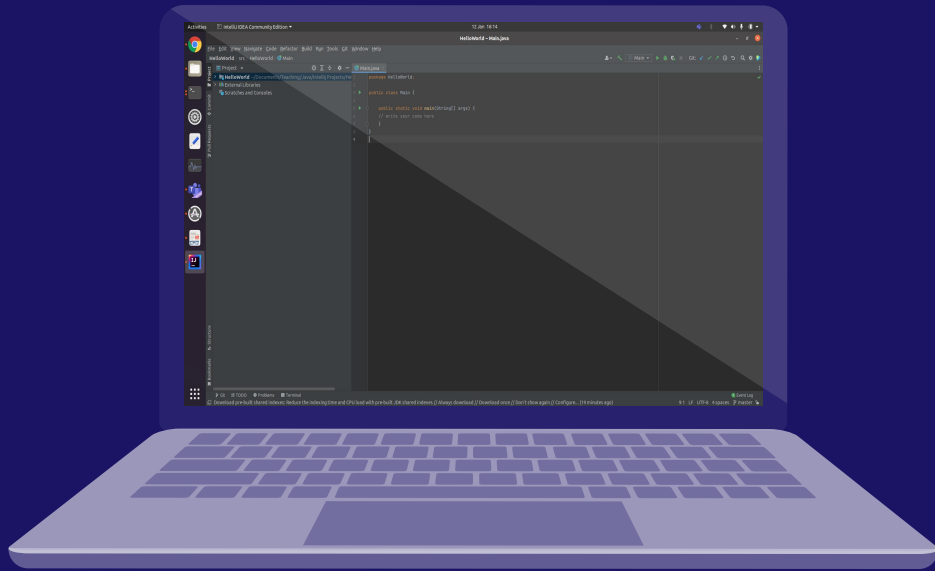# JAVA BASICS

A few things to get you started

INSTALL INTELLIJ

This session assumes you have IntelliJ installed on your machines following the guidance on BlackBoard. It is vital that you have followed all of the instructions and have the correct SDK installed to progress with this module.

USE INTELLIJ

IntelliJ is a powerful tool that can do a lot of work for you if you use it properly.

THIS SESSION

01

Java
Versions

02

Running
Projects

03

Method
Complexity

04

Other
Projects

PROJECTS

Unlike in C, you will be given projects with some source code provided. These will be maven projects, which have been chosen to standardise the process of marking by enabling compiling on the command line without relying on an IDE like IntelliJ.

01

JAVA VERSIONS

VERSION 11 OR 17?

You will have either 11 or 17 installed by this point, and can check by running 'java --version'. It doesn't matter what version you have, but you will need to change the pom.xml file if you have version 11. Version 17 has new features, although we don't use them in this course, this version has been used to accommodate the M1 macs.

POM.XML EDITING

```
<maven.compiler.source>11</maven.compiler.source>
<maven.compiler.target>11</maven.compiler.target>
```

These lines in the pom.xml file determine the version of Java you are using for the project. Change the version number to the version of Java you're using (must be >= 11).
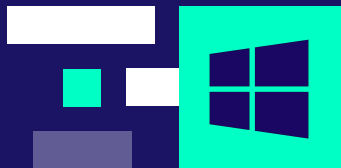
02

RUNNING PROJECTS

MAVEN COMPILE AND EXECUTE

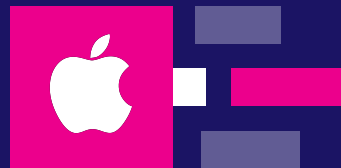I've been told that there's some discrepancy as to what instructions work for this...
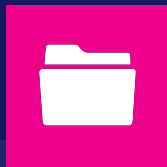
Windows

```
.\mvnw clean compile
.\mvnw exec:java
```

