# ­I:Users:a540788:Pictures:Java Tewksbury.pngI:Users:a540788:Downloads:java.png Introduction

## Students

1. Sign-up sheet
2. Tell us about you!
3. Share with us your motivation in attending this class!
4. Have you ever worked as a developer?
5. Tell us your expectations for this class?

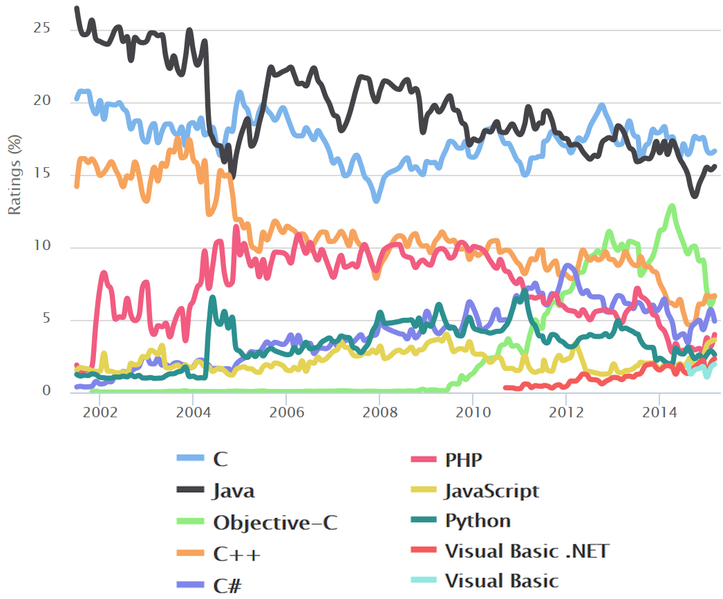
## Instructor

1. Who is the instructor?
2. Why teach a class on Java?
3. What are the expectations of the instructor for this class?
4. What is the format of this class?
5. My email: [giuliano.amantini@gmail.com](mailto:giuliano.amantini@gmail.com), when getting in touch with me use this title: *Tewksbury Library Java 1*
6. Next class is September 21 2015

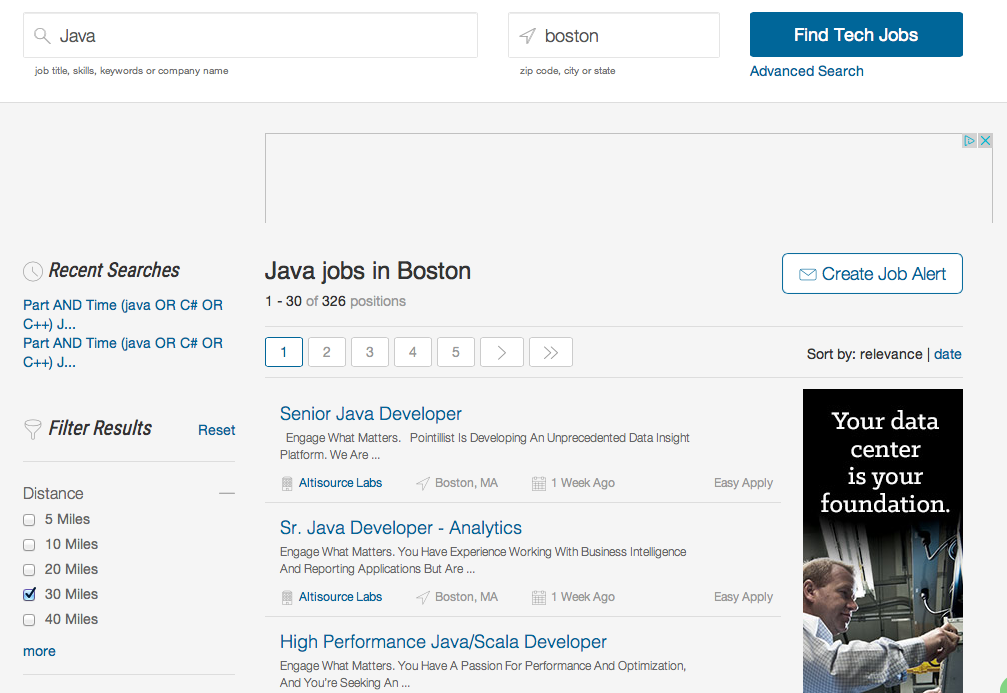
# I:Users:a540788:Downloads:java.png What is Java and why would I use it?

