Massey University

159.251 Software Design and Construction

Tutorial 3

Software metrics and code analysis tools

Prerequisites (what you are expected to do before you start the tutorial)

1. Study the lecture material on software metrics
2. Get the Java project required for this tutorial from GitHub
3. Make sure git and maven are installed...
4. Before you start a task, install the tools required to complete the tasks (see the instructions below each task about the tool)

Tools Required

○ A software metrics tool	(Task 1)
○ PMD maven plugin	(Task 2)

Objectives

1. Get familiar with using software metrics tools to measure several aspects of a program.
2. Learn about other aspects of static analysis tools such as finding potential source code bugs in a program.

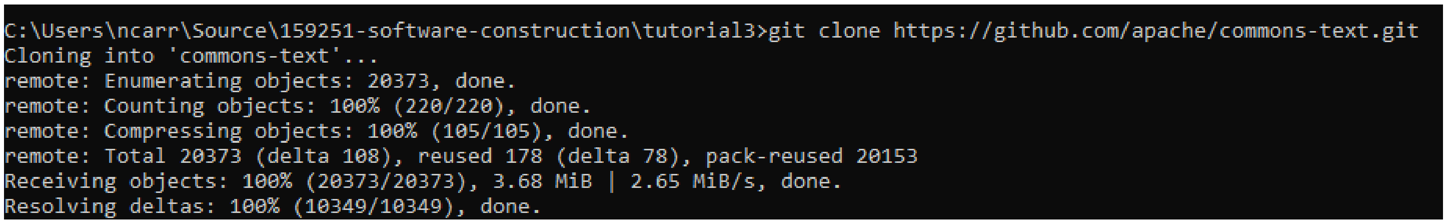
Deadlines and penalties

You must submit your final work using the stream submission system no later than 4 August at 11.59pm. Late submissions are accepted with penalties. The penalty is 20% deducted from the total possible mark for every day delay in submission (one day late – out of 80%, two days late then out of 60% … etc.).

Java Project Required to complete the tasks

Apache Commons Text: https://github.com/apache/commons-text

Apache Commons Text is a library that provides different algorithms to facilitate working on strings. The functionality of the tool is not important for this tutorial, as you will not need to work with the source code directly!



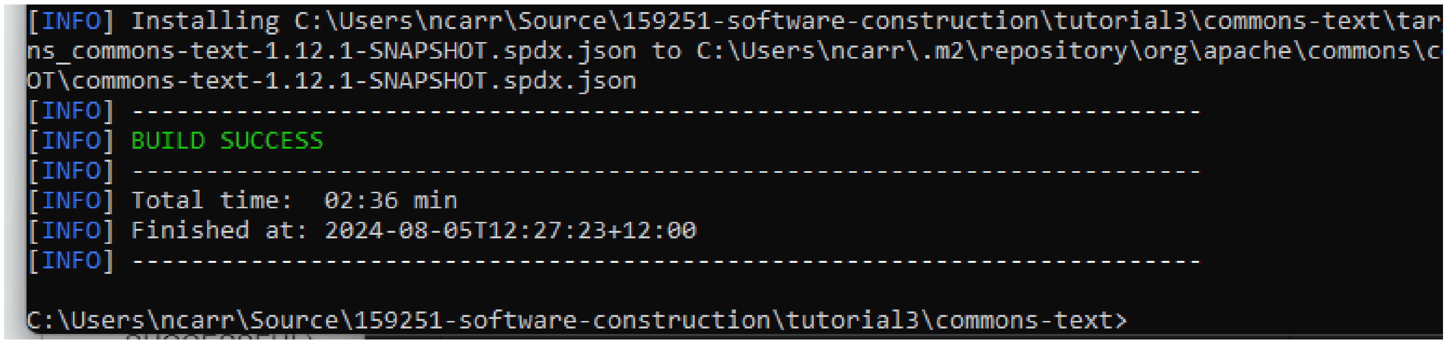
```
C:\Users\ncarr\Source\159251-software-construction\tutorial3>git clone https://github.com/apache/commons-text.git
Cloning into 'commons-text'...
remote: Enumerating objects: 20373, done.
remote: Counting objects: 100% (220/220), done.
remote: Compressing objects: 100% (105/105), done.
remote: Total 20373 (delta 108), reused 178 (delta 78), pack-reused 20153
Receiving objects: 100% (20373/20373), 3.68 MiB | 2.65 MiB/s, done.
Resolving deltas: 100% (10349/10349), done.
```

Clone this repository to your local directory/folder

change to the folder for the project you just cloned and run Maven to resolve dependencies (see Maven tasks in Tutorial 2 for more information) :

```
    mvn clean install
```