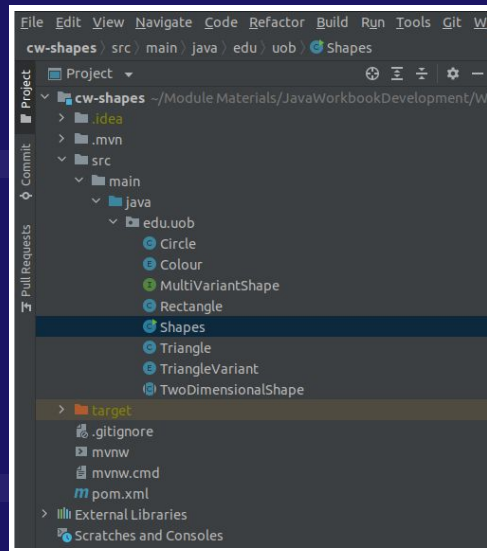## Mac

```
./mvnw clean compile
./mvnw exec:java
```

Linux

```
./mvnw clean compile
./mvnw exec:java
```
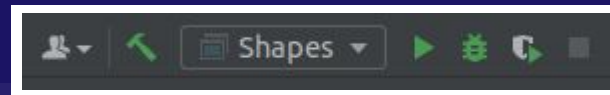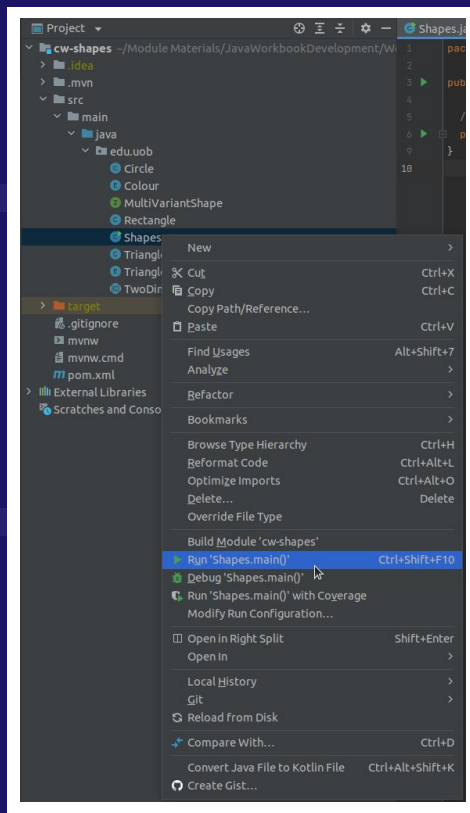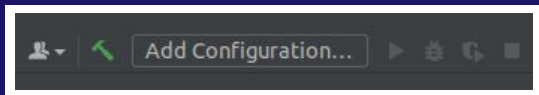
OPEN PROJECT INTELLIJ

Root folder

You **MUST** open the project folder in IntelliJ. This means either right clicking the cw-shapes folder and opening in intellij or starting intellij and choosing File->Open. This ensures all the files are correctly handled by IntelliJ.

