older Version(s) of JDK/JRE**

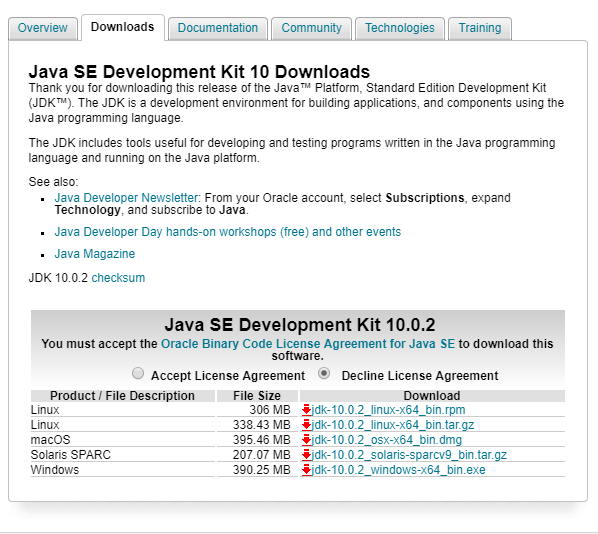
1. Recommendation that you install only the *latest* JDK. Although you can install multiple versions of JDK/JRE concurrently, it is messy.
2. If you have previously installed older version(s) of JDK/JRE, un-install ALL of them. Goto "Control Panel" ⇒ "Programs" ⇒ "Programs and Features" ⇒ Un-install ALL programs begin with "Java", such as "Java SE Development Kit ...", "Java SE Runtime ...", "Java X Update ...", etc.

##### **Step 1: Download JDK**

1. Goto Java SE download site @ <http://www.oracle.com/technetwork/java/javase/downloads/index.html>.



1. Under "Java Platform, Standard Edition" ⇒ "Java SE 10.0.{x}", where {x} denotes a fast running update number ⇒ Click the JDK's "Download" button.
2. Under "Java SE Development Kit 10.0.{x}" ⇒ Check "Accept License Agreement".
3. Choose the JDK for your operating system, i.e., "Windows" (for 64-bit Windows OS), and download the installer (e.g., "jdk-10.0.{x}\_windows-x64\_bin.exe" - 390MB).



##### **Step 2: Install JDK and JRE**

Run the downloaded installer (e.g., "jdk-10.0.{x}\_windows-x64\_bin.exe"), which installs both the JDK and JRE.

By default:

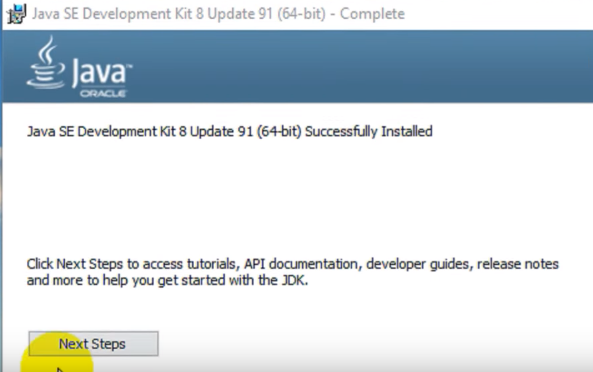
* JDK is installed in directory "C:\Program Files\Java\jdk-10.0. {x}", where {x} denotes the upgrade number; and
* JRE is installed in "C:\Program Files\Java\jre-10.0. {x}".

Notes: In 64-bit Windows, "Program Files" is meant for 64-bit programs; while "Program Files (x86)" for 32-bit programs.

Whereas you can change the default path by clicking on Change button displayed on the window.

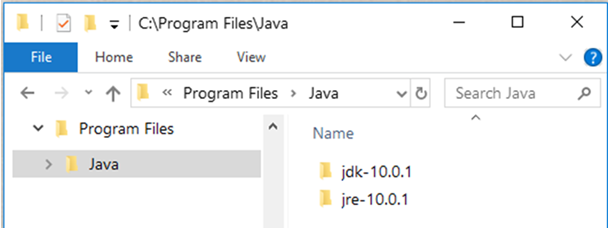
Accept the defaults and follow the screen instructions to install JDK and JRE. First JDK is installed then in same installation JRE window will be opened, accept the defaults or change accordingly your path but don’t change any other default options.

Click on “close” button in below window. JDK Installed successfully.



Use the "File Explorer", goto "C:\Program Files\Java" to inspect these folders. Take note of your **JDK installed directory**, the varying upgrade number, which you will need in the next step.

In the following diagram, the JDK installed directory is "C:\Program Files\Java\jdk-10.0.1", where {x}=1.

****

Your version can be different

**Step 3: Include JDK's "bin" Directory in the PATH**

Windows' Shell searches the current directory and the directories listed in the PATH environment variable (system variable) for executable programs. JDK's programs (such as Java compiler javac.exe and Java runtime java.exe) reside in the sub-directory "bin" of the JDK installed directory. You need to include "bin" in the PATH to run the JDK programs.

To edit the PATH environment variable in Windows 7/8/10:

1. Launch "Control Panel" ⇒ (Optional) System and Security ⇒ System ⇒ Click "Advanced system settings" on the left pane.
2. Switch to "Advanced" tab ⇒ Push "Environment Variables" button.
3. Under "System Variables" (the bottom pane), scroll down to select "Path" ⇒ Click "Edit...".
4. **For Windows 10 (newer releases)**:  
   You shall see a **TABLE** listing all the existing PATH entries (if not, goto next step). Click "New" ⇒ Enter the JDK's "bin" directory "c:\Program Files\Java\jdk-10.0.{x}\**bin**" (Replace {x} with your installation number!!!) ⇒ Select "Move Up" to move this entry all the way to the TOP.

**Prior to newer Windows 10**:

**(CAUTION: Read this paragraph 3 times before doing this step! Don't push "Apply" or "OK" until you are 101% sure. There is no UNDO!!!)**  
(To be SAFE, copy the content of the "Variable value" to Notepad before changing it!!!)  
In "Variable value" field, INSERT "c:\Program Files\Java\jdk-10.0.{x}\bin" (Replace {x} with your installation number!!!) IN FRONT of all the existing directories, followed by a semi-colon (;)which separates the JDK's bin directory from the rest of the existing directories. DO NOT DELETE any existing entries; otherwise, some existing applications may not run.

1. Variable name: PATH

Variable name: **c:\Program Files\Java\jdk-10.0.{x}\bin;[*do not delete exiting entries...*]**

**Notes:** Starting from JDK 1.8, the installation created a directory "c:\ProgramData\Oracle\Java\javapath" and added to the PATH. It contains only JRE executables (java.exe, javaw.exe, and javaws.exe), but NOT the JDK executables (e.g., javac.exe).

