# Module 3 Project 1

Michael Grandori

Arizona State University

IFT: 458

Professor Dinesh Sthapit

September 20, 2022

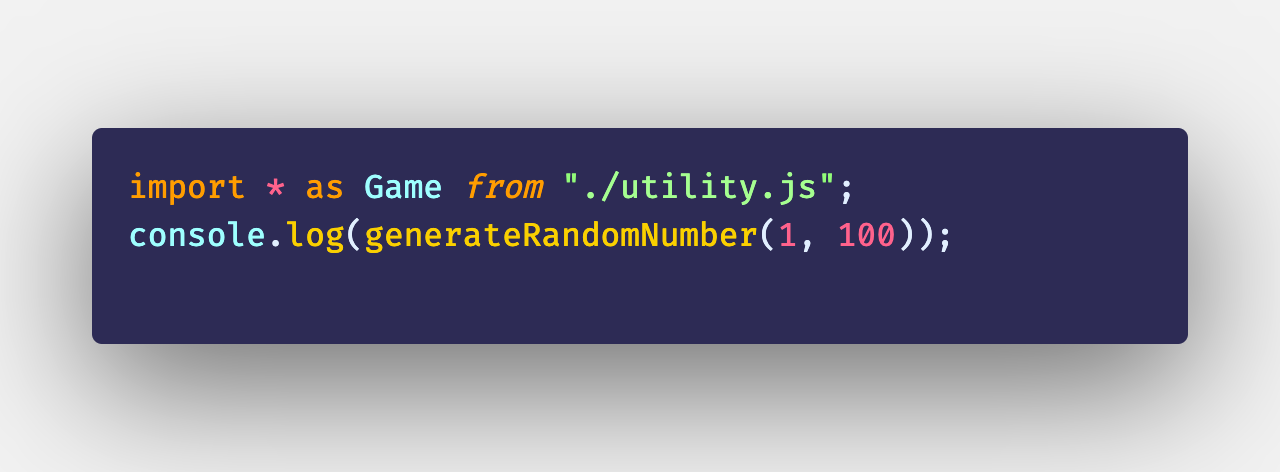
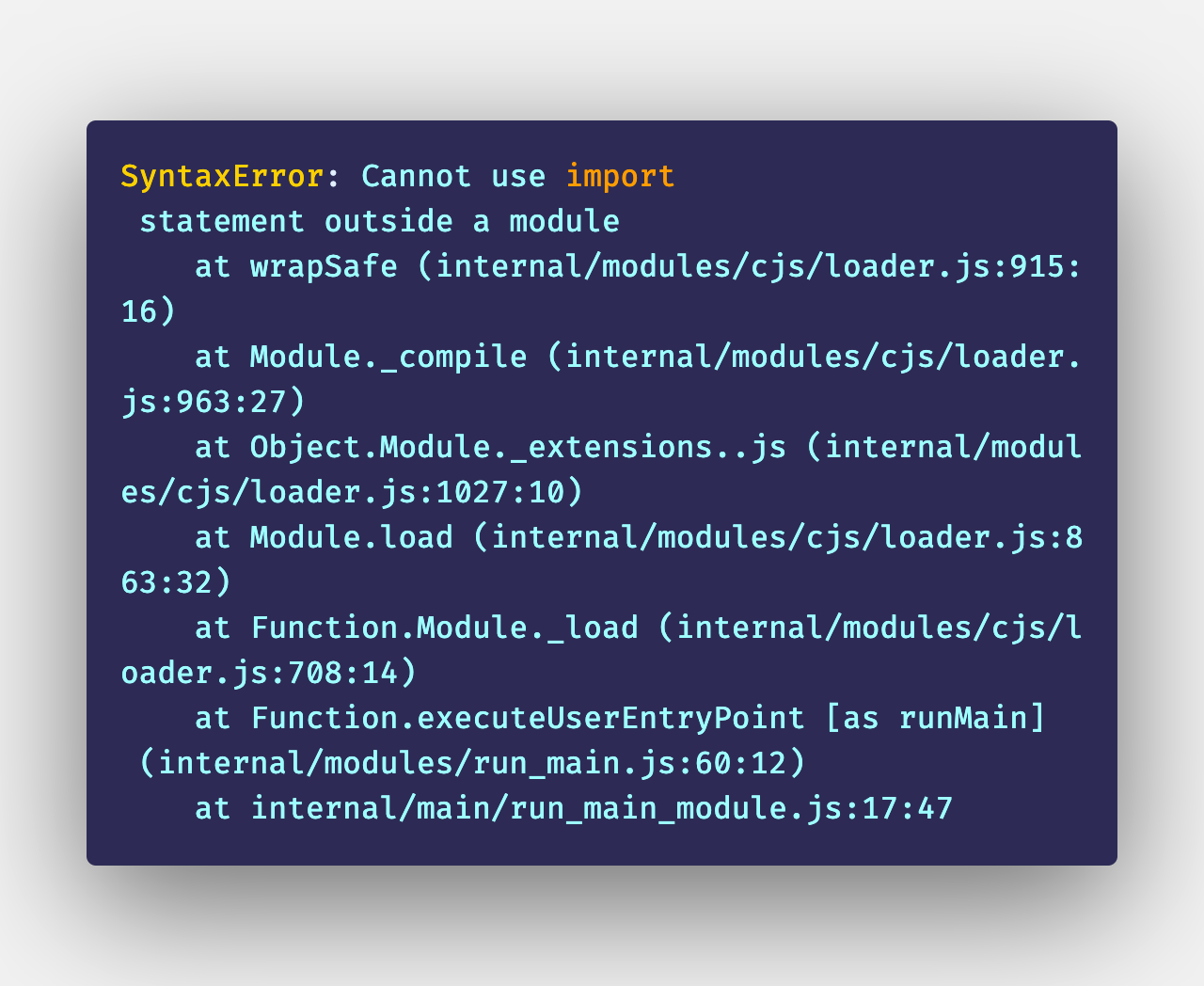
In this assignment, we focused on how imports and exports work in relation to a package.json file in a Node.js project. This is important since learning the steps that are executed when running a file and how the entry point of that file is acted upon is critical to truly understand the application you are building. It also is important to understand how modules are imported and exported throughout a file to avoid module loading errors.