With a clean run, you should have a successful build with no issues (should show BUILD SUCCESSFUL).



```
[INFO] Installing C:\Users\ncarr\Source\159251-software-construction\tutorial3\commons-text\tar
ns_commons-text-1.12.1-SNAPSHOT.spdx.json to C:\Users\ncarr\.m2\repository\org\apache\commons\c
OT\commons-text-1.12.1-SNAPSHOT.spdx.json
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  02:36 min
[INFO] Finished at: 2024-08-05T12:27:23+12:00
[INFO] ------------------------------------------------------------------------

C:\Users\ncarr\Source\159251-software-construction\tutorial3\commons-text>
```

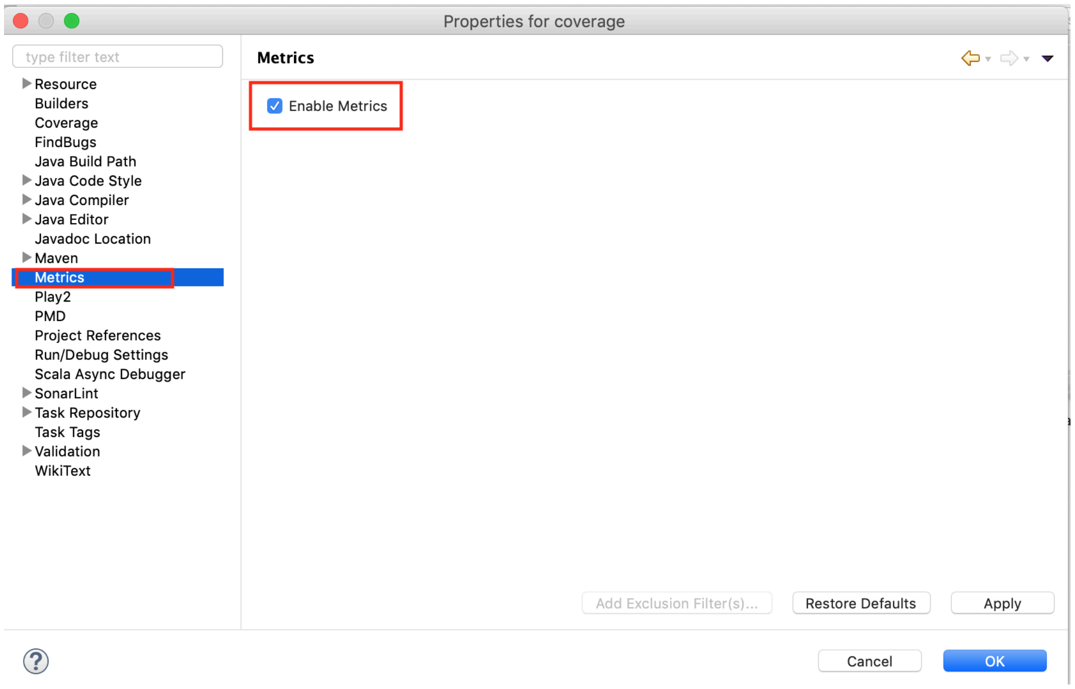
<u>Part 1: Measure your code size and complexity using Eclipse Metrics Plugin</u>

Get the Tool First!

<u>Using a metrics tool/plugin:</u> you can collect various source code size and complexity metrics including Line of Code (LOC), Coupling Between Objects (CBO) and Cyclomatic Complexity (CC).

For this exercise, you need to install a plugin (I used the metrics2 plugin for Eclipse as an example, but alternatives are also shown below), and then activate the plugin so it reads your project).

1. Install the metrics2 plugin (Help menu -> Install New Software ... On the opening dialog click on the Add ... button. Add a new Remote Site with the following URL **http://metrics2.sourceforge.net/update/** and follow the instructions for installation. Restart Eclipse (**File-> Restart**)
2. import your commons-text project into Eclipse by File -> Import -> Maven (Existing Maven Project) and select the project.
3. Check that you are in the Java Perspective (Windows -> Perspective -> Open Perspective -> Java.)
4. Now make sure that you activate the plugin for your project by **right-click on commons-text -> Properties -> Metrics -> Enable Metrics (see below)**



