Debugging App from VSCode

The "vscode-chrome-debug" extension allows you to attach VSCode to a running instance of chrome. This makes it very convenient because you can put breakpoints in your TypeScript code and run the debugger to debug your app. Let's get started.

- In order to install the extension open the prompt in VSCode with `command + shift + p` and type:

  ```
  > install extension
  ```

  hit enter and then type:

  ```
  debugger for chrome
  ```

  Then just click on the result to install the extension. Restart VSCode when you are prompted.

- After installing the extension, we need to update or create a `launch.json` file for debugging. You can create one in the `.vscode` folder. After you created the file, put in the following:

  ```
  {
    "version": "0.1.0",
    "configurations": [
      {
        "name": "Launch Chrome Debugger",
        "type": "chrome",
        "request": "launch",
        "url": "http://localhost:8080",
        "sourceMaps": true,
        "webRoot": ".",
        "runtimeExecutable": "/Applications/Google Chrome.app/Contents/M
        "runtimeArgs": ["--remote-debugging-port=9222", "--incognito"]
      }
    ]
  }
  ```

Depending on your platform you need to change the `runtimeExecutable` path to Chrome's executable path. After configuring the debugger you need to have a server running serving the app. You can change the `url` value accordingly. Also make sure that the `webRoot` path is set to the root of your web server.

- After that it is a good idea to close all the instances of chrome. Then, put a breakpoint in your code and run the debugger. If everything is set up correctly, you should see an instance of chrome running in incognito mode. To trigger the breakpoint, just reload the page and you should be able to see the debugger paused at the breakpoint.
- Also make sure that you have the compiler running so that you can use the JavaScript output and the sourcemaps to use the debugger. See the TypeScript and VSCode set up for more details.