**Step 4: Verify the JDK Installation**

Launch a CMD shell via one of the following means:

1. Click "Search" button ⇒ Enter "cmd" ⇒ Choose "Command Prompt", or
2. right-click "Start" button ⇒ run... ⇒ enter "cmd", or
3. (Prior to Windows 10) click "Start" button ⇒ All Programs ⇒ Accessories (or Windows System) ⇒ Command Prompt, or
4. (Windows 10) click "Start" button ⇒ Windows System ⇒ Command Prompt.

Issue the following commands to verify your JDK installation:

1. Issue "path" command to list the contents of the PATH environment variable. Check to make sure that your <JAVA\_HOME>\bin is listed in the PATH.  
   Don't type prompt>, which denotes the command prompt!!! Key in the command (highlighted) only.
2. // Display the PATH entries
3. prompt> path

PATH=c:\Program Files\Java\jdk-10.0.{x}\bin;[other entries...]

1. Issue the following commands to verify that JDK/JRE are properly installed and display their version:
2. // Display the JRE version
3. prompt> java -version
4. Java version "10.0.{x}" 2018-04-17
5. Java(TM) SE Runtime Environment 18.3 (build 10.0.1+10)
6. Java HotSpot(TM) 64-Bit Server VM 18.3 (build 10.0.1+10, mixed mode)
7. // Display the JDK version
8. prompt> javac -version

javac 10.0.{x}

**Lab Task 2:**

Following are the installation steps for Eclipse 4.8 (Photon):

**Step 0: Install JDK**

To use Eclipse for Java programming, you need to first install Java Development Kit (JDK). Read "[How to Install JDK for Windows](http://www3.ntu.edu.sg/home/ehchua/programming/howto/JDK_HowTo.html#jdk-install)".(Lab Task 1)

**Step 1: Download**

Download Eclipse from [https://www.eclipse.org/downloads](https://www.eclipse.org/downloads/). Under "Get Eclipse Photon" ⇒ Click "Download Packages". For beginners, choose the 4th entry "**Eclipse IDE for Java Developers**" and "**Windows 64-bit**" (e.g., "eclipse-java-photon-R-win32-x86\_64.zip" 195MB) ⇒ Download.

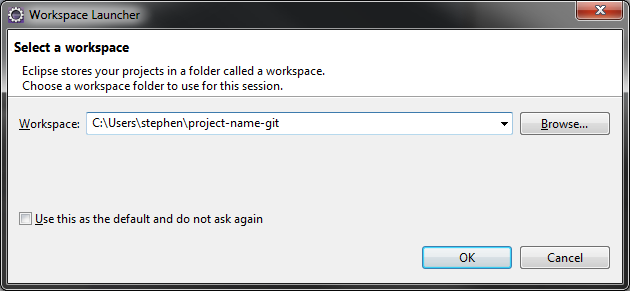
**Step 2: Unzip**

To install Eclipse, simply unzip the downloaded file into a directory of your choice (e.g., "c:\myproject").

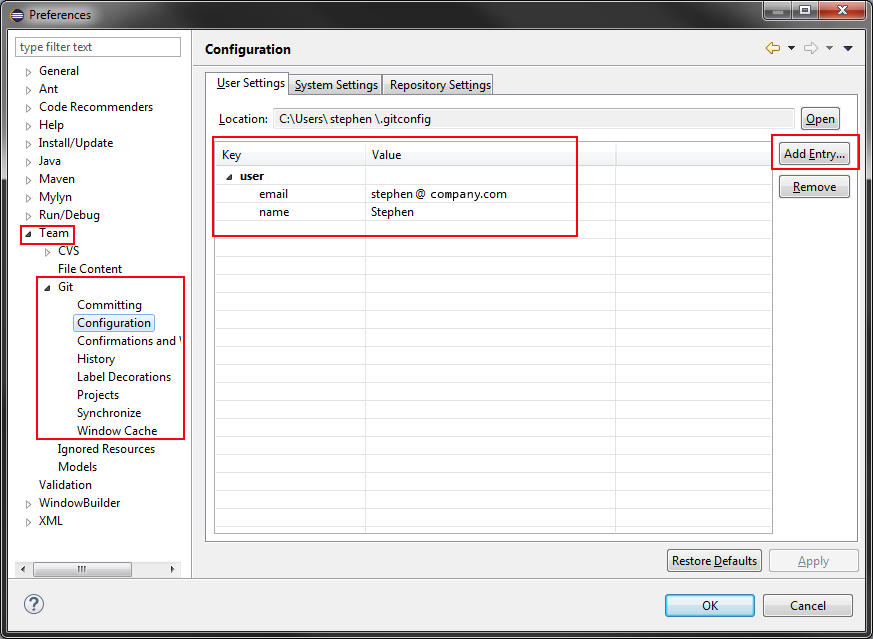
There is no need to run any installer. Moreover, you can simply delete the entire Eclipse directory when it is no longer needed (without running any un-installer). You are free to move or rename the directory. You can install (unzip) multiple copies of Eclipse in the same machine.

**Step 3: Create new Workspace**

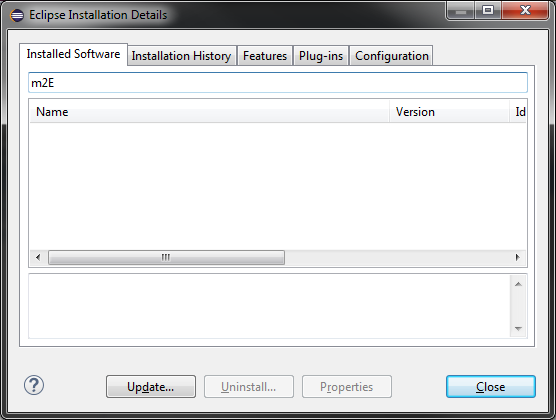
Created a brand-new workspace.



Then **(Window -> Preferences)** and saw that it recognized some settings for Git that you must have configured in a previous lifetime. If you haven't done this, just add some entries for your user settings (or System settings).



