Lab 1 - Setup

Summary

The purpose of this assignment is to ensure everyone has a functional development environment installed and tested for ITMD4515. We will also learn the basics of Maven by creating our first Java web application, and create a basic HTML web app to introduce ourselves.

https://classroom.github.com/a/xVUWSZEj

Requirements

Canvas and Beacon LMS

You will get your assignments, submit your assignments, and receive grades/feedback through Canvas or Beacon LMS. Make sure you can login to the system you have been assigned!

GitHub

We are using GitHub for several reasons. First, to simulate real organizations and projects. Git is widely used, if not industry standard at this point. Second, this will enable me to better assist you with debugging your issues this semester. Third, we will write our project documentation using GitHub README files.

For Lab 1, perform the following steps:

1. Create a GitHub Account with your **myIIT username,** and using your IIT email as your email.

   If you already have a GitHub account, you can use that. Just accept the assignment URL using your existing account, and make sure you select your myIIT username in GitHub classroom!

2. Create a **Personal Access Token** for use with NetBeans:
   1. Access your Settings by clicking on the upper right menu by your profile picture
   2. Access Developer Settings
   3. Click Personal Access Tokens and Generate New Token
      1. *Make sure you choose the Classic token, and not the fine-grained token*
   4. Give it a name like Spring 2024 ITMD4515
   5. Give it a duration that will last all semester, or unlimited if you intent to use it in the future.
   6. Give it all repo and user scopes

    7. Generate the token, and save the token in your password safe! It will never be displayed in clear text again.
3. Open the Canvas assignment for Lab 1
4. Click on the assignment invitation URL
5. Login to GitHub using the account your just created with your **myIIT username**
6. Join the classroom roster by selecting *your* **myIIT username** from the list

   Make sure you select your username and not someone else's username. GitHub has made this a one-click deal. If you mistakenly click someone else's username, you will have to email me and ask for help.

7. Accept the assignment
8. Your Lab 1 repository will then be created automatically and shared with me
9. You will receive a confirmation email:

@spyrison has invited you to collaborate on the itmd4515/itmd4515-s24-lab1-sspyriso repository

Download and Install

Download and install the following:

1. Download **OpenJDK 17 (LTS)** with the Hotspot JVM from https://adoptium.net/temurin/releases/?version=17 and install with the defaults. Version numbers below are for example only.

   If you already have the latest version of another JDK, you can use that. **You must have a JDK installed, not just a JRE.**

    1. This will be an MSI installer on windows, a PKG installed for Mac, or a tar.gz archive on linux. On Mac, if you use the homebrew package manager you can also install with the brew command:

       linux/macOS

       ```
       # linux
       cd /your/preferred/location/for/the/jdk
       tar -zxvf OpenJDK21U-jdk_x64_linux_hotspot_17.0.1_12.tar.gz

       # macOS only - if you are using the homebrew package manager
       brew install openjdk
       ```

    2. Run the installer with default options, but on Windows make sure to select the checkbox to **Set JAVA_HOME environment variable** which is not selected by default. This will save you a lot of time, especially if you are unfamiliar with environment variables or operating systems in general.

   3. Additional installation instructions are available for all platforms here: https://adoptium.net/installation/
   4. For Mac users, you may need to update your own environment variables (ref: http://www.mkyong.com/java/how-to-set-java_home-environment-variable-on-mac-os-x/):
      1. Open the Terminal application
      2. Add the following to the profile you use, typically .profile or .bash_profile:

         MacOS

         ```
         # if you installed OpenJDK as outlined above, via PKG
         installer or homebrew, you can probably just do this:
         export JAVA_HOME=$(/usr/libexec/java_home -v 17)
         export PATH=${JAVA_HOME}/bin:${PATH}
         ```

   5. For Linux users, add the following to the profile you use, typically .profile or .bash_profile:

      Linux

      ```
      export JAVA_HOME=/wherever/you/extracted/the/JDK/from/above/jdk-
      17.0.1_12
      export PATH=${JAVA_HOME}/bin:${PATH}
      ```

      If you choose to use Linux in this class, I applaud your choice but also expect you to know the command line.

2. Test! Open up a cmd prompt or PowerShell on Windows (PowerShell illustrated below) and do the following. Make sure your output is sane. The following is an abbreviated example, not complete. Make sure you **see** the environment variables you just set, and that you can execute commands within the JDK. For Mac/Linux, open your native Terminal application:

   ```
   PS C:\Users\sas691> gci env:
   ...
   JAVA_HOME                      C:\Program Files\AdoptOpenJDK\jdk-
   17.0.1_12-hotspot\
   ...
   Path                           C:\Program Files\AdoptOpenJDK\jdk-
   17.0.1_12-hotspot\bin;C:\ProgramData\DockerDesktop\version-
   bin;C:\Program
   Files\Docker\Docker\Resources\bin;C:\WINDOWS\system32;C:\WIND...
   ...

   PS C:\Users\sas691> java -version
   openjdk version "17.0.9" 2023-10-17
   OpenJDK Runtime Environment Homebrew (build 17.0.9+0)
   OpenJDK 64-Bit Server VM Homebrew (build 17.0.9+0, mixed mode, sharing)
   ```

```
PS C:\Users\sas691> javac -version
javac 17.0.9

# on linux or Mac, you can view environment variables differently in a
terminal:
env | egrep 'JAVA|PATH'
```

