SENG 371 – Spring 2024
Lab 02 – Concept Location

This lab explores basic techniques of concept location. By the end of this lab, you will:

- Copy a public repository from GitHub into a private repository;
- Clone a repository from GitHub for your developer work;
- Create a working directory from a project repository in the Eclipse IDE;
- Explore a repository in your working directory using the Eclipse IDE;
- Locate simple concepts in the source code of a software project.

Install some software first (please do this part before the lab session)

Make sure you installed the Java Development Kit version 20 or newer (JDK 20 is needed for running JabRef). Fix both the PATH and the JAVA_HOME variables because some JabRef scripts depend on them.

After that, install Eclipse IDE for Java Developers (pick the version for your operating system):

https://www.eclipse.org/downloads/packages/

This includes the essential tools for Java developers, including a Java IDE, a git client, XML editor, Maven and Gradle integration.

To make sure you are running all the necessary Java tools, go to **Help > Install new software > Work with**, choose -- All Available Sites -- and select **Programming Languages -> Eclipse Java Development Tools**, and follow the default steps to install the tools. You will be using **JUnit 5** from the Eclipse Java Development tools to run your tests.

To run JabRef build scripts, you cannot use the Windows **cmd** tool. Instead, you need a POSIX-compliant shell, such as **bash** or **zsh**. If you run Linux or MacOS, you know you have them. If you run Windows, install **cygwin** first and use it as your shell (recall that Cygwin considers your C drive as `/cygdrive/c/` in Unix-like directory trees.).

Create a private JabRef repository from GitHub

Log in to your GitHub account and create a private copy of the JabRef project, which is at:

https://github.com/JabRef/jabref

To do so, do not fork the original repo, because a forked repo will be public, you want it to be private. Follow these steps instead.

1. Either open your command-line git client (for instance, `git bash` on Windows, available at https://gitforwindows.org/) or use the terminal and your regular `git` client on Linux/Mac.
2. Make sure you have configured your git client with your own data (replace name and email with your own).

   ```
   git config --global user.name "John Doe"
   git config --global user.email johndoe@example.com
   ```
3. Create a bare clone of the repository by typing:

   ```
   git clone --bare https://github.com/JabRef/jabref.git
   ```
4. Create a new private repository on GitHub and name it `frozen-jabref` . If you are unable to create a private repo, you can request unlimited private repos as a student by getting the student pack from GitHub.
5. Before you push your private repo, you first need to create an RSA key to be able to use SSH instead of HTTPS, since GitHub requires using SSH for private repositories. A simple way to create an RSA key that works on all operating systems is using Eclipse to create the RSA key to you. Enter Eclipse and:
   a. Create an SSH key (skip this step when you already have one):
      i. In Eclipse, in the **Window > Preferences > General > Network Connections > SSH2**, go to the **Key Management** tab and hit the **Generate RSA Key...** button

ii. Hit **Save Private Key...** and choose a location, preferably the subfolder `.ssh` of your user home directory
iii. Check your .ssh directory and make sure you see two files there, a private key (usually id_rsa) and a public key (usually, id_rsa.pub). Make sure you change the permissions of your private key file so only you can read it.

b. Upload your public key to your GitHub account:
i. For a new created key, copy the string shown in the **Key Management** tab to the clipboard; for an existing key add it in the preferences **General > Network Connections > SSH2**, go to the **General** tab and copy the content of the public key file <name>.pub
ii. Go to your GitHub Account Settings (click on your icon/picture), choose the **SSH and GPG keys** section and hit the **New SSH key** button
iii. Paste the copied public key into the Key field and hit the **Add SSH Key** button

6. Push your bare clone to your new `frozen-jabref` repository. Replace <your_username> with your actual GitHub username in the URL below.

```
cd jabref.git
git push git@github.com:<your_username>/frozen-jabref.git
```

7. After you push the repo clone, you may browse your new `frozen-jabref` GitHub repo.
8. Remove your temporary local repository your created in step 2.

```
cd ..
rm -r jabref.git
```

9. You can now clone your `frozen-jabref` repository on your machine (for instance, in the `~/git` folder).

```
cd ~/git
git clone git@github.com:<your_username>/frozen-jabref.git
```

10. You are done. You may check your repo by entering the `frozen-jabref.git` directory and browsing your files. If you enjoy committing from the command line, there you go.

