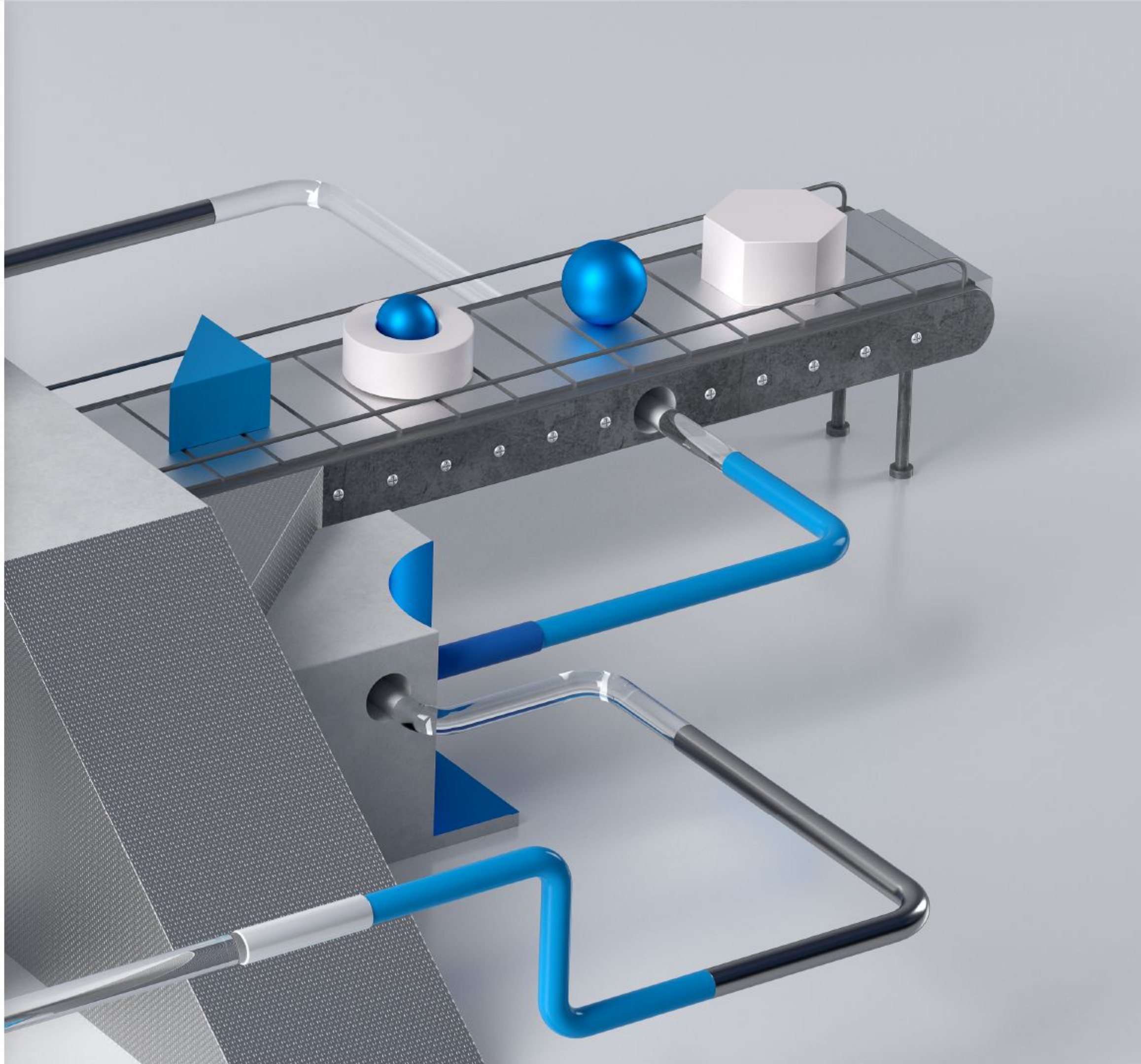
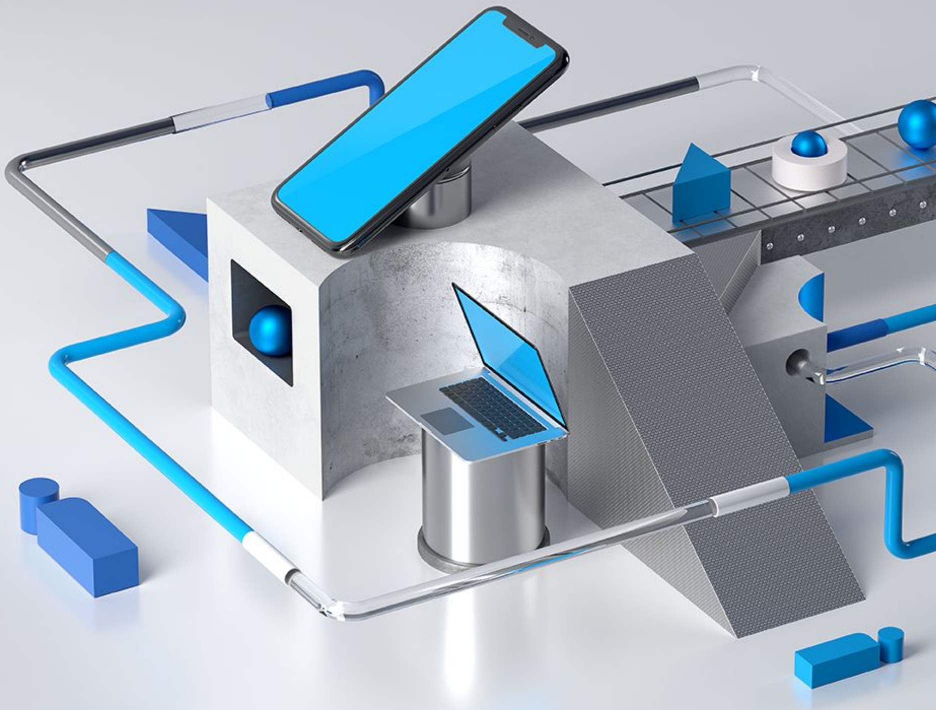
TDD — And Why You Should Bother