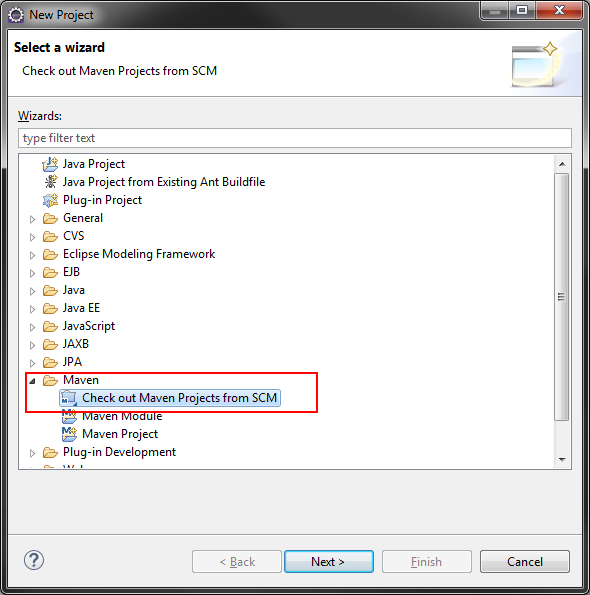
Eclipse has a handy feature called Installation Details and can be accessed by **Help -> Installation Details**



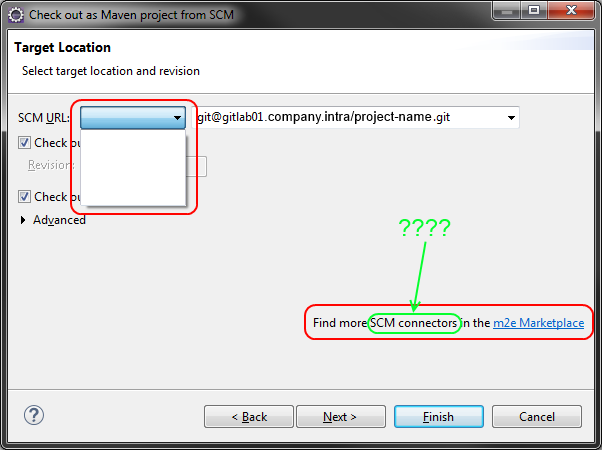
If Maven not present there you can try two options: you can install this plugin. Or you should just try a different Eclipse package, perhaps the 'Eclipse IDE for Java EE Developers' as that looks to include a bunch of Maven integration tools.

Checked the Maven settings (**Window -> Preferences**again) and saw that it picked up on my user settings.

Create a new project. Right-clicked on the empty Project Explorer view. New -> Project... -> Maven -> Check out Maven Projects from SCM.



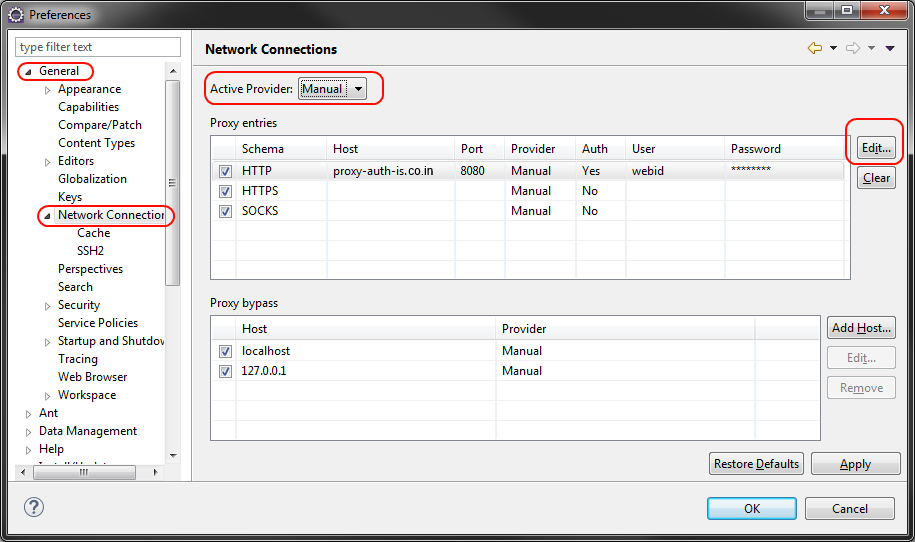
Enter in the URL from gitlab.



Click Next. At the next screen, accepted the default settings and clicked Finish.

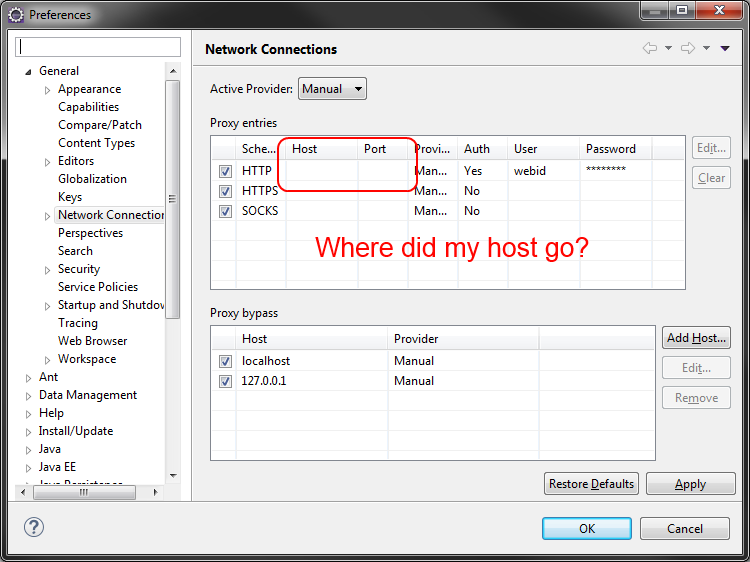
No projects or folders in the Project Explorer. Waite a moment. Maybe there's debugging info in the console. Open that up. Window -> Show View -> Console.

Find the proxy settings:

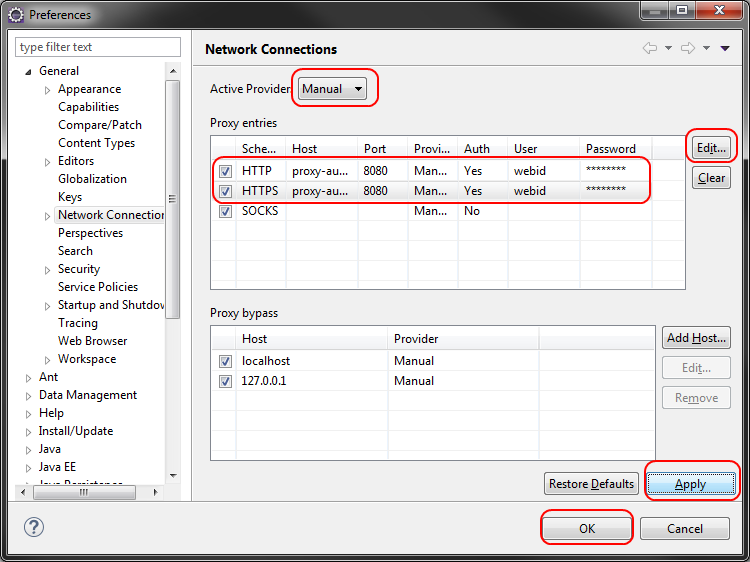
****

Hit apply and close it. Right-clicked on the empty Project Explorer view again. New -> Project... -> Maven -> Check out Maven Projects from SCM. Click on the m2e Marketplace link.

