VSCode Lab Book

Table of Contents

Trying out the Environments

1. Open the Maven Training Repository link
2. Look around
3. Open either on your machine or on gitpod or codespaces, or both!
4. Let's weigh the pros and cons

Adding Your Themes

1. Add and Choose your Theme
2. Discuss which ones you like

Using Maven in VSCode

1. Reopen your Maven session
2. Run some maven commands in either the terminal or in the tool windows

Using Gradle in VSCode

1. Open the Gradle Training Repository link
2. Look around

3. Open either on your machine or on gitpod or codespaces
4. Run some gradle commands in either the terminal or in the tool windows

Using Bazel in VSCode

1. Open the Bazel Training Repository link
2. Look around
3. Open either on your machine or on gitpod or codespaces
4. Run some of the commands as seen in the *README.md* page:

```
$ bazel query "//src/main/java/com/xyzcorp:*"
$ bazel build "//src/main/java/com/xyzcorp:vs_code_bazel_runner"
$ bazel query "//src/test/java/com/xyzcorp:*"
$ bazel test "//src/test/java/com/xyzcorp:vs_code_bazel_tests"
```

5. Let's also use the *.bazelproject* to add and remove modules

Refactoring

1. Use a template from Ted Young's Blackjack Website
2. Open on Gitpod, Codespaces, or Local
3. Let's try some refactoring techniques!

Running Tests

1. In this lab we will try out Test Driven Development by adding a test for our Wallet.
2. We we create a balance method that will return the balance of the wallet
3. View the reports if successful

Opening Pet Clinic in VSCode

1. Go to the PetClinic Repository
2. Click on the Gitpod or Codespaces Link
3. Choose Maven for the Build Tool
4. Install the Spring Boot Extension Pack
5. Navigate the project
6. Hit `F5` to run the project and open in the browser

Run an HTTP Editing

1. Install the Extension, REST Client Extension by Huachao Mao
2. In any of the java projects, create a file in *src/main/resources* called http_example.http
3. Run the following content:

```
GET https://postman-echo.com/get
Content-Type: application/json
```

4. Try some other commands. Use the reference from their website
5. Note, that you can use # to separate HTTP calls in the same document
6. Note, that payloads must have a carriage return after the headers

Docker Compose

1. Open nfjs_kafka-half-day in Gitpod.io
2. A docker-compose up will already be running
3. Run a docker compose up for container `web` only
4. Open and expose the ports for 9091 and 8000
5. Let's follow the Producer example to produce information to kafka

SQL Tools

1. Install the SQL Tools Extension by Matheus Teixerira
2. Install the SQLTools MySQL/MariaDB/TiDB Extension by Matheus Teixerira
3. Open a terminal and run: `docker run -p 3313:3306 -d sakiladb/mysql:latest`.
4. Open the SQL Tools Window
5. Click *Add Connection*
6. In *Connection Name* enter `sakila`
7. In *Database* enter `sakila`
8. In *Username* enter `sakila`
9. In *Password Mode* select, `Save as plaintext in settings`
10. In *Password* enter `p_ssW0rd`
11. Click Save Connection
12. Run the following: `SELECT * FROM actor LIMIT 50;`
13. Create a file in the *src/main/resources* directory

Version Control

1. Let’s setup teams where we collaborate on a project
2. Fork any of the projects we used: Maven, Gradle, or Bazel
3. Add more features to `Wallet`.
4. Practice performing commits, pushes, and pulls
5. Force and orchestrate a conflict