November 15th, 2019

Fiix



Who Are We?



Let’s Talk About Software

- Software can be thought of as a collection of services
- A service is a collection of features combined to achieve a purpose
- A feature is a collection of functions combined to achieve a goal
- A function is the smallest meaningful division of software

Ex: The File Menu in Google Sheets

- Contains the following features:
  - Share
    - To where?
  - New:
    - Different file types of the Google Suite
  - Open
    - Open what?
  - Import Slides
  - Make a Copy
  - Rename
  - etc.

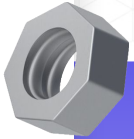
What is a Test?

A test is a **repeatable** activity to **assert** the validity of an **expectation**.

Manual Testing
(by yourself)

Outsourced Testing
(quality assurance)

Automated Testing
(tests as code)



Types of Testing

- Unit Testing
- Integration Testing
- End to End Testing
- Stress Testing
- Smoke Testing

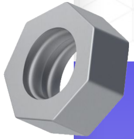
Test-Driven Development

- A software development process
- Tests are written **first**
  - Traditionally focused primarily on unit tests
  - Tests must be **simple**
- Uses a method called “red - green - refactor”
  - Write a test, write the code to pass the test, clean up the code, repeat

Tests… First?

Thinking About Functionality

- Focusing on functionality avoids the pitfall of trying to test code paths
  - Your code paths should follow your functions, not the other way around
    - If you end up with code that doesn't have a test, ask yourself how and why it exists
- Tests first create reliability
  - You know your code works as you write it
- Makes future development safer
  - When you refactor code, you have immediate feedback of anything you may have broken



Things to Avoid

- Tests should not introduce state
  - Be stateless whenever possible
  - when state is inevitable or a requirement, clean up the state once the test is done
- Tests should not test implementation details
  - A test doesn’t know, nor care how its functionality is executed
  - A test only cares that a given input results in an expected outcome
- Tests should be independent
  - If the test relies on other tests, it can create false failures when issues occur in other tests

What is Git?