If Nothing happens. Go back to the Network Connections preference. The Host and Port are empty, but the User and Password are filled.

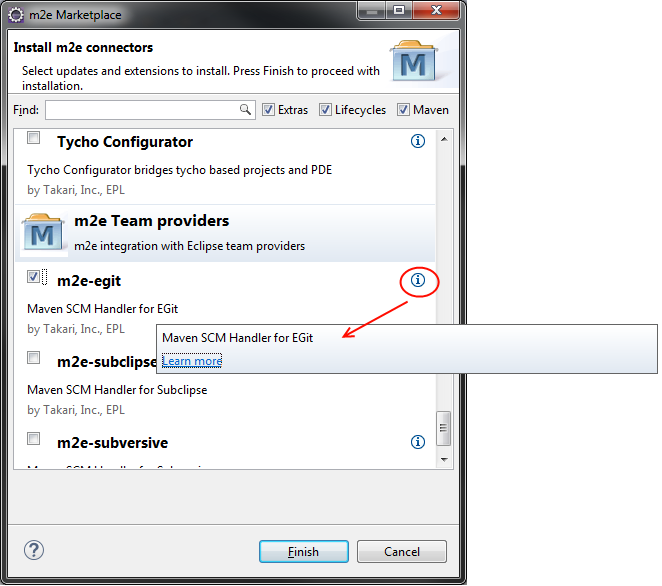


Try setting BOTH the HTTP and HTTPS proxy to use the same thing. Click Apply and OK again.

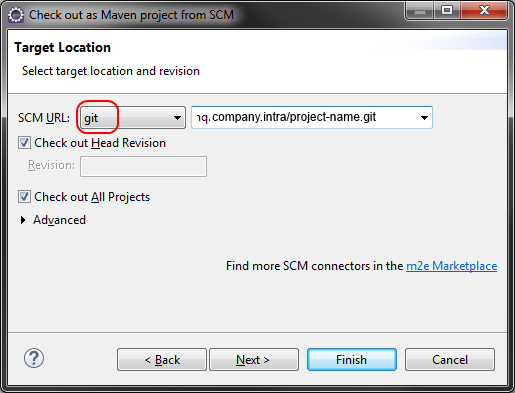


Right-clicked on the empty Project Explorer view again. New -> Project... -> Maven -> Check out Maven Projects from SCM. Click on the m2e Marketplace link.

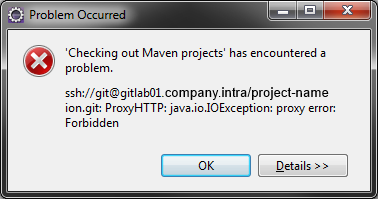
EGit is the name of the Git plugin for Eclipse.



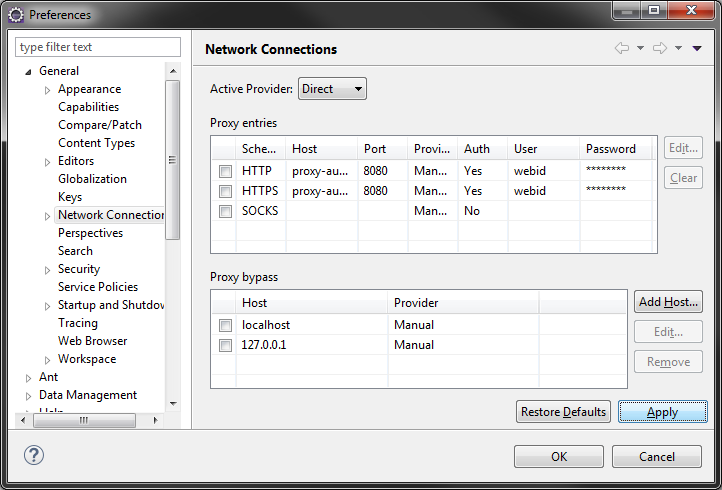
Right-clicked on the empty Project Explorer view again. New -> Project... -> Maven -> Check out Maven Projects from SCM.



Turns out that blank field. Click through these screens.



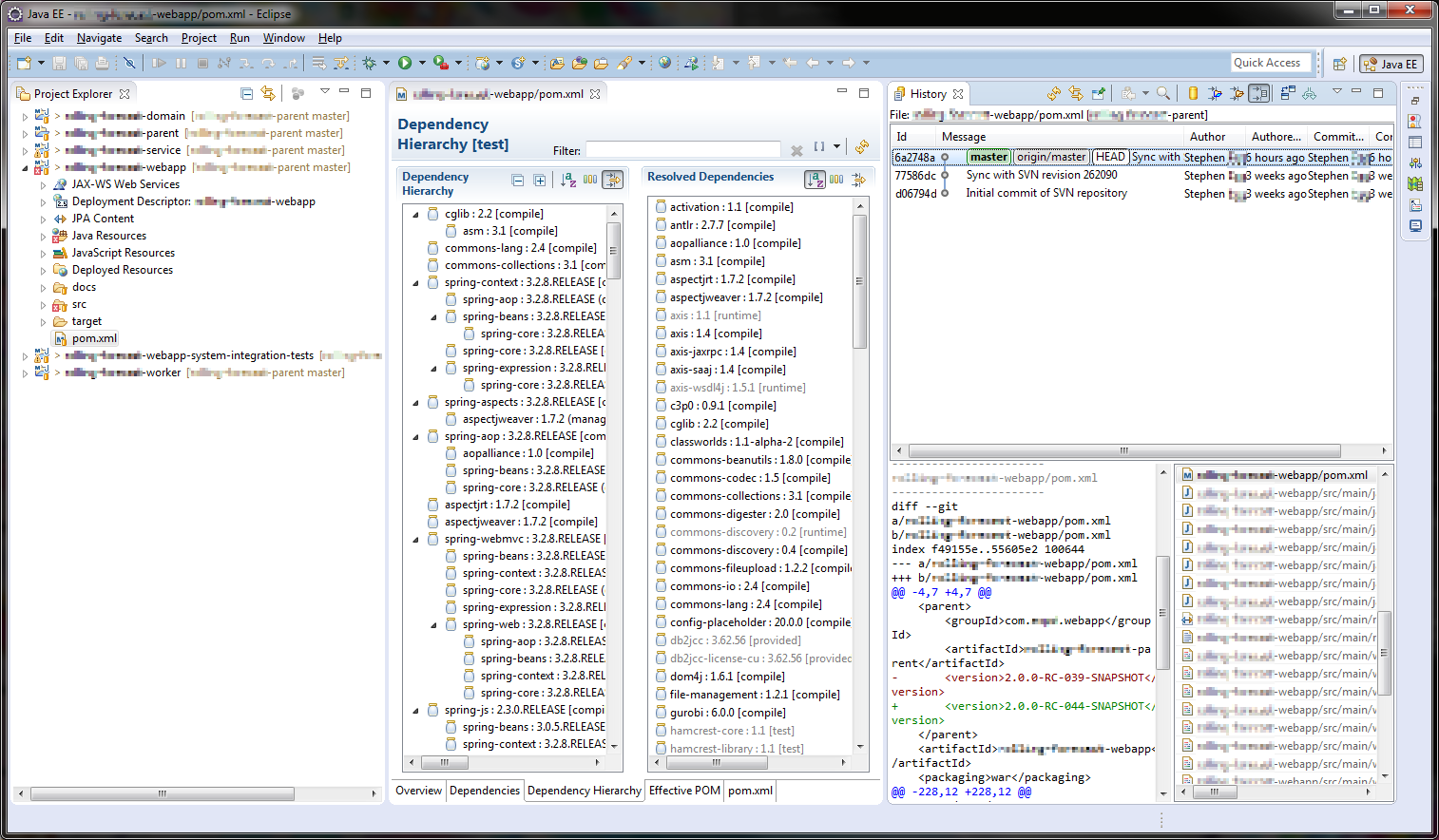
Looks like a proxy error, probably related to not having turned off the proxy, disabled the proxy for internal addresses (and this host is clearly an internal one). Both SSH/Git protocols should not be operating over HTTP. Proxy was only for HTTP and HTTPS. But change those settings again.



