Part Four: Setting Up

NP AL 创建者：Nick Patrikeos
由 Amanda Lu 最后更新于 3月 03, 2023 • 49 人已查看

This page contains instructions for setting up and working locally.

Prerequisites

- JDK11
  - JDK12+ aren't supported by the course, but *should* work in the assignment-ii build scripts we've setup, given that you don't use any of the new features
- VSCode (Recommended) or IntelliJ
- Windows/MacOS/Linux
  - CSE is supported through VLAB
  - VSCode + SSH aren't supported
  - VSCode + WSL aren't explicitly supported, but the setup we use should work for it, however there is no `xdg-open` command in WSL (but it exists in linux) so you'll have to open the browser manually to the url `localhost:4568/app/` the `app/` part is important.
- Your assignment-ii repository (which we'll presume you'll be okay with cloning and managing)

How to open?

- You can open vscode and then go File -> Open (/Folder on MacOS) and select the folder `assignment-ii` that you've cloned locally.
  - *Make* sure you have just the assignment-ii opened.
- To open in VSCode:
  - `code assignment-ii` (if working locally)
  - `2511 code assignment-ii` (if working on CSE machines)

How to run?

Firstly guarantee that the folder that you have opened is correct, you can see the default hierachy below, ensure that the name at the top says `assignment-ii` if it doesn't you don't have the right folder opened :).

Next, just locate the `App.java` file and click the run button on that file.

```
Run | Debug
public static void main(String[] args) throws Exception {
    Scintilla.initialize();
    GsonBuilder gsonBuilder = new GsonBuilder();
```

This will then synchronise your frontend with the current latest version and once that's done (should be very quick) it'll start the server.

How to generate coverage reports?

You can generate coverage reports through the use of `gradle test`. This will generate both human readable ones (html) as well as more computer readable ones (xml) which an extension can read and show you inline coverage.

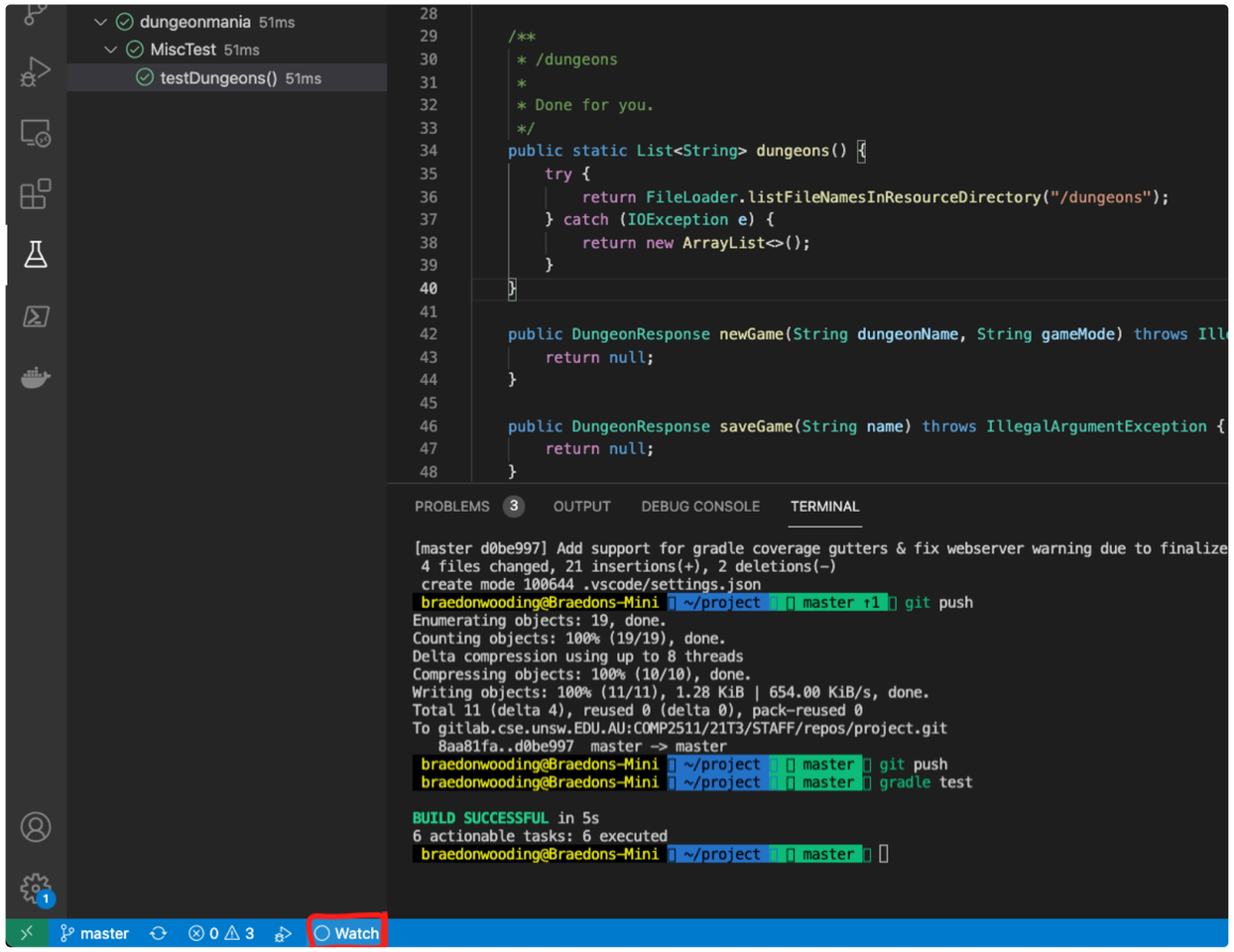
You can see inline coverage reports via; Coverage Gutters which is a VSCode extension. An example of this is shown below.

```
28
29      /**
30       * /dungeons
31       *
32       * Done for you.
33       */
34      public static List<String> dungeons() {
35          try {
36              return FileLoader.listFileNamesInResourceDirectory("/dungeons");
37          } catch (IOException e) {
38              return new ArrayList<>();
39          }
40      }
41
42      public DungeonResponse newGame(String dungeonName, String gameMode) throws IllegalArgumentException {
43          return null;
44      }
45
46      public DungeonResponse saveGame(String name) throws IllegalArgumentException {
47          return null;
48      }
```