RUN PROJECT INTELLIJ

IntelliJ does not start
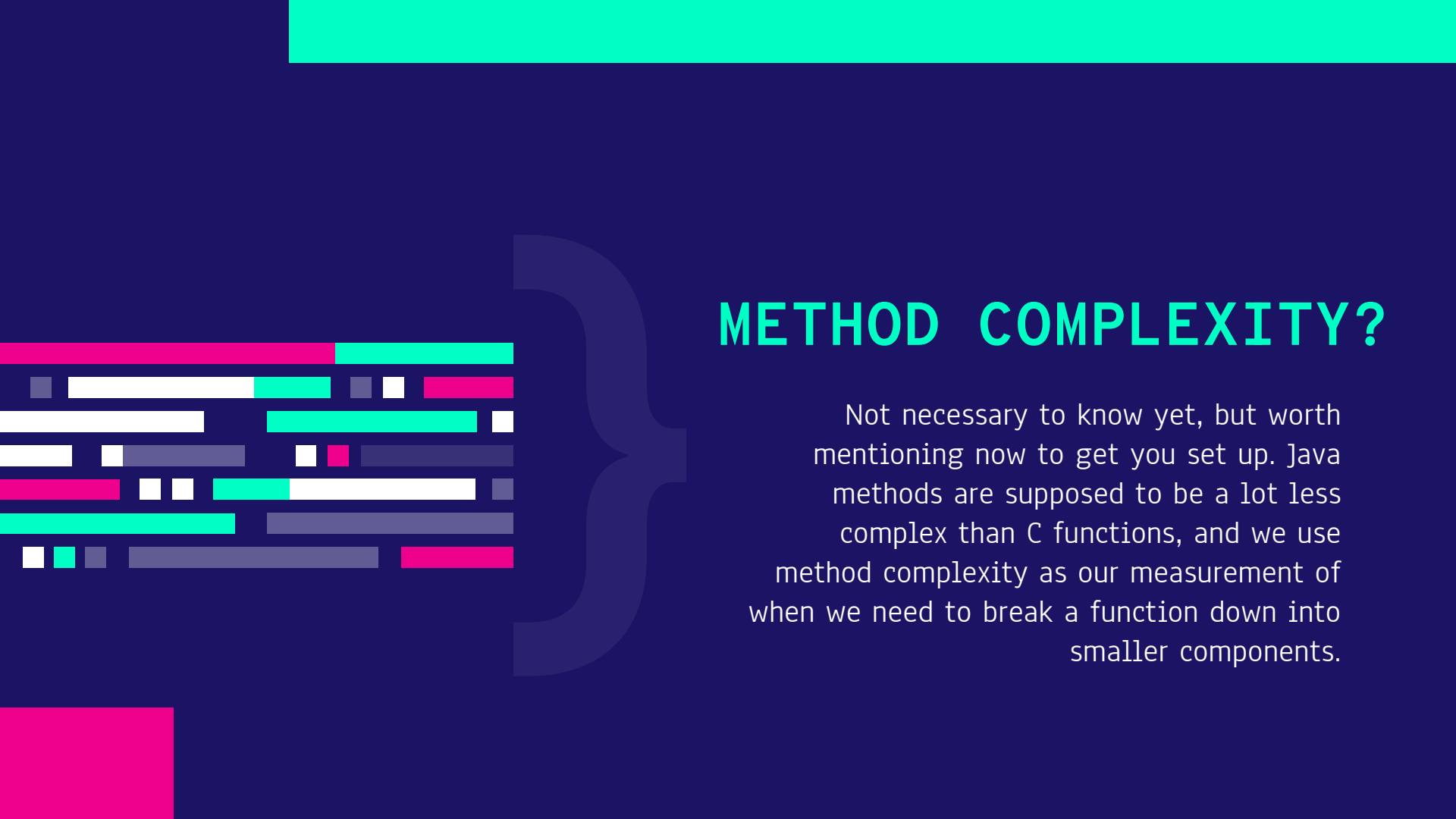with a configuration
setup automatically.

Once we run
Shapes.main() once,
the configuration is
set up.

03

METHOD COMPLEXITY

METHOD COMPLEXITY?

Not necessary to know yet, but worth mentioning now to get you set up. Java methods are supposed to be a lot less complex than C functions, and we use method complexity as our measurement of when we need to break a function down into smaller components.

CODEMETRICS PLUGIN

This useful plugin will show the complexity of each method and which components are contributing to the complexity.

INSTALL CODE METRICS INTELLIJ

CodeMetrics

Automatically shows how complex your methods are.