The proxy settings ignoring internal hosts correctly. Just clicked the proxy settings to say Direct and hit Apply then OK.

Right-clicked on the empty Project Explorer view again. New -> Project... -> Maven -> Check out Maven Projects from SCM. I click through these screens.

After a few minutes of copying the repository and building the project, everything is there. Maven recognizes it and the Git plugin recognizes it!



Creating new workspaces. Use [the second, not accepted answer here](http://stackoverflow.com/questions/2078476/how-to-share-eclipse-configuration-over-different-workspaces) to transfer your settings over quickly. Using a package manager / build tool to be able to checkout a project from scratch, resolve dependencies, and get it running in 5 minutes or less.

**Lab Task 3:**

Following are the installation steps for Kotlin:

[Step 1: Make sure your computer is ready and](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-1---make-sure-your-computer-is-ready-for-visual-studio) Android studio is installed on your computer.

[Step 2 - Download latest versio](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-2---download-visual-studio)n @ <https://developer.android.com/studio/preview/>

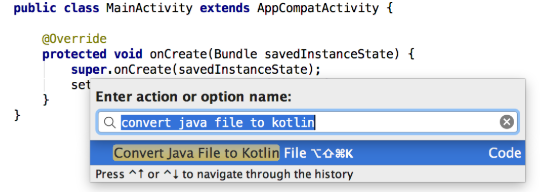
The Kotlin plugin is bundled with Android Studio starting from [version 3.0](https://developer.android.com/studio/preview/index.html). If you use an earlier version, you'll need to install the Kotlin plugin.

[Step 3 -](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-3---install-the-visual-studio-installer) If you are looking at the "Welcome to Android Studio" screen, choose Configure | Plugins | Install JetBrains plugin…

[Step 4 - Converting Java code to Kotlin](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-4---select-workloads)

Open MainActivity.java file. Then invoke action **Convert Java File to Kotlin File**. You can do it by several ways. The easiest one is to invoke [Find Action](https://www.jetbrains.com/idea/help/navigating-to-action.html) and start typing an action name (like in a screencast below).

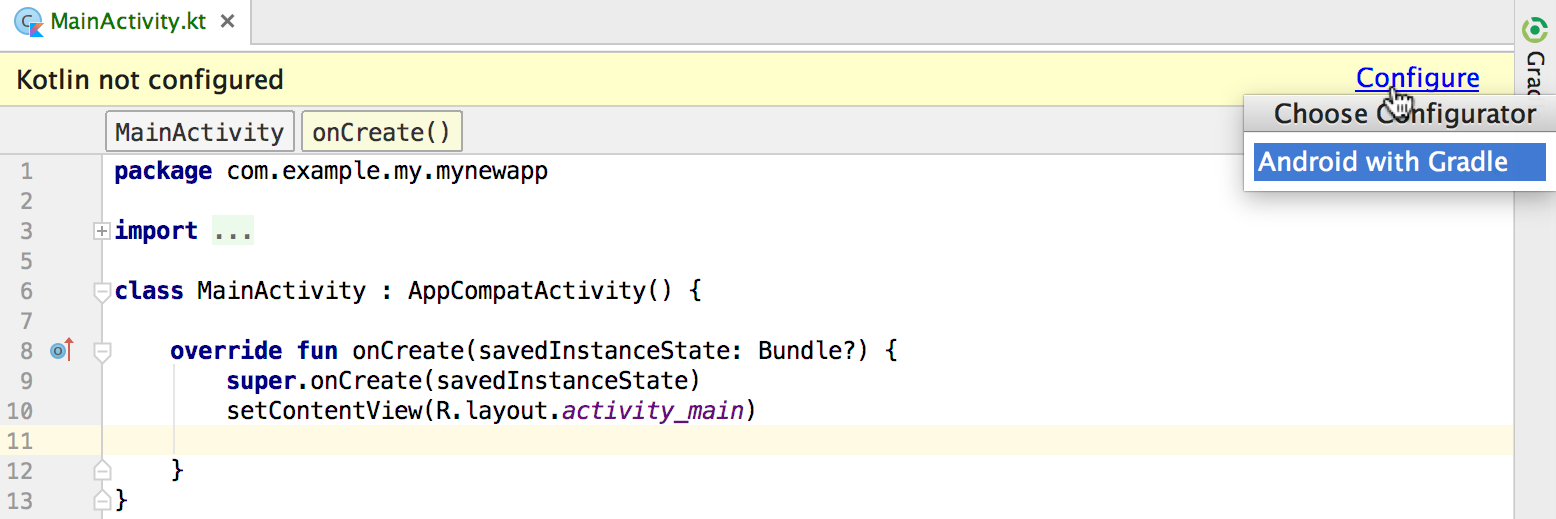
Alternatively, you can call this option via the *Code | Convert Java File to Kotlin File* menu entry or by using the corresponding shortcut (you can find it at the menu entry).