What is Version Control?

- Users can track changes to code
- Allows developers to view older versions of their code
- Infinitely useful for collaborative software development
- Git is an example of a version control system
- GitHub is one of many hosts for version control using Git

Setting Up GitHub

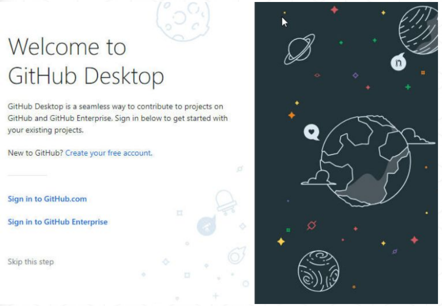
1. Create a GitHub account at: https://github.com/

2. After signing up, download the GitHub desktop client from:

   https://desktop.github.com/

3. Follow the on-screen installation instructions

4. Sign-in to your GitHub account on the desktop client



Welcome to
GitHub Desktop

GitHub Desktop is a seamless way to contribute to projects on GitHub and GitHub Enterprise. Sign in below to get started with your existing projects.

New to GitHub? Create your free account.

Sign in to GitHub.com

Sign in to GitHub Enterprise

Skip this step

Getting Ready

Setting Up Maven

Prerequisites:

1. Java 11
2. A Java IDE: Eclipse, IntelliJ, or NetBeans

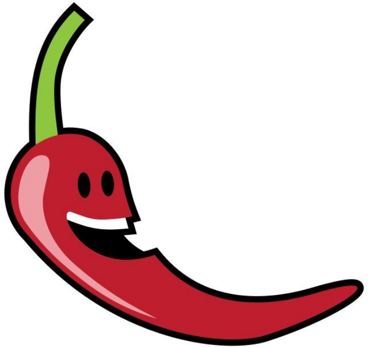
Steps:

Mac OS	Windows 10
1. Make sure you have Homebrew installed (see https://brew.sh/) 2. Open Terminal 3. Type: *brew install maven* 4. Restart your terminal 5. Type *mvn -v* to verify your installation	Follow the instructions available at: https://bit.ly/2XeNtMA

Setting Up Lombok

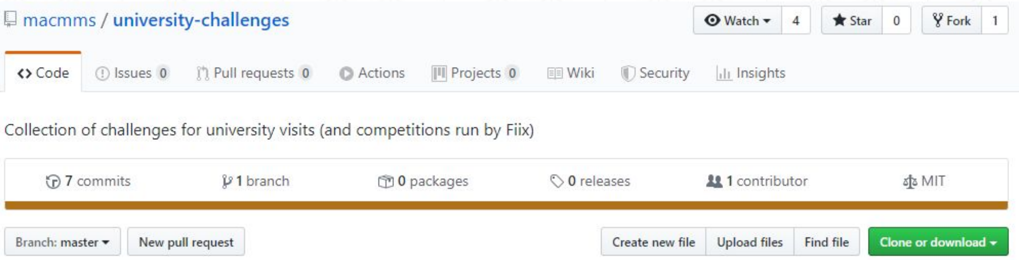
Lombok is a library for auto-generating code in Java!

1. In your **IDE** of choice, go to your **plugins** market
2. Search for “**Lombok**”
3. **Install** the plugin & restart your IDE
4. You’re good to go!



Downloading the Project

1. In your browser, navigate to https://github.com/macmms/university-challenges



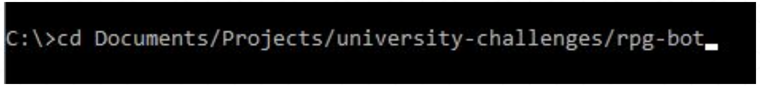
macmms / university-challenges

Collection of challenges for university visits (and competitions run by Fiix)

2. Click the Fork 1 button at the top right of the page
3. In the GitHub Desktop client, search for ‘*university-challenges*’
4. Clone this repository
5. Take note of the local path that your repository was cloned to