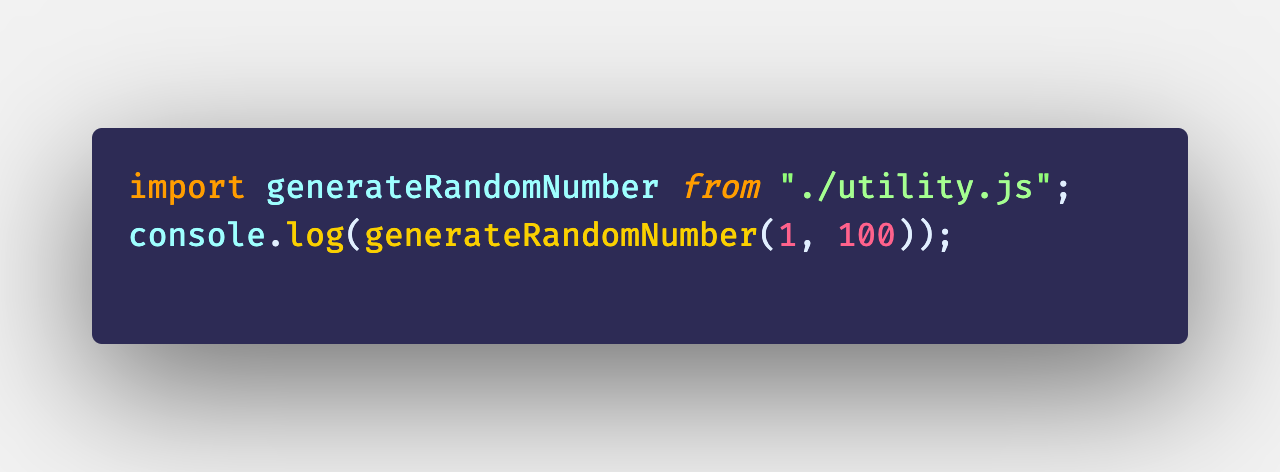
We first created a simple function that will generate a number between two provided argument numbers. This random function will work file by logging it in the file it is written. If, however, we want to use that function in another file, the new file will have no reference to it. We can use it by using the `import` and `export` keywords. By exporting the function in the module, we can re import it into another file. We did this by using `import \* as Game` with the relative path associated with the module that contained the function. This was an important part, but we also need to specify the type within the package.json. By specifying `type: module` inside of the package.json, we can now run the new module and have it import the existing module first, which gives the new module reference to our random function.