[Step 5 - After the conversion you should have an activity written in Kotlin.](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-5---select-individual-components-optional)

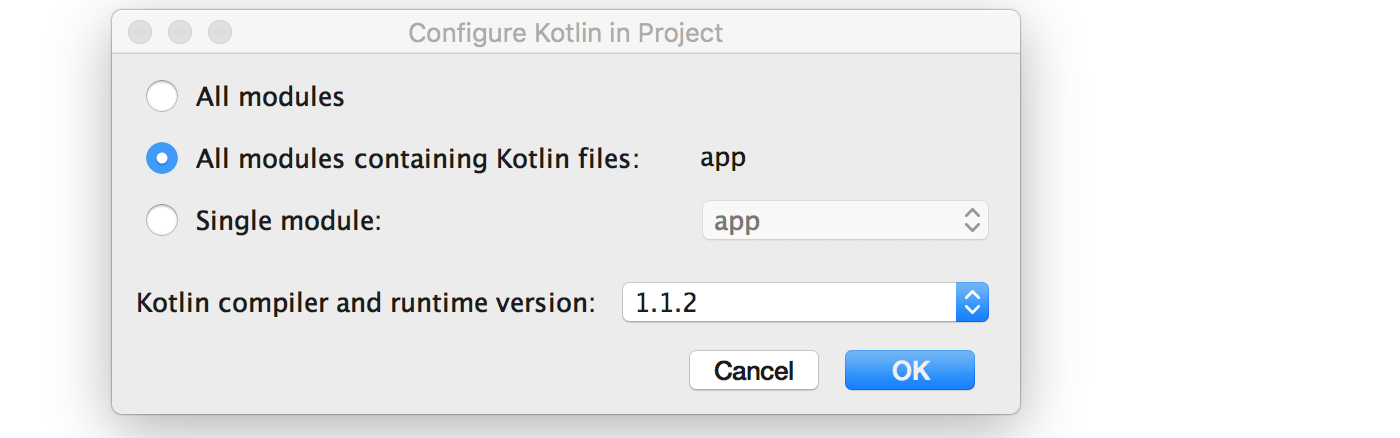
[Step 6 -](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-6---install-language-packs-optional) Configuring Kotlin in the project

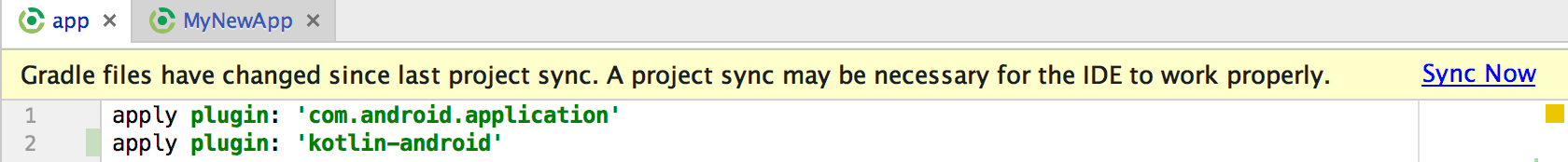
If you start editing this file, Android Studio shows you a prompt that Kotlin is not configured, so you can configure it. Alternatively, you can invoke the configuration by selecting Tools | Kotlin | Configure Kotlin in Project from the main menu.

Alternatively, you can invoke the configuration by selecting Tools | Kotlin | Configure Kotlin in Project from the main menu.

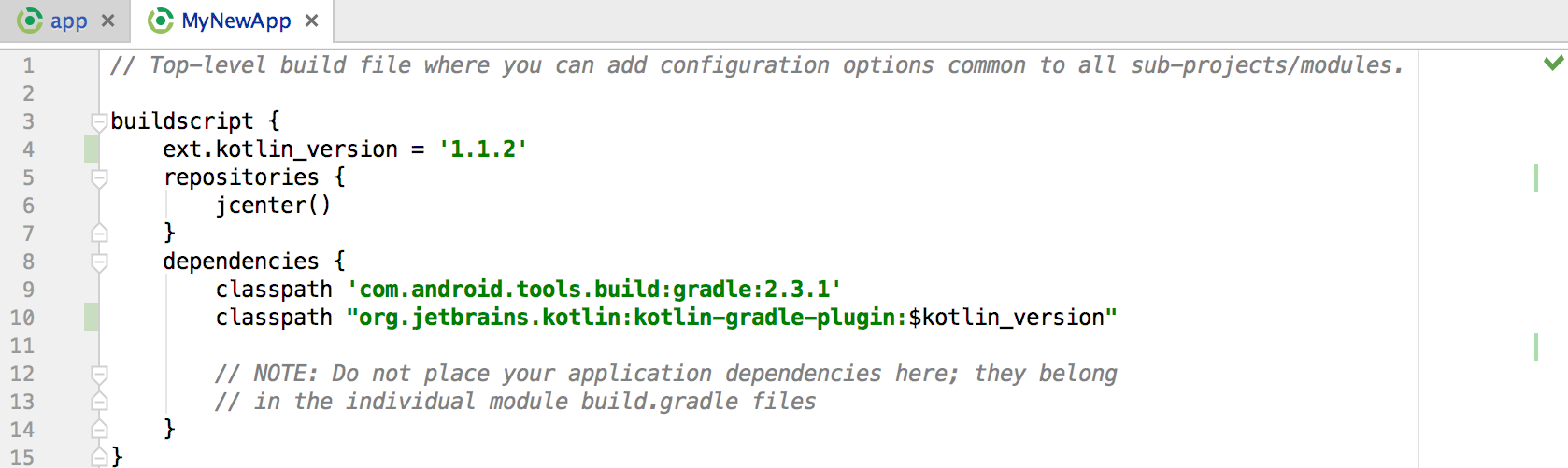
You are then prompted for the version of Kotlin.

Step 7- Choose the latest available from the list of installed versions.



Step 8- After you configure Kotlin, build.gradle file for the application should be updated. Now you can see that *apply plugin: 'kotlin-android'* and the kotlin-stdlib dependency were added.

Step 9- The last thing to do is to sync the project. You can press 'Sync Now' in a prompt or invoke an action **Sync Project with Gradle Files**.



Step 10- Building and publishing the Kotlin application for Android

**Lab Task 4:**

Following are the installation steps for Visual Studio:

[Step 1 - Make sure your computer is ready for Visual Studio](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017" \l "step-1---make-sure-your-computer-is-ready-for-visual-studio)

Before you begin installing Visual Studio:

1. Check the [system requirements](https://docs.microsoft.com/en-us/visualstudio/productinfo/vs2017-system-requirements-vs). These requirements help you know whether your computer supports Visual Studio 2017.
2. Apply the latest Windows updates. These updates ensure that your computer has both the latest security updates and the required system components for Visual Studio.
3. Reboot. The reboot ensures that any pending installs or updates don't hinder the Visual Studio install.
4. Free up space. Remove unneeded files and applications from your %SystemDrive% by, for example, running the Disk Cleanup app.