3. Download the IDE you wish to use. The only requirement is that it supports a standard maven project structure. If you would like to use Apache Netbeans, do the following:
    1. Download the latest Apache NetBeans from https://netbeans.apache.org/download/index.html. Install with the defaults.
    2. The first time you run NetBeans, or perhaps the first time you run a maven project within NetBeans, you might find a very slow and long-running task named Unpacking index for Central Repository. Let it run and finish the first time, which might take as long as 10 or 20 minutes. After it finishes, you can control how often this task runs in your NetBeans preferences.
    3. Additionally, you will find that at various initial stages (first time you open a maven project, first time you start a Jakarta EE project, first time you add a server, etc) NetBeans will want to install and activate various plugins. Allow it to do so. If it appears that NetBeans is locked up, first be patient - it may just be activating a plugin. If it doesn't respond after a reasonable amount of time, close and re-open NetBeans.
4. Download the latest non-beta version of Payara Server Community, which is fully compatible with Jakarta EE: https://www.payara.fish/downloads/payara-platform-community-edition/
    1. This will be a zip file. Extract it somewhere on your computer, to a path that ***does not*** have any spaces. I used C:\Users\me\Documents\IIT\payara6 for mine. On Mac, I use a path such as /Users/me/Documents/IIT/payara6. The specific path doesn't matter. Making sure it doesn't have spaces **does** matter.
    2. Follow my demo to add Payara to NetBeans as a server.
    3. Important - on macOS, you may need to adjust the NetBeans proxy settings (select No Proxy). If Payara starts normally for you, don't worry about this. If Payara is slow to start, or NetBeans complains about starting/stopping Payara, then set the "No Proxy" and try again. (ref: https://netbeans.org/bugzilla/show_bug.cgi?id=268076)
5. Download the latest MySQL Community Server **8.0** from https://dev.mysql.com/downloads/mysql/
    1. ***Do NOT use the version 8.x Innovation version!***
    2. I recommend selecting a Custom Setup Type, so you don't end up with Excel and Visual Studio add-ons.
    3. Then choose the following:
        1. MySQL Server 8.0 (latest version)
        2. MySQL Workbench (latest version)
        3. Samples and Examples 8.0 (latest version to match MySQL Server above)
    4. If you are missing any dependencies, the installer will prompt you to install them. For example, on my Windows VM I needed Visual C++ Runtime. The installer will download the missing dependencies for you, and prompt you to execute them.

5. After successful installation, MySQL will prompt you to configure. Accept the default Development Machine configuration, and set a MySQL root password of your choosing.
6. On the "Windows Service" screen, it is up to you whether to start the MySQL service at boot. I chose **not** to do so on my Windows environments, because I prefer to manually control the database. I recommend de-selecting the checkbox next to "Start the MySQL Server at System Startup."
7. Finish the installation
8. MySQL Notifier
    1. This doesn't seem to be bundled with the installer anymore. You need to download the Notifier directly from here: https://downloads.mysql.com/archives/notifier/
    2. In the past I have had trouble with the MySQL Notifier the first time I use it to stop/start a MySQL instance. Restart your computer after installing MySQL notifier, or close and re-open the Notifier if you don't want to fully restart. If you are still having trouble with MySQL notifier on Windows after restarting, ask a question in our discussion forums so I can help.
9. Mac users: There are several different ways to install MySQL on Mac. If you already have a preferred way, use it. There is a PKG or DMG installer that will give you a MySQL System Preferences control panel which many of you might find helpful. If you need assistance or a recommendation, please reach out to me on the discussion forums. Personally, I use the Homebrew package manager for MySQL and many other things on Mac, and I manually start and stop MySQL when I need it for development. ***Mac users will need to install Workbench separately, as well as any sample/example databases in future labs.***

    MacOS

    ```
    # the fully qualified path below is because homebrew doesn't put
    # mysql@8.0 on the homebrew PATH anymore.  You can add to your
    # PATH if you want of course
    brew install mysql@8.0
    brew services start mysql@8.0
    /usr/local/Cellar/mysql@8.0/8.0.35/bin/mysql_secure_installation
    brew services stop mysql@8.0
    ```

10. Linux users: This will vary by distribution. ***Linux users will need to install Workbench separately, as well as any sample/example databases in future labs.*** Here are some articles for various distributions:
    1. Install MySQL Server on Ubuntu (from Rackspace Support)
        1. **Note** - unless you are setting up a server for remote access, you can skip the "Allow remote access" section. The important commands are:

            Linux

            ```
            sudo apt-get update
            ```

```
sudo apt-get install mysql-server
systemctl start mysql
systemctl enable mysql

# if mysql_secure_installation didn't run
automatically after installation, run it manually
```

6. Open MySQL Workbench
    1. Make sure you can connect as the root user to one of the sample databases.
    2. Run a simple select * from table query to show you are properly connected

MySQL Database and User

The purpose of this is to simulate real organizations and projects at work. Most of you will not have access to OS or database root passwords. Following the principle of least privilege, you (or your application) will have access to an account that has enough privileges to function, but not system-wide administrative privileges. We will discuss this more in class.