Go to marketplace

File->Settings->Plugins

Ensure you are in the Marketplace tab

Search

Type 'CodeMetrics' into the search bar

Install

Click install
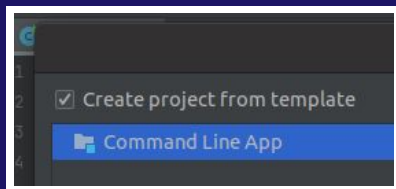
Restart

Restart IDE

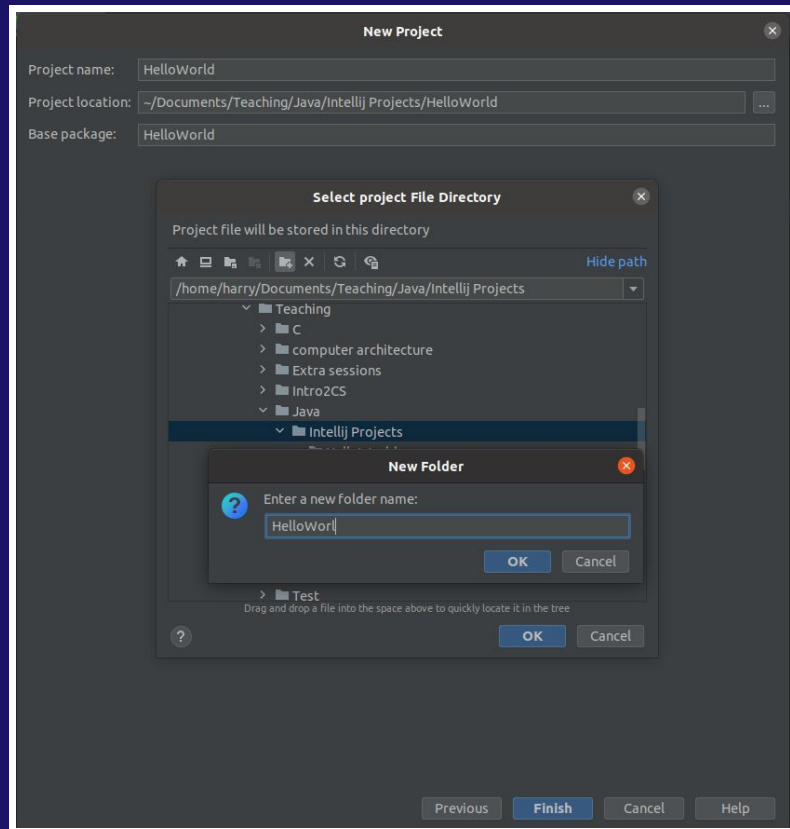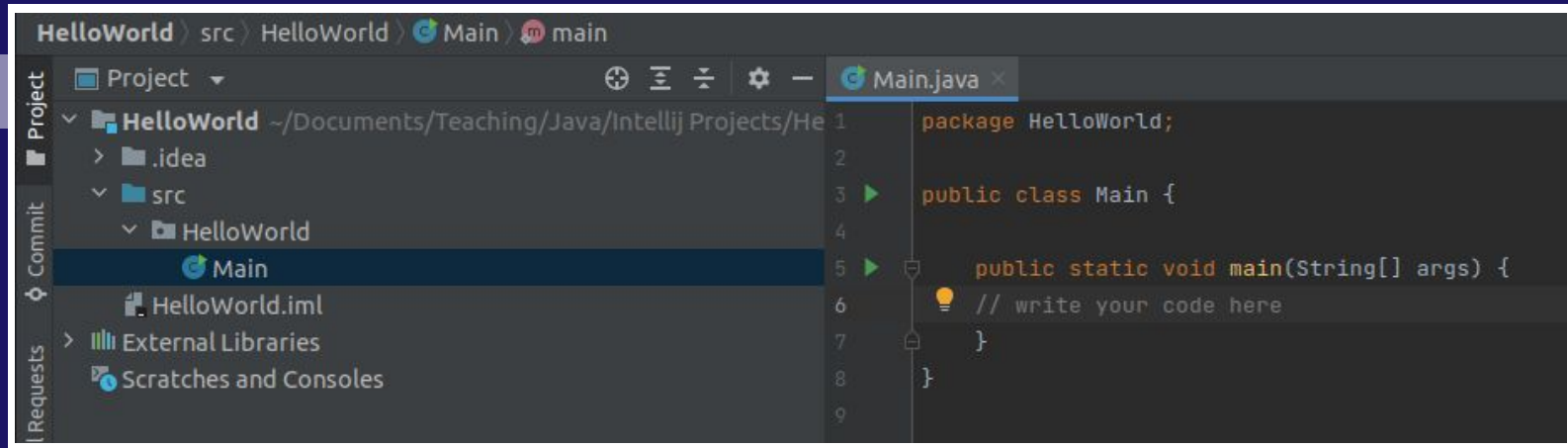# 04

OTHER PROJECTS

PROJECT SETUP

File -> New -> Project

We will want a class with a Main() method each time we create a project.

We create a new folder for each project.

RESULT

END