[Step 2 - Download Visual Studio](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-2---download-visual-studio)

Next, download the Visual Studio bootstrapper file. To do so, click the following button, select the edition of Visual Studio 2017 that you want, click **Save**, and then click **Open folder**.

[**Download Visual Studio 2017**](https://visualstudio.microsoft.com/downloads/?utm_medium=microsoft&utm_source=docs.microsoft.com&utm_campaign=button+cta&utm_content=download+vs2017)

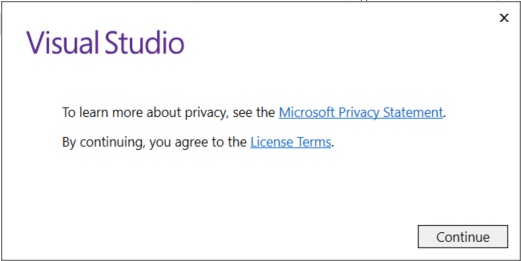
[Step 3 - Install the Visual Studio installer](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-3---install-the-visual-studio-installer)

Then, run the bootstrapper file to install the Visual Studio Installer. This new lightweight installer includes everything you need to both install and customize Visual Studio 2017.

1. From your **Downloads** folder, double-click the bootstrapper that matches or is similar to one of the following files:
   * **vs\_enterprise.exe** for Visual Studio Enterprise
   * **vs\_professional.exe** for Visual Studio Professional
   * **vs\_community.exe** for Visual Studio Community

If you receive a User Account Control notice, click **Yes**.

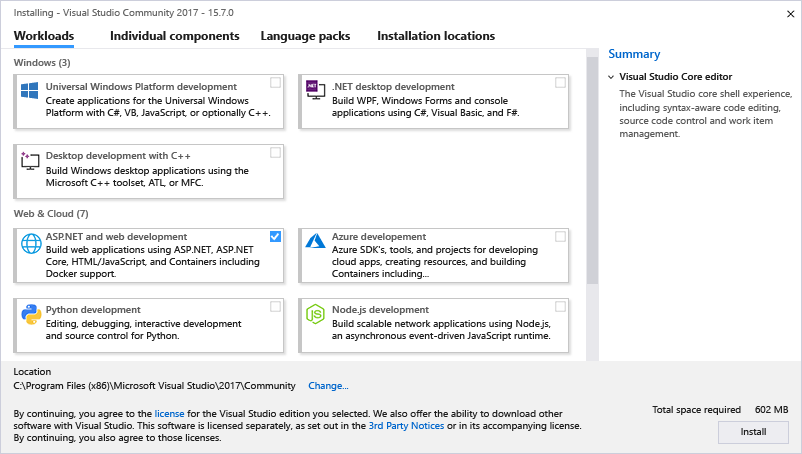
1. We'll ask you to acknowledge the Microsoft [License Terms](https://visualstudio.microsoft.com/license-terms/) and the Microsoft [Privacy Statement](https://go.microsoft.com/fwlink/?LinkID=824704). Click **Continue**.



[Step 4 - Select workloads](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-4---select-workloads)

After the installer is installed, you can use it to customize your installation by selecting the feature sets—or workloads—that you want. Here's how.

1. Find the workload you want in the **Installing Visual Studio** screen.



For example, choose the ".NET desktop development" workload. It comes with the default core editor, which includes basic code editing support for over 20 languages, the ability to open and edit code from any folder without requiring a project, and integrated source code control.

1. After you select the workload(s) you want, click **Install**.

Next, status screens appear that show the progress of your Visual Studio installation.

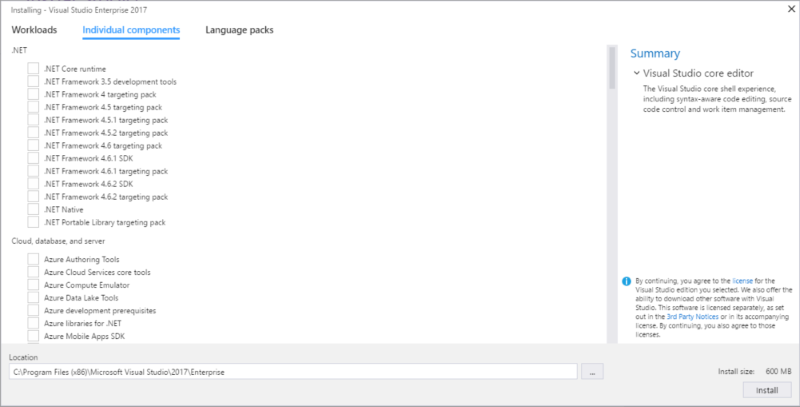
1. After the new workloads and components are installed, click **Launch**.

**Note:**

At any time after installation, you can install workloads or components that you didn't install initially. If you have Visual Studio open, go to **Tools** > **Get Tools and Features...** which opens the Visual Studio Installer. Or, open **Visual Studio Installer** from the Start menu. From there, you can select the workloads or components that you wish to install, then click **Modify**.

[Step 5 - Select individual components (Optional)](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-5---select-individual-components-optional)

If you don't want to use the Workloads feature to customize your Visual Studio installation, you can do so by installing individual components instead. To select individual components, click the **Individual components** option from the Visual Studio Installer, select what you want, and then follow the prompts.



[Step 6 - Install language packs (Optional)](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-6---install-language-packs-optional)

[Step 7 - Change the installation location (Optional)](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-7---change-the-installation-location-optional)

[Step 8 - Start developing](https://docs.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2017#step-8---start-developing)

After Visual Studio installation is complete, click the **Launch** button to [get started developing with Visual Studio](https://docs.microsoft.com/en-us/visualstudio/ide/get-started-developing-with-visual-studio?view=vs-2017).

1. Click **File**, and then click **New Project**.
2. Select a project type.

For example, to [build a C++ app](https://docs.microsoft.com/en-us/visualstudio/ide/getting-started-with-cpp-in-visual-studio?view=vs-2017), click **Installed**, expand **Visual C++**, and then select the C++ project type that you want to build.   
  
To [build a C# app](https://docs.microsoft.com/en-us/visualstudio/ide/walkthrough-create-a-simple-application-with-visual-csharp-or-visual-basic?view=vs-2017), click **Installed**, expand **Visual C#**, and then select the C# project type that you want to build.

**Lab Task 5:**

Following are the installation steps for Python:

You should download and install the latest version of Python. The current latest (as of Fall 2018) is Python 3.6.4. or Python 3.7.0.

The Python download requires about 30 Mb of disk space; keep it on your machine, in case you need to re-install Python. When installed, Python requires about an additional 90 Mb of disk space.