1. Create a new MySQL database schema named **itmd4515**. We will be using this database for the remaining projects this semester.
    1. Enter a schema name of itmd4515. Leave the default collation blank. Click apply.
    2. Click apply on the subsequent confirmation window. This window displays the actual SQL commands used to create the database.
2. Create a MySQL user named **itmd4515** with full rights to the database(s). The user must have a password of **itmd4515**. Do not use the root user for connection pools. This user may already exist from your prior home work. If so, you do not need to re-create.
    1. If your Workbench is displaying **Schemas,** then click on **Management** or **Administration** to access the management options
    2. Click on **Users and Privileges**
    3. Click on **Add Account**
        1. Give your new account a Login Name of itmd4515
        2. Limit your new account to Hosts Matching localhost
        3. Give your new account a password of itmd4515
        4. Click on Apply to create the account
        5. Click on **Schema Privileges,** add your new **itmd4515** schema, select "All" privileges, and click apply.
3. Finally, create a NetBeans database connection to your new MySQL database. This database connection will be by various tools within NetBeans. This will be demonstrated in class.

Document this with a screenshot on your readme file showing that you have created the user, and granted full permissions to ***only*** the itmd4515 schema.

Project Setup

Next create a **Java with Maven Web Application** project in NetBeans with the following coordinates and configuration:

1. Project Name: **uid-lab1** (uid means your myIIT username)
2. Artifact ID: **uid-lab1** (default is fine)
3. Group ID: **edu.iit.sat.itmd4515.uid**
4. Version: **1.0-SNAPSHOT** (default is fine)
5. Use a base package for your Java code of **edu.iit.sat.itmd4515.uid**
6. Under Server, choose Payara Server and Jakarta EE 10 Web

Once your project is open in the NetBeans project window, highlight your project root (sspyriso-lab1 in my example) and then right-click to get the New → File menu. Create a new file named **readme.md.** Write some initial content in the file such as "Your Name Lab 1 README" and save the file

For example, my project will be called “**sspyriso-lab1**" with a base package and group id of **edu.iit.sat.itmd4515.sspyriso**.

Finally, configure your project for source control:

1. Using NetBeans or the Git command line, initialize your project as a Git repository and add/commit your project
    1. In NetBeans, this can be done in **Team -> Git -> Initialize Repository**
    2. At the root level of your project, choose **Team -> Commit**
    3. Enter a commit message such as "Initial project import" and Commit.
2. Add your GitHub repository as a remote repository. In NetBeans, this can be done via **Team -> Remote -> Push**. The first time you run this command, it will prompt you for the following to setup a remote:

   If you clone from a remote repository, it will automatically be named "origin." In this case, we are going to add our remote repository with the name origin to satisfy that naming convention. Refer to the "Quick Setup" help section on your GitHub repository homepage.

    1. In the subsequent Remote Repository dialog box, enter (copy/paste) the https address of your GitHub repository.
        1. For example, mine would be https://github.com/itmd4515/itmd4515-s24-lab1-sspyriso.git
    2. Enter your GitHub Username, but for the Password paste your **Personal Access Token**!
    3. Select master -> master as the local branch, and master -> origin/master as the local reference
    4. Note, this set of steps has also executed a git push for you, to the remote repository you have just configured
3. Browse your GitHub repository - especially if you are new to source control or Git.

4. Document this with a screenshot of the populated repository (Source view, after you push)

Continue to perform git push operations to your remote throughout your development process. I want to see multiple commits in your repository, with commit messages. You can do this via **Team -> Remote -> Push**, only use the repository you configured above, as opposed to setting up a new one each time.

*Important - as you edit your README file in GitHub, you will need to **Pull** those changes down to your local repository before you **Push** any new changes.*

Introductory Maven Web Application

Create an introductory web page (or set of web pages) to introduce yourself to me. This can be as simple as an electronic version of your resume, or as complex as you choose to make it. There are no technical requirements associated with the web page(s). My goal is the following:

1. Ensure your setup is fully functional, including iterative pushes to GitHub
2. Verify that everyone has basic HTML and CSS skills
3. I want to get to know you a little better

You are under no obligation to include personal information on this web page. As I said above, please consider resume content as you decide what to write. You can add additional HTML (or other) files. As you update HTML content, just "Run" your project. It should automatically deploy within the Payara server and start your default browser to display the HTML content.

Maven project setup will be reviewed during class.

Document this with a screenshot on your readme file. The screenshot should display your browser after the application is running in Payara.

Project Submission

1. Submit to Canvas or Beacon
    1. Right your uid-lab1 project and select "Clean"
    2. Go to your NetBeans Projects directory. Create a zip file of the uid-lab1 folder and submit it to the Canvas or Beacon assignment.