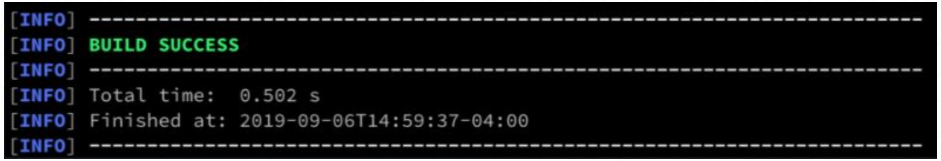
Compiling the Project

1. Open your Terminal (or Command Prompt)
2. Navigate to the path of you cloned the repository to using the *cd* command

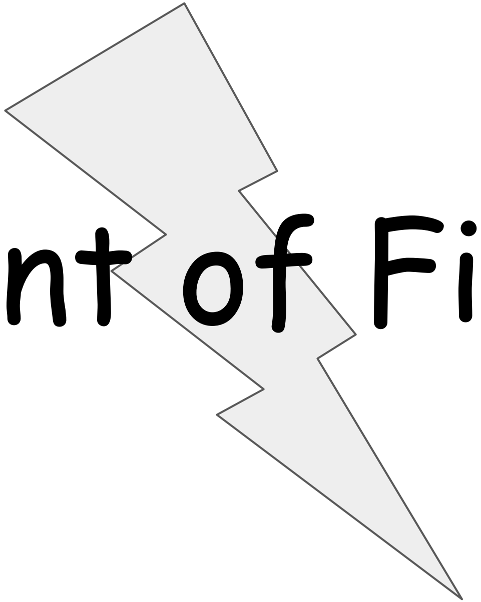


```
C:\>cd Documents/Projects/university-challenges/rpg-bot
```

3. Make sure you are in the *rpg-bot* directory
4. In your Terminal, run *mvn clean install*
5. Once it's finished, you should see a message similar to this:



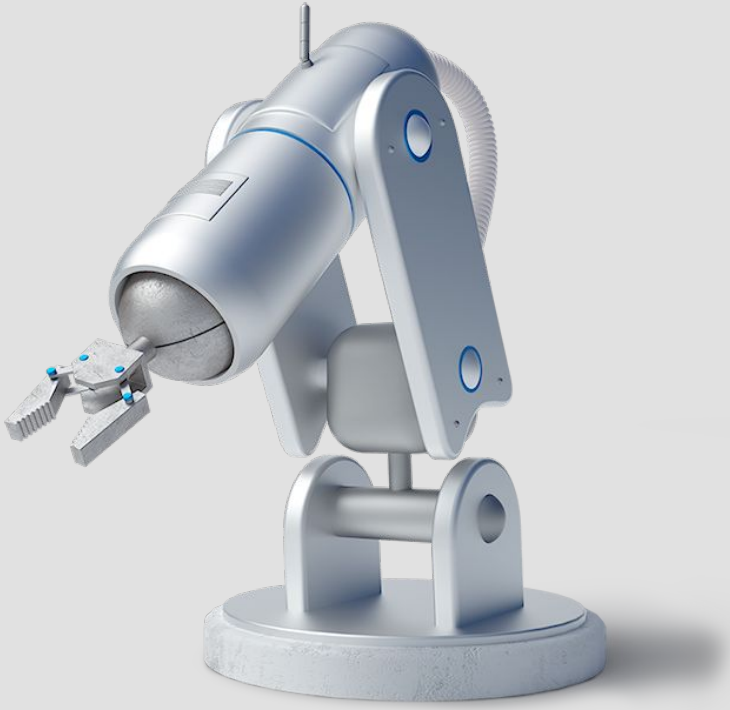
```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  0.502 s
[INFO] Finished at: 2019-09-06T14:59:37-04:00
[INFO] ------------------------------------------------------------------------
```



Figment of Fitecraft

Let the Games Begin!

Congratulations!
You’re a tried and
tested programmer.



Thank you!

35 Golden Ave Suite A-201
Toronto, ON, Canada
info@fiixsoftware.com
Local: +1 (647) 317-9055
Toll Free: +1 (855) 884-5619