**Step 1: Downloading:**

1. Goto @ <https://www.python.org/downloads/>
2. Click the Download Python 3.6.2 button (or the latest version available).
3. The file named python-3.6.2.exe should start downloading into your standard download folder. This file is about 30 Mb so it might take a while to download fully if you are on a slow internet connection (it took me about 10 seconds over a cable modem).
4. The file should appear as

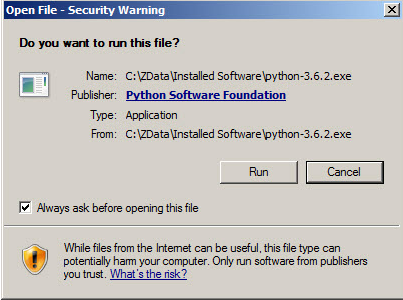
https://www.ics.uci.edu/~pattis/common/handouts/pythoneclipsejava/images/python/exefile.jpg

1. Move this file to a more permanent location, so that you can install Python (and reinstall it easily later, if necessary).
2. Feel free to explore this webpage further; if you want to just continue the installation, you can terminate the tab browsing this webpage.
3. Start the **Installing** instructions directly below.

**Step 2: Installing:**

1. Double-click the icon labeling the file python-3.6.2.exe.

An Open File - Security Warning pop-up window will appear.



1. Click **Run**.

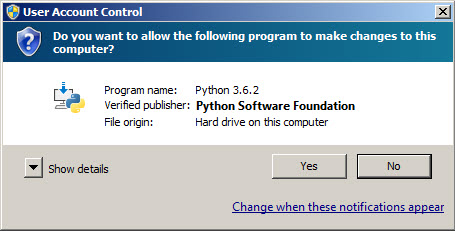
A **Python 3.6.2 (32-bit) Setup** pop-up window will appear.

****

1. Ensure that the **Install launcher for all users (recommended)** and the **Add Python 3.6 to PATH** checkboxes at the bottom are checked.

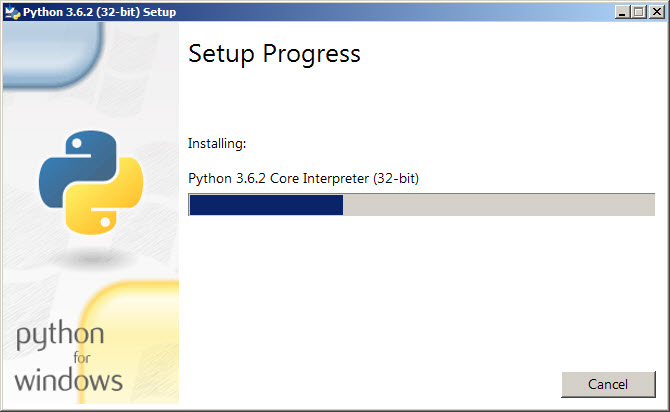
If the Python Installer finds an earlier version of Python installed on your computer, the **Install Now** message will instead appear as **Upgrade Now** (and the checkboxes will not appear).

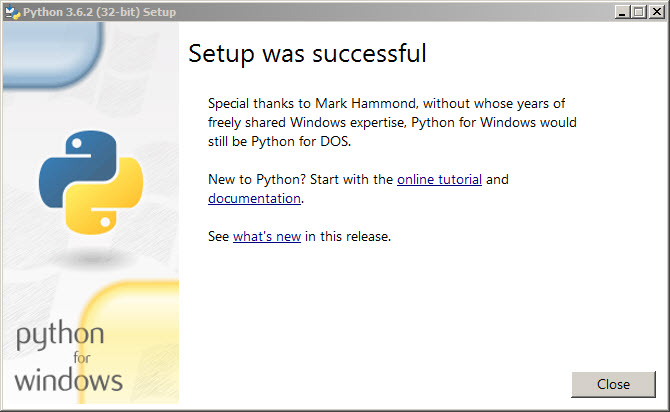
1. Highlight the **Install Now** (or **Upgrade Now**) message, and then click it.

****A **User Account Conrol** pop-up window will appear, posing the question **Do you want the allow the following program to make changes to this computer?**

1. Click the **Yes** button.

A new **Python 3.6.2 (32-bit) Setup** pop-up window will appear with a **Setup Progress** message and a progress bar.



During installation, it will show the various components it is installing and move the progress bar towards completion. Soon, a new **Python 3.6.2 (32-bit) Setup** pop-up window will appear with a **Setup was successfuly** message.

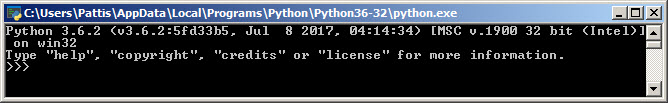
1. Click the **Close** button.

Python should now be installed.

### Step 3: Verifying

To try to verify installation,

1. Navigate to the directory **C:\Users\Pattis\AppData\Local\Programs\Python\Python36-32** (or to whatever directory Python was installed: see the pop-up window for Installing step 3).
2. Double-click the icon/file **python.exe**.

The following pop-up window will appear.

A pop-up window with the title **C:\Users\Pattis\AppData\Local\Programs\Python\Python36-32** appears, and inside the window; on the first line is the text **Python 3.6.2 ...** (notice that it should also say 32 bit). Inside the window, at the bottom left, is the prompt **>>>**: type **exit()** to this prompt and press **enter** to terminate Python.

**Lab Task**

1. Install JDK if already installed, uninstall the older and install new version.
2. Install Eclipse, after JDK installation.
3. Install Kotlin, in Android Studio as a plugin.
4. Install latest version of Visual Studio. (You can install an earlier version as well because this is a heavy software that slow down your system, replacement can be 2012 or 2015 version)
5. Install Python latest version.
6. [Optional] Run simple programs to check all the software are working properly.

For help use

* <https://www3.ntu.edu.sg/home/ehchua/programming/howto/JDK_Howto.html>
* http://www.oracle.com/technetwork/java/javase/documentation/index.html

## Deliverables

* Each submission is individual with the following composition:
  + Documentation (Comments)
* Convert your submission file into a zip folder and name it as given below, finally upload the zip folder to LMS.
  + - Name – Registration No. – Section

## Grade Criteria

This lab is graded. Min marks: 0. Max marks: 10.

|  |  |  |
| --- | --- | --- |
| **Activity** | **Minimum** | **Maximum** |
| Task 1 | 0 | 2 |
| Task 2 | 0 | 2 |
| Task 3 | 0 | 2 |
| Task 4 | 0 | 2 |
| Task 5 | 0 | 2 |
| Task 6 (Bonus) | 0 | 2 |
| **Total** | **0** | **10** |