Use the Eclipse IDE and the Eclipse IDE plugin to interact with your repo

To work in Eclipse with your git `frozen-jabref` repository, make it available to Eclipse first. To do so:

1. Use the SSH URL of your already cloned repository:
   a. On GitHub, go to the `frozen-jabref` repo, and click on the **Code** button, choose **SSH** and copy the URL;
   b. In Eclipse, go to **Window > Show View > Other > Git > Git Repositories**, and, in the Git Repositories view, choose the **Clone a Git Repository**;
   c. In the **Source Git Repository** window, paste the remote SSH URL:
   `url = git@github.com:<your_username>/frozen-jabref.git`
   Obs.: In case a repo is already found there (because of the previous cloning you did in the previous section), you may choose to delete the existing `frozen-jabref` directory inside your `~/git` directory to let Eclipse clone the repo.
   d. Finish the process of cloning the repo. Your repo will be cloned at your git folder as configured in Eclipse, and your project will be available to you.
   Obs.: In Eclipse, you may always go to the **Git Repositories** view. If you had used the git before, the repo might already be available. Otherwise, choose the "Add an existing git repository to this view" button, choose the directory where your repo is located (for instance, `C:\Users\<your_username>\git\frozen-jabref` or `~/git/frozen-jabref`), click the Search button, and choose the repo on the search results.
2. Import your JabRef project into a working directory. Go to **File > Import > Projects from Git > Existing local repository > Next**, choose the existing Java `frozen-jabref` and follow the default steps until **Finish**.
3. From your project root directory (for instance, `C:\Users\<your_username>\git\frozen-jabref` or `~/git/frozen-jabref`), run some scripts to see the JabRef code working.
   a. `./gradlew run` (runs the JabRef desktop application)

   If you want to develop, you will need to generate additional code. To do so, run both:

b. `./gradlew generateSource` (generates some source code)
c. `./gradlew eclipse` (generated Eclipse-related code)

4. After you run those scripts, go to **Window > Show View > Other > Gradle > Gradle Tasks**. In the Gradle Tasks window, click on the link **Import a Gradle Project**. Choose your project root directory (for instance, `C:\Users\<your_username>\git\frozen-jabref` or `~/git/frozen-jabref`), follow the steps until clicking the Finish button. With that, you will have some build targets available from Eclipse.
5. Browse your `frozen-jabref` working directory on the **Package Explorer > `frozen-jabref`** project. The `src` folder has the source files for this project. Look for `main > java > org > jabref` (or, even better, if your build scripts ran well, `src/main/java` will organize the packages under `org.jabref`. The main Java code for JabRef is there. Browse the code a bit to get familiar with the code organization of this project.

Concept location by GREP search

Change Request:

> Include the name of the members of your team/pair in the screen "**Help > About JabRef**", inside the section "**Maintainers**".

Using a GREP search strategy, identify the class where you will perform this change. Record your steps to later show the TA which way(s) you followed. To do that in Eclipse, use the **Search > File Search**.

Locate a less simple concept on source code

Change Request:

> Change the shortcut key for the menu item "**File > Close library**" from Ctrl+W to Ctrl+Shift+W.

Using a class dependency search strategy, identify the class where you will perform this change. Record your steps to later show the TA which way(s) you followed.

Hint: you may want to start from the `org.jabref.gui.actions` package, since shortcuts are usually related to actions.

Present your results to your TA

Take a few minutes to present the results of your work to your TA. That is how you will be assessed in this lab.

You may use more than one checkpoint during the lab session to show the TA your results as soon as you finished each relevant task. For example:

1. Creating and populating the private JabRef repository on GitHub;
2. Interacting with JabRef from Eclipse (running JabRef from your Eclipse working directory);
3. Locating the first concept;
4. Locating the second concept.