MP1: Java and Build System Basics

1 Introduction

The best way to learn software engineering is to practice software engineering in the actual software development process. Therefore, in this and the following MPs, you will work with a real-world project named Jsoup to learn basic concepts, methodologies, and techniques in software engineering.

Jsoup (https://jsoup.org) is a popular Java library for working with HTML. It provides a very convenient API for fetching URLs and extracting and manipulating data, using the best HTML5 DOM methods and CSS selectors.

All our MPs will be based on a specific Jsoup version (https://github.com/CS427-UIUC/jsoup), which you can obtain via the following command:

- `git clone https://github.com/CS427-UIUC/jsoup`
- Note: The command above will download Jsoup on your local machine in a directory called "jsoup"

Please make sure that you obtain the above Jsoup version before you start any of the MPs. You can use the same copy for all your MPs (i.e., you do not need to clone a fresh copy of the above Jsoup version for each MP). You can use Mac, Linux, or Windows to complete all the MPs (Mac/Linux will be recommended). You are also recommended to install the required software below before proceeding to the rest of MPs:

- Java: 1.8+
  - The Java tutorials: https://docs.oracle.com/javase/tutorial/
  - Introduction to programming in Java: https://math.hws.edu/javanotes/index.html
  - **We use Java 1.8 for grading**
- Maven: 3.5+
  - Maven in 5min: https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html
  - Maven getting started guide: https://maven.apache.org/guides/getting-started/index.html
  - **We use Maven 3.8.8 for grading**
- Ant: 1.10+ (needed only for MP1, not later MPs)
  - Ant manual: https://ant.apache.org/manual/
  - Installation:
    - *Mac*: `brew install ant`
    - *Windows & Linux* https://ant.apache.org/manual/install.html
  - **We use Ant 1.10.13 for grading**
- Gradle: 8.0+ (needed only for MP1, not later MPs)
  - Gradle manual: https://docs.gradle.org/current/userguide/userguide.html
  - Installation: https://gradle.org/install/

    * *Mac*: `brew install gradle`
    * *Linux*: `sdk install gradle 8.10`
    * *Windows*: https://stackoverflow.com/questions/34856932/how-do-i-install-gradle-on-windows-10
  - **We use Gradle 8.10 for grading**

This Jsoup version comes with a default build file in Maven (i.e., pom.xml); you should be able to directly run `mvn test` to execute all its 658 JUnit tests. To run this command, ensure you see the pom.xml file in your path. Otherwise, you will receive the following error:

```
[ERROR] The goal you specified requires a project to execute but there is no POM in the
    current directory
```

If successful, it takes a few minutes to build the project (the first time, when the dependencies are downloaded, and it takes less time later) and execute the test suite. You should see the following message at the end:

```
Results :
Tests run: 658 , Failures: 0, Errors: 0, Skipped: 0
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 01:09 min
[INFO] Finished at: 2023-01-01T01:30:34+04:30
[INFO] ------------------------------------------------------------------------
```

Task: You are expected to write two other build files under Ant and Gradle, respectively, to run the same set of tests. Please follow the steps below to finish these tasks:

- If you want to refer to the lecture on "GitHub and Build Systems", you can check out the PDF version on coursera.
- You can find the examples discussed in the lecture on different build systems, including Maven, Ant, and Gradle, here: https://github.com/CS427-UIUC/scm-example. You are required to reuse and modify them for this MP.
- Write two build files, namely "build.xml" and "build.gradle", to complete the tasks for this MP.

2 Instructions for Ant

- Your build file should include the following steps (avoid including other steps, as this may interfere with the auto-grading):
  - *Variable declarations*: All the jar files used in the Maven build file have been downloaded for you and stored in the `ant-libs` directory. In the path declaration tag, introduce all the jar files in the "ant-libs" directory. Not including these jar files in the path declaration will fail the test execution. Use relative paths to access the libs when building the project. Your path id in the Ant build file should be set to your NetID.classpath.
  - Compile all source Java files from `src/main/java` to `ant-target/classes`. You can achieve this via modifying the `build` target from "scm-example". Please make sure that you include the encoding="UTF-8" option for javac to avoid encoding issues on some specific OS settings.
  - Compile all test Java files from `src/test/java` to `ant-target/test-classes`. You can achieve this via modifying the `build-test` target from "scm-example". Please make sure that you include the `encoding="UTF-8"` option for `javac` to avoid encoding issues on some specific OS settings.

- Copy all test resource files from `src/test/resources` to `ant-target/test-classes`. You can either add this to the existing `build-test` target or define a new target. If you define a new target, make sure that the "test" target depends on the new target so that the tests can access the resource files.
- Define the actual test execution target. You only need to change the classpath setting for the `test` target from "scm-example". Please change the JUnit testing report output directory (under `junitreport` tag in the build file) to `ant-target/reports`, so that the HTML JUnit testing reports will be generated under the directory `ant-target/reports/html/`; otherwise, the autograder may not be able to find the HTML report for your test execution and you may lose all your points.

- You can test the correctness of written build files via the following commands: `ant clean test`

Note that you may receive the "build successful" message in the command line, while your tests failed during execution. You should look into the test results to ensure the correct build and test execution (all the tests pass). You will get the full score only if the "success rate" in the HTML report is 100%. Ant does not print the test execution results by default. To obtain the number of test executions, you can check the corresponding HTML report: `ant-target/reports/html/index.html` (automatically generated once you follow the above instructions and correctly configure the report output directory under `junitreport`). Please make sure that you can generate the HTML report under the specified report path; otherwise you may lose all the points.