## Why would I use it?

1. It is free!
2. One of most popular programming languages, as TIOBE index reports below:



1. when I went on [www.dice.com](http://www.dice.com) on September 9 2015, and searched for Java, there were 326 positions open in a radius of 30 miles around Boston, as the figure below shows:



## What is it?

Java is a general-purpose programming language that is:

1. Concurrent:
   1. time-slicing, single or multiprocessor machine
   2. parallel computing, only on multiprocessor machines
2. Class-based, Object-oriented:
   1. class is prototype, a blueprint
   2. object is an instance of class
   3. What is inheritance? Helps create a contract with users of classes
   4. What is encapsulation? Helps create extensions to classes
3. WORA – write once run anywhere through compiled code running on JVM

## What is the JVM?

The JVM is:

1. An abstract computer, the specification of the JVM is platform specific
2. One of the organizational units of the JVM is a class
   1. *.class* files must be recognized by the JVM, they are a type of binary format
   2. The JVM knows nothing of the Java programming language, only knows *.class* files
   3. A *.class* file contains JVM instructions, called bytecodes
   4. *.class* is platform-independent
3. The JVM operates on two kinds of types:
   1. Primitive types: int, double, …
   2. Reference types, pointers to objects

## Java Certification

Oracle offers certification programs, which can be found at:

<http://education.oracle.com/pls/web_prod-plq-dad/db_pages.getpage?page_id=458&p_org_id=1001&lang=US&get_params=p_track_id:JSE7Prog>

# Verifying and Installing the Java Development Kit



There are two types of installation:

1. JRE, this is a runtime environment, to check for the existence and version of this open a command window in Windows, or a terminal on a Mac and type:
   1. Java –version
   2. If the command returns the version of Java currently installed on the machine, that is a good sign otherwise one needs to install the JRE
   3. Go to website <http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html>
   4. Download the specific version for your operating system
2. JDK, this is a runtime + development environment, ships with a compiler (javac) to check for the existence and version of this open a command window in Windows, or a terminal on a Mac and type:
   1. Javac –version
   2. If the command returns the version of Java currently installed on the machine, that is a good sign otherwise one needs to install the JRE
   3. Go to website: <http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>
   4. Download the specific version for your operating system
   5. In Windows add the path of /bin in the JDK install to the PATH environment variable: as a way of example for JDK version 1.8.0\_60, PATH should contain C:/Program Files/Java/jdk1.8.0\_60/bin

# I:Users:a540788:Downloads:java.png Obtaining and Installing the Eclipse IDE

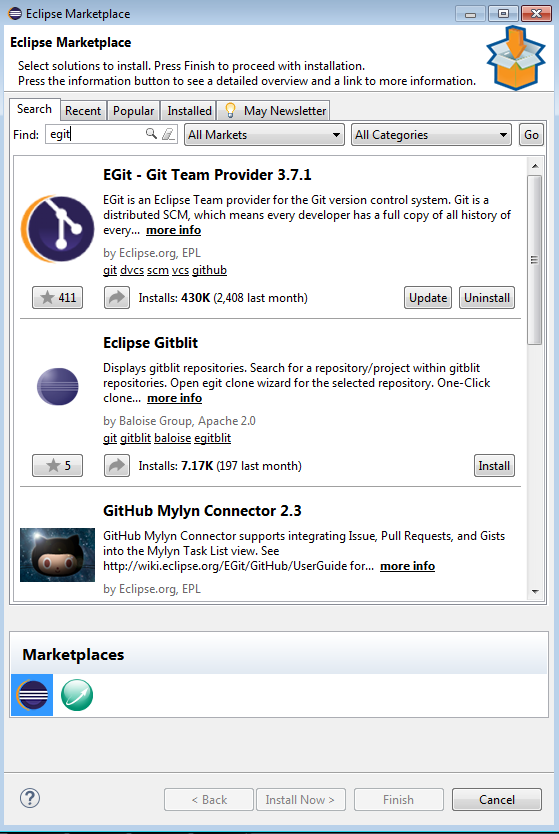
## The Eclipse IDE

This is one of the most popular IDEs for Java development.