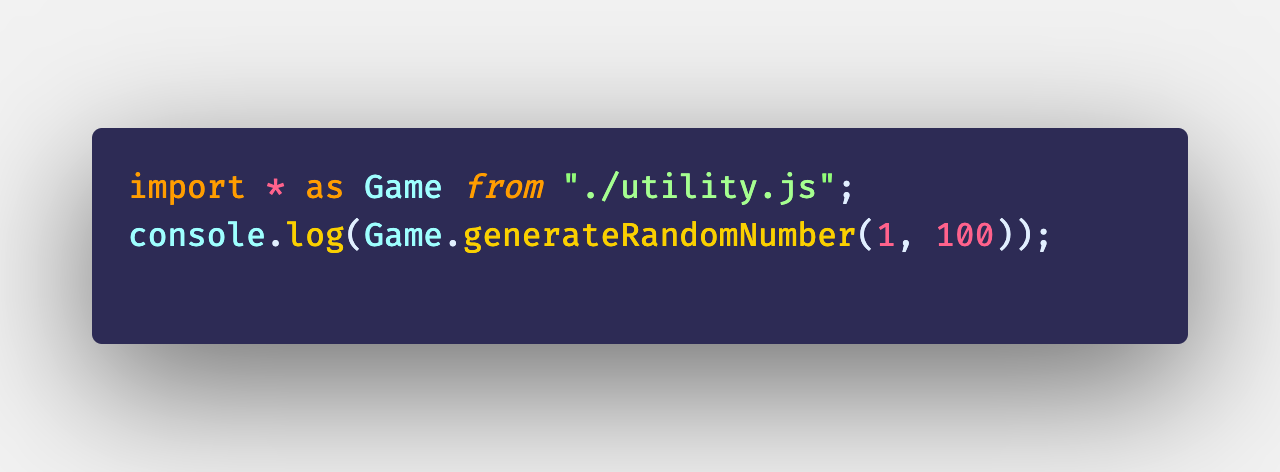
Next, we went over some different serialization and data types that we can consume from APIs. Amongst these are JSON, CSV, and XML. In the public API that we were using, a query parameter called ‘format’ could be appended onto the end of the path to specify which data type we were requesting. The server receiving the request can read this query param and then respond with the specified type if it is supported.

JSON is the most common used serialization and deserialization data type used, and therefore we opted to create a fetch function that would return a JSON response and then deserialize it into a JavaScript object. We were then able to consume that data in our application.