Summary

Tests	Failures	Errors	Skipped	Success rate	Time
658	0	23	0	96.50%	4.977

Note: *failures* are anticipated and checked for with assertions while *errors* are unanticipated.

Packages

Name	Tests	Errors	Failures	Skipped	Time(s)	Time Stamp	Host
org.jsoup.helper	44	5	0	0	0.510	2023-01-14T21:28:34	MacBook-Pro-260.local
org.jsoup.integration	12	11	0	0	0.151	2023-01-14T21:28:35	MacBook-Pro-260.local
org.jsoup.internal	9	0	0	0	0.139	2023-01-14T21:28:36	MacBook-Pro-260.local
org.jsoup.nodes	203	1	0	0	1.338	2023-01-14T21:28:36	MacBook-Pro-260.local
org.jsoup.parser	234	6	0	0	1.900	2023-01-14T21:28:40	MacBook-Pro-260.local
org.jsoup.safety	34	0	0	0	0.166	2023-01-14T21:28:45	MacBook-Pro-260.local
org.jsoup.select	122	0	0	0	0.773	2023-01-14T21:28:45	MacBook-Pro-260.local

Figure 1: Example of an Ant test report where some tests fail because the test resources are not properly copied

3 Instructions for Gradle

- Your build file should include the following components:
  - *Plugins*: Identifies that the build is for a Java project.
  - *Repositories*: Use `https://repo.maven.apache.org/maven2` to resolve all the required dependencies in the project.
  - *Dependencies*: You should specify the same dependency versions used in the Maven build file so that Gradle can automatically fetch them from the Maven repo, or directly use the downloaded jar files in the `ant-libs` directory. Failing to do so will result in exceptions while running the build file.

Summary

Tests	Failures	Errors	Skipped	Success rate	Time
658	0	0	0	100.00%	5.157

Note: *failures* are anticipated and checked for with assertions while *errors* are unanticipated.

Packages

Name	Tests	Errors	Failures	Skipped	Time(s)	Time Stamp	Host
org.jsoup.helper	44	0	0	0	0.616	2023-01-14T21:34:13	MacBook-Pro-260.local
org.jsoup.integration	12	0	0	0	0.261	2023-01-14T21:34:14	MacBook-Pro-260.local
org.jsoup.internal	9	0	0	0	0.132	2023-01-14T21:34:15	MacBook-Pro-260.local
org.jsoup.nodes	203	0	0	0	1.348	2023-01-14T21:34:15	MacBook-Pro-260.local
org.jsoup.parser	234	0	0	0	1.886	2023-01-14T21:34:19	MacBook-Pro-260.local
org.jsoup.safety	34	0	0	0	0.164	2023-01-14T21:34:24	MacBook-Pro-260.local
org.jsoup.select	122	0	0	0	0.750	2023-01-14T21:34:24	MacBook-Pro-260.local

Figure 2: Example of a successful Ant build report (note the difference in the success rate)

  - *Description*: The value for description should be your NetID (`description = 'Your NetID'`)
  - *Output directory*: Instead of using the default `build` output directory, your build file should use `gradle-target` as your output directory to store all the build outputs (including class files and reports).

- You can test the correctness of written build files via the following commands: `gradle test`

Note that you may receive the "build successful" message in the command line, while your tests failed during execution. You should look into the test results to ensure the correct build and test execution (all the tests pass). You will get the full score if the "success" in the HTML report is 100%. Like Ant, Gradle does not print the test execution results by default. To obtain the number of test executions, you can check the corresponding HTML report: `gradle-target/reports/tests/test/index.html` (default report directory for Gradle once you set the output directory to `gradle-target`). Here is the example output for Gradle:

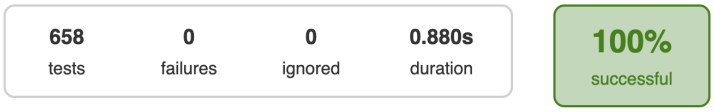

Figure 3: A successful build results in 100% success for test execution in the report

4 Deliverables

You are expected to upload a zip file including only the `build.xml` and `build.gradle` files you completed (**no directories** please). The name of the zip file should be your NetID. In other words, the hierarchy of your submission zip file should be like this (replace `YourNetID` with your NetID):

```
YourNetID.zip
|-- build.xml
|-- build.gradle
```

To create such a zip file, you can run the following command in your terminal:

```
zip YourNetID.zip build.xml build.gradle
```

The autograder will copy your zip file into the `jsoup` directory, unzip it, and run the build files. Failing to follow this structure may result in losing all the points.

Warning: you may lose all your points if you violate the following rules:

- Please **DO NOT use any absolute paths** in your solution since your absolute paths will not match our grading machines. Also, please **only use "/"** as the file separator in your relative paths (if needed for this MP) since Java will usually make the correct conversions if you use the Unix file separator "/" in relative paths (for both Windows and Unix-style systems).
- Please **DO NOT fork any assignment repo** from our GitHub organization, `CS427-UIUC`, or share your solution online.
- Please make sure that you **follow the detailed instructions** to use the specified output directories. The autograder will only look for the specified directories; thus you may lose all points if you fail to use the same directories (even if you can pass all tests).

5 Grading rubric

Overall score (5pt)

- Ant build file (3pt): the autograder will check `ant-target/reports/html/overview-summary.html`
  - If you pass all 658 tests, you get 3pt
  - Else if you pass 600+ tests, you get 2pt
  - Else if you pass some tests, you get 1pt
  - Else you get 0pt
- Gradle build file (2pt): the autograder will check `gradle-target/reports/tests/test/index.html`
  - If you pass all 658 tests, you get 2pt
  - Else if you pass 600+ tests, you get 1pt
  - Else you get 0pt