**IS312 Web Design – TypeScript/JavaScript**

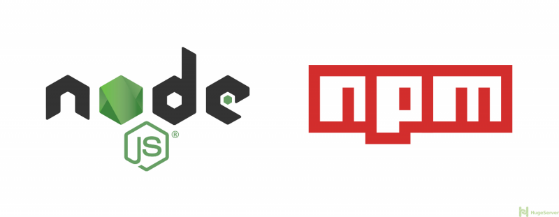
**HOP01B TypeScript Values, Types, Operators, and Program Structure**

8/26/2019 Developed by Kevin Kuanting Chen

5/18/2020 Updated by Amrutha

10/03/2020 Revised Kim Nguyen

Center for Information Assurance (CIAE) @City University of Seattle (CityU)

**Before You Start**

* The directory path shown in screenshots may be different from yours.
* Some steps are not explained in the tutorial**.** If you are not sure what to do:
  1. Consult the resources listed below.
  2. If you cannot solve the problem after a few tries, ask a TA for help.

**Learning Outcomes**

Students will be able to:

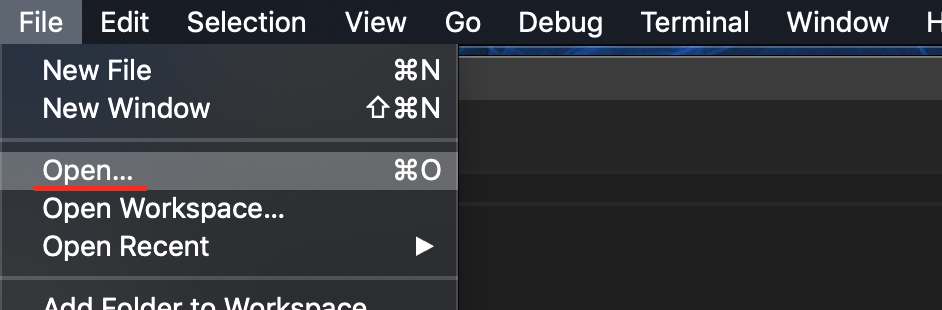
* Understand different values, types, operators, and program structure in Typescript

**Resources**

* [W3School Data Types](https://www.w3schools.com/js/js_datatypes.asp)

**Preparation**

1. Open the VS Code and open the repository you cloned from Github, if you have not cloned, go back to the Github repository generated when you accept the HOP assignment, read the instruction on how to clone the repository, before proceeding.