> isn't that pretty :D

If they aren't showing up for you make sure that the bottom left "Coverage" button is showing something

If they aren't showing up for you make sure that the bottom left "Coverage" button is showing something like `X% coverage` (just informs about current opened file). If it isn't just click it!



How to test?

You can either run `gradle test` or a bit more nicely go to the test file and click the run test button that'll appear to the left of the icon.

> How to debug tests? You just have to right click and then debug test.

Resources

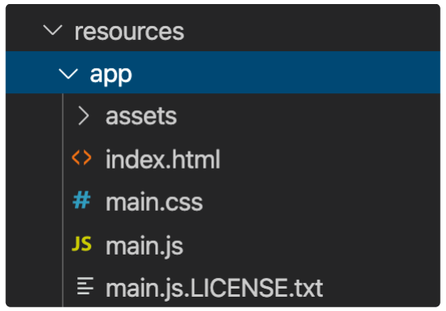
For running the frontend locally, you will need to place your dungeons and configurations inside

For running the frontend locally, you will need to place your dungeons and configurations inside `src/main/resources/`. If you want your resources to be accessible via **Gradle and your JUnit tests** then you will need to put them in `src/test/resources`.

The `FileLoader` class we have provided you with will load the resources from the correct directory automatically.

Recompiling the Frontend Locally

The frontend is simply some compiled JavaScript served as a static resource out of the `main/resources/app` folder.

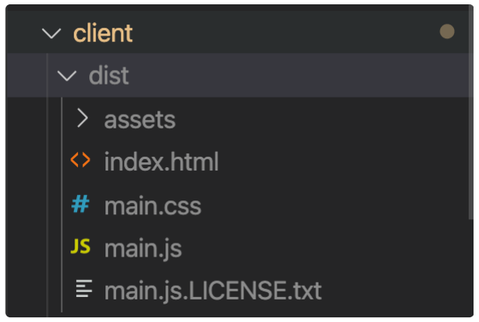


You will need to install `Node.js` if you do not already have it installed on your machine.

To recompile the frontend:

```
cd client # Go into the client directory
npm install # Install required libraries
npm run build # Compile the frontend code
```

You will see a folder called `dist` (short for distribution, since this is what you'd use to deploy on a service like AWS) has now appeared inside `client` which has the exact same structure as `resources/app`:



Copy and paste the contents of `dist` into `resources/app`, overwriting everything that was there before.

When you next `Run` the frontend locally it should include your changes

When you next "kill" the frontend locally it should include your changes.

成为第一个添加回复的用户