1. Go to <https://eclipse.org/>
2. Download and run the installer for eclipse

## Installing EGit in Eclipse

Go to Help in Eclipse, and then select Eclipse marketplace. Then type “egit” in the window just like figure below:



Click on the egit update (or probably install, if this is the first time you install egit).

## What is Version Control?

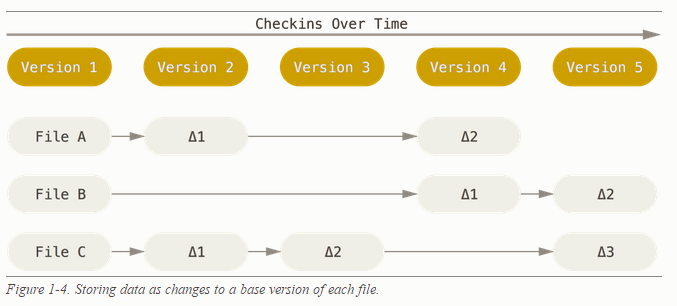
In computer software engineering, revision control is any kind of practice that tracks and provides control over changes to source code. Its usefulness:

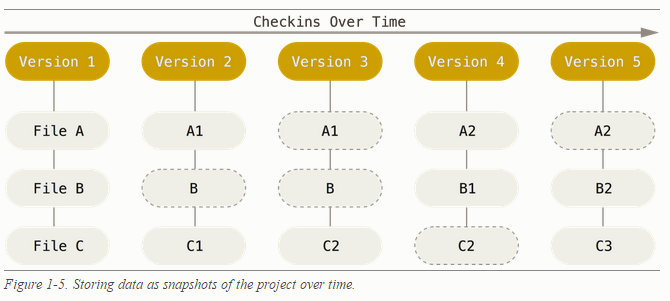
1. *Manage the process by which the control of items passes from one person to another*. Source control providers support both shared and exclusive file access. If access to project files is exclusive, the source control provider allows only one user at a time to check files out and modify them. If access is shared, more than one user can check out the script file, and the source control provider provides a mechanism for merging the versions as they are checked in.
2. *Archive successive versions of source-controlled items.* A source control provider stores the data that distinguishes one version of a source-controlled item from another. Consequently, you can retrieve any version of a source-controlled item. You can also designate any version to be the latest version of that item.
3. *Maintain detailed historical and version information on source-controlled items.* Source control stores the date and time on which the item was created, when it was checked out or checked in, and the user who performed the action.
4. *Collaborate across projects*. File sharing makes it possible for multiple projects to share source-controlled items. Changes to a shared item are reflected in all the projects that share the item.
5. *Automate* frequently repeated source control operations. A source control provider may define an interface from the command prompt that supports the key features of source control. You can use this interface in batch files to automate the source control tasks that you perform regularly.
6. *Recover from accidental deletions*. You can restore the latest file version checked into source control.
7. *Conserve disk space on both the source control client and server*. Some source control providers, such as Microsoft Visual SourceSafe, support disk space conservation on the server by storing the latest version of a file and the differences between each version and the version that precedes or follows it. On the client, Visual SourceSafe supports disk space conservation. You can cloak folders and files so that they are not downloaded to your local disk.

## What is Git

### Snapshots, Not Differences

The major difference between Git and any other VCS (Subversion and friends included) is the way Git thinks about its data. Conceptually, most other systems store information as a list of file-based changes. These systems (CVS, Subversion, Perforce, Bazaar, and so on) think of the information they keep as a set of files and the changes made to each file over time.

Git doesn’t think of or store its data this way. Instead, Git thinks of its data more like a set of snapshots of a miniature filesystem. Every time you commit, or save the state of your project in Git, it basically takes a picture of what all your files look like at that moment and stores a reference to that snapshot. To be efficient, if files have not changed, Git doesn’t store the file again, just a link to the previous identical file it has already stored. Git thinks about its data more like a stream of snapshots.