☑ Enable Metrics

5. Make sure that you have the correct Metrics View from **Windows -> Show view -> Metrics view**

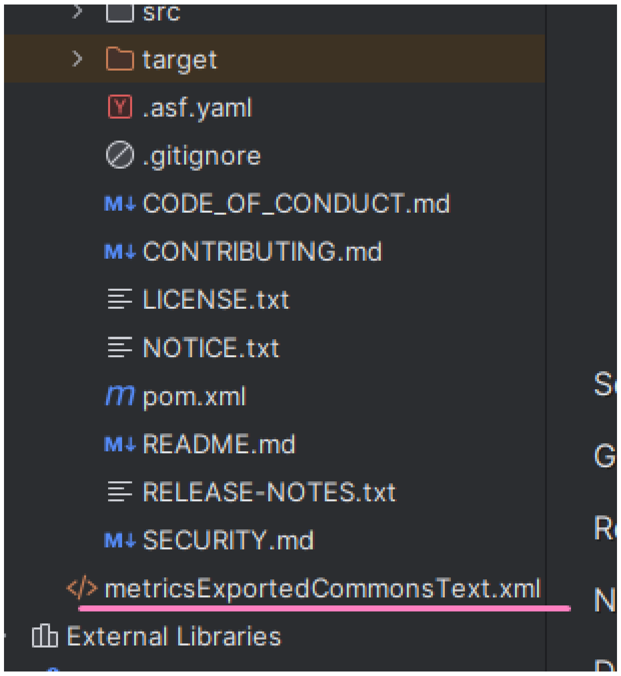
Alternative Plugins

IntelliJ IDEA: MetricsReloaded (tested) and MetricsTree

Eclipse: CodeMR (limited for free), also available for IntelliJ

Task

1. Generate the metrics reports for Apache Commons Text. You can click on the export icon in the Metrics window to save it in XML format.



src
target
.asf.yaml
.gitignore
CODE_OF_CONDUCT.md
CONTRIBUTING.md
LICENSE.txt
NOTICE.txt
pom.xml
README.md
RELEASE-NOTES.txt
SECURITY.md
metricsExportedCommonsText.xml
External Libraries

2. Open a separate text file (.txt), and report the following values from your tool:

```
a. number of files
b. number of classes
c. name of the largest class file (.java) in the project based on
   the LOC value
d. The name of the two classes (or packages) with the highest
   coupling values (Using one of the following metrics:
   Afferent/Efferent coupling or Coupling Between Objects (CBC))


e. The name of the class with the highest Cyclomatic Complexity (or
   weighted complexity) value.
```

Notes:

- If you are using MetricsReloaded, Coupling metrics (i.e., CBC) can be found under the Chidamber-Kemerer metrics block.

- Also, for IntelliJ plugins, you can generate reports only for one category of metrics at a time. If you are using MetricsReloaded, then generate an XML report for Chidamber-Kemerer metrics *only*.

Part 2: Measuring code quality using PMD

PMD is a popular source code analyser that statistically checks for potential bugs and flaws in your code. It finds problems and issues in your code caused by programming behaviour and coding style. You will use the Maven plugin to generate PMD code quality reports. You don't need your IDE for this task!

Task

There is a PMD Maven task in the Apache Common Text project that can generate PMD code check reports. First, check if the PMD plugin is already in your project's pom.xml file.

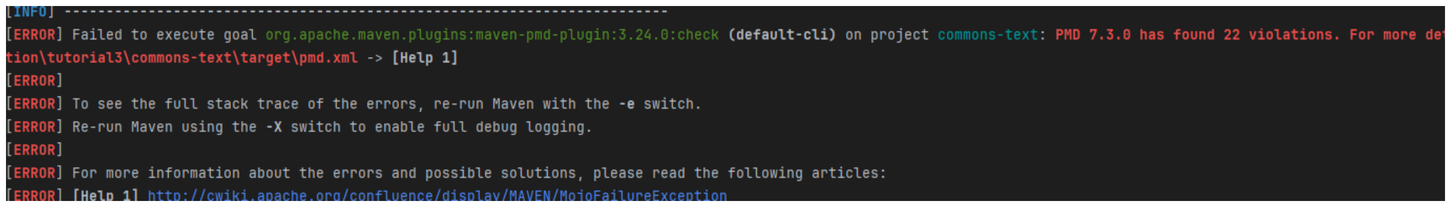
First, navigate to the commons-text project from the terminal...

Next, run a PMD check using the following command:

```
mvn pmd:check
```

This will check for violations using default rulesets. This should show a failed build (Build Failure) and will tell you how many PMD violations you have in the code. In this case, there should be a number of violations.

Take a screenshot of the result screen showing the violations.



```
[INFO] ------------------------------------------------------------------------
[ERROR] Failed to execute goal org.apache.maven.plugins:maven-pmd-plugin:3.24.0:check (default-cli) on project commons-text: PMD 7.3.0 has found 22 violations. For more de
tion\tutorial3\commons-text\target\pmd.xml -> [Help 1]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/MojoFailureException
```

Run and generate reports from Maven using:

```
mvn pmd:pmd
```

Output files can be obtained from <Project_Home>/target. You will need the following reports:

```
~/commons-text/target/pmd.xml
```

```
~/commons-text/target/site/pmd.html
```

In the generated pmd.html, have a look at those violations, and check the lines they refer to in the files.

What to submit

Part 1:

- a text file, including all metrics data, as explained in Part 1.
- The XML report that you generated for the commons-text project.

Part 2:

- copy the two reports (`pmd.html` and `pmd.xml`) files generated from the commons-text project.
- screenshot of the Failed Build (showing the number of violations found).

Submit the following...

Once done with the analysis, create a folder Tutorial3 and add the two folders you have created in the previous tasks (Metrics and PMD) - don't forget to copy files within the sub-folders too ;)

Zip the folder 'Tutorial 3' and submit the file on Stream.