**Screenshots**

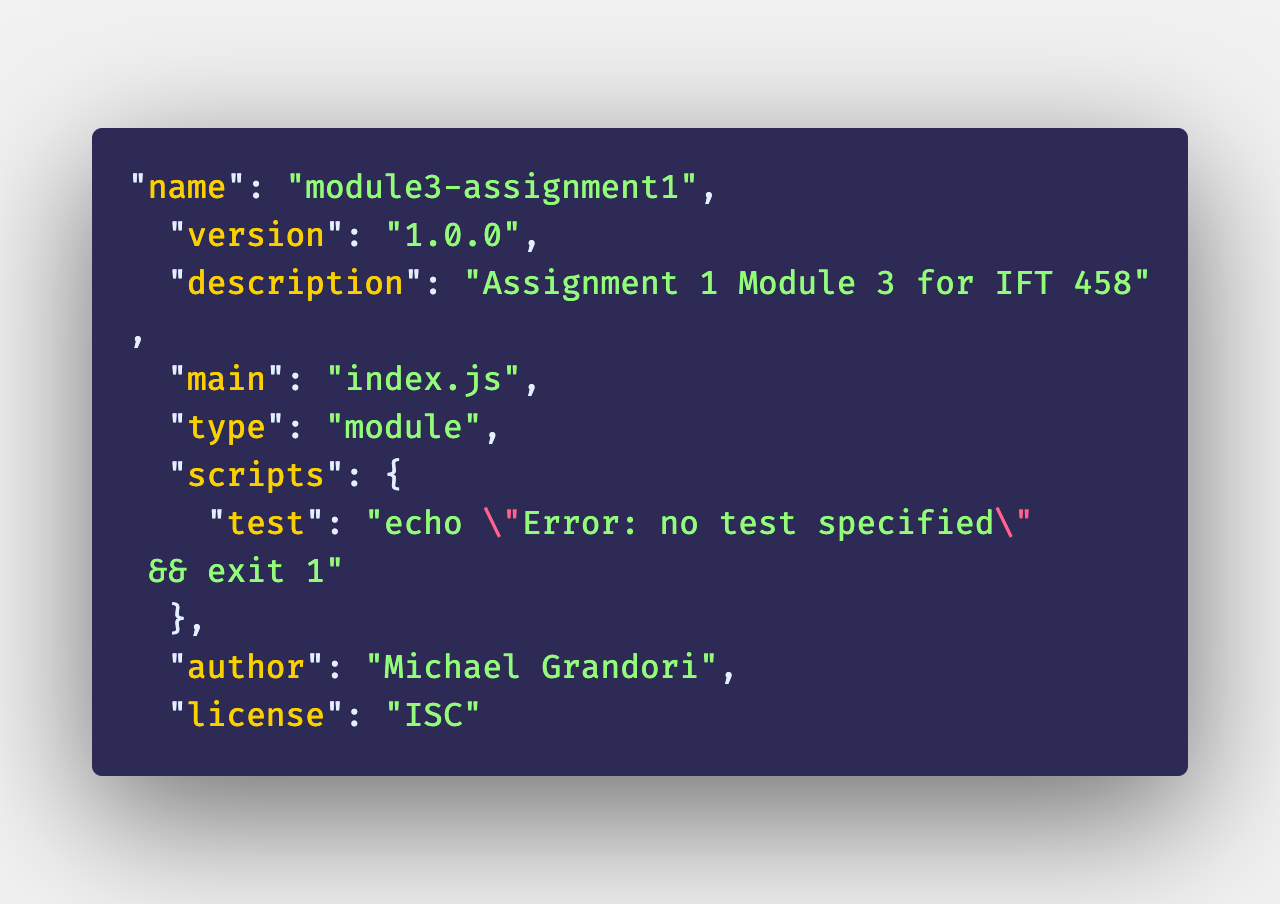
****

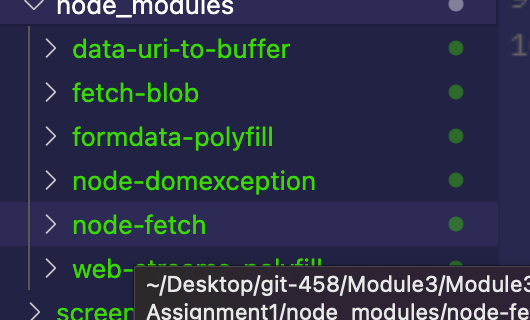
****

****

****

****

****

****