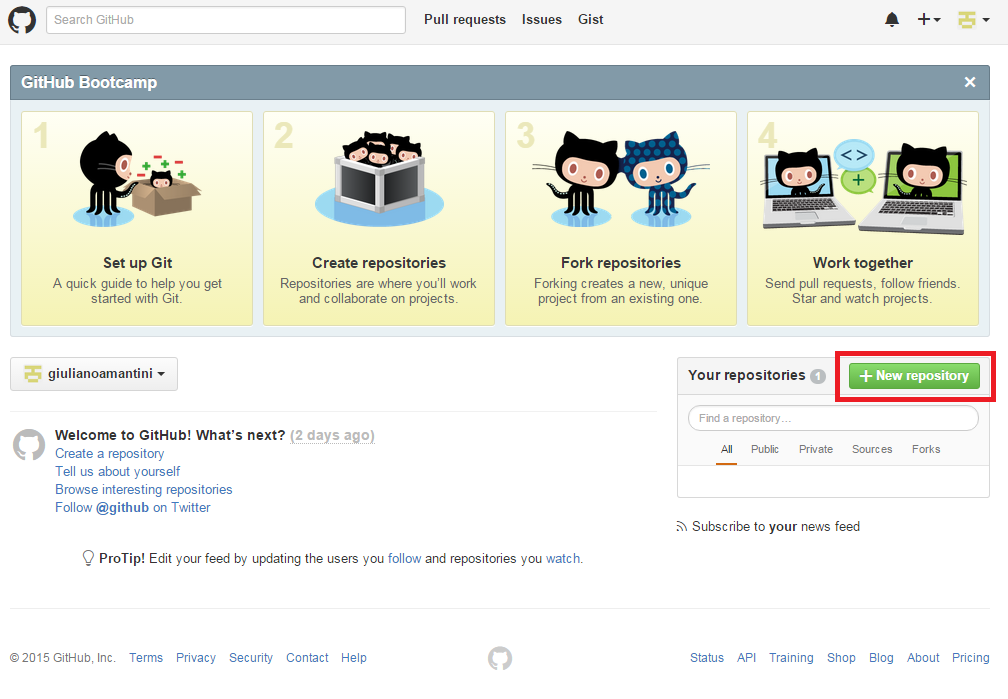
This is an important distinction between Git and nearly all other VCSs. It makes Git reconsider almost every aspect of version control that most other systems copied from the previous generation. This makes Git more like a mini filesystem with some incredibly powerful tools built on top of it, rather than simply a VCS. We’ll explore some of the benefits you gain by thinking of your data this way when we cover Git branching in Git Branching.

## What is Github?

GitHub is a Web-based Git repository hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.

## Installing GitHub

1. Go to: <https://github.com/> and create a free account with GitHub.
2. This will allow you to create only public repositories
3. Log into your account and create a new repository called hello-world, and add a brief description, and also add the README.md file.



1. This will create a page which provides you with an HTTPS address ending in .git.
2. Go to Eclipse and click on the following sequence
   1. Window
   2. Perspective
   3. Open perspective
   4. Other
   5. Git
3. The previous series of clicks should have opened the Git Perspective, now click on “clone a Git repository”
4. Copy the URL from github, this should be an address that ends in .git, and also add your credentials

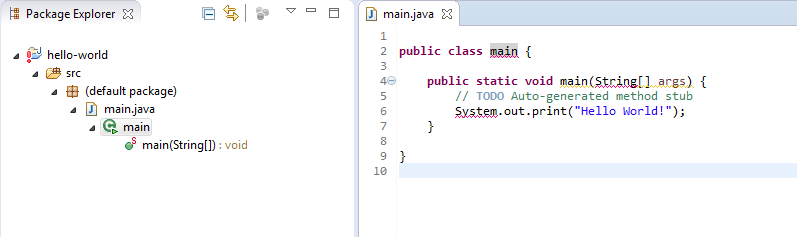
# Hello World



This is the first Java program that we are going to write. Follow the steps below:

## Writing the program

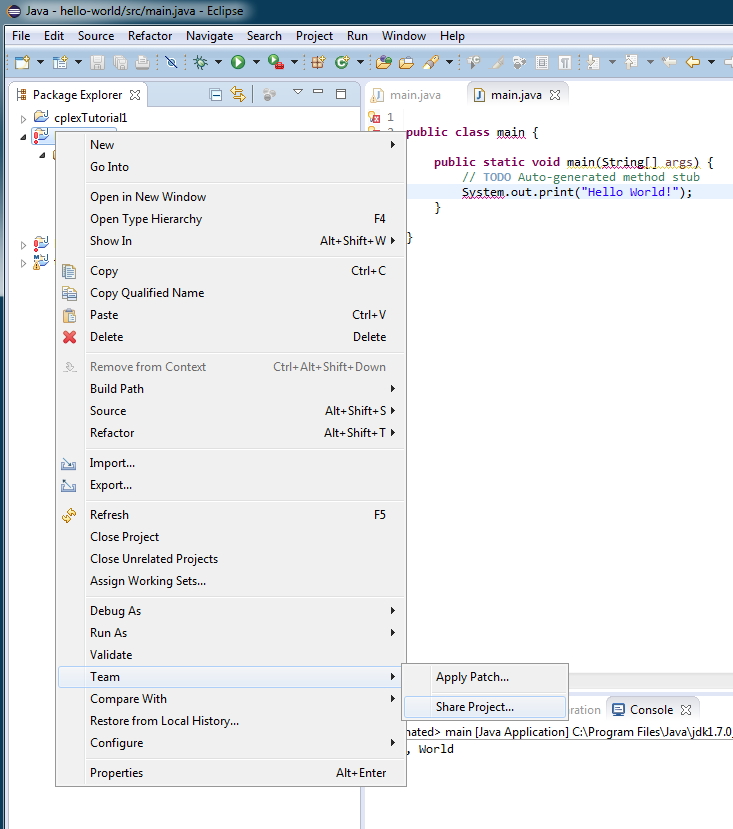
1. Make sure you are in the Java perspective in the Eclipse IDE
2. In the package explorer, right click on it and select New -> Java Project
3. In the window that pops up write hello-world in the project name, and then click next and then click finish
4. Click on the newly created hello world package and create a new class
5. Enter main in the name of the class and select public static void main
6. In the main method type *System.out.print(“Hello World!”);* and execute
7. The folder structure and the main method should look as in the figure below:



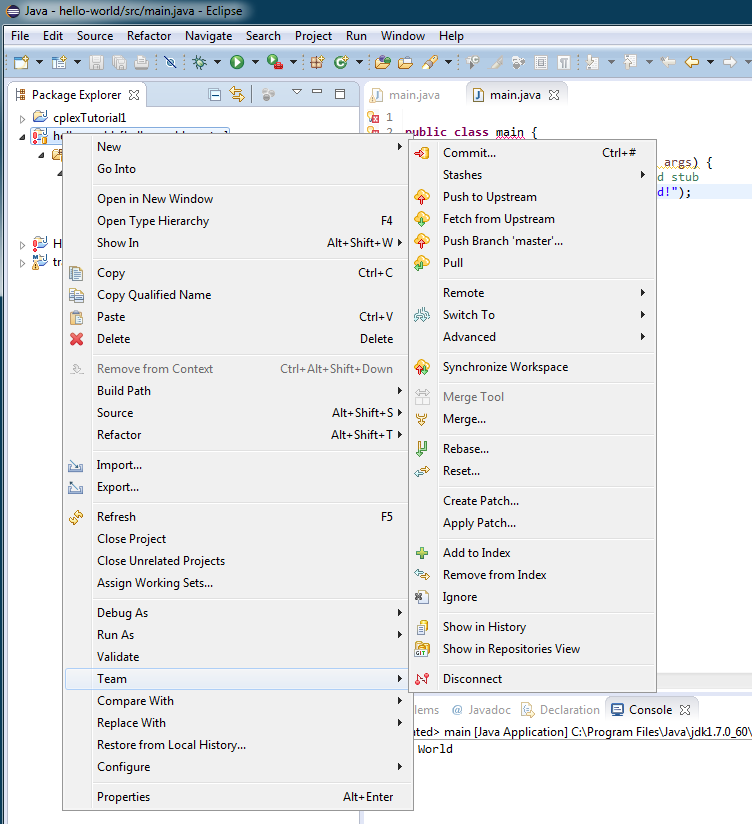
## Checking it into Github

In the Eclipse IDE:

1. Select the project hello-world
2. Right click on it and select Team-> Share Project



1. Select the repository that we loaded earlier, and click finish
2. Select the project hello-world again
3. Right click on it and select Team-> Commit, as below



1. Select the .*classpath* file, the .*project*, and *main.java* and then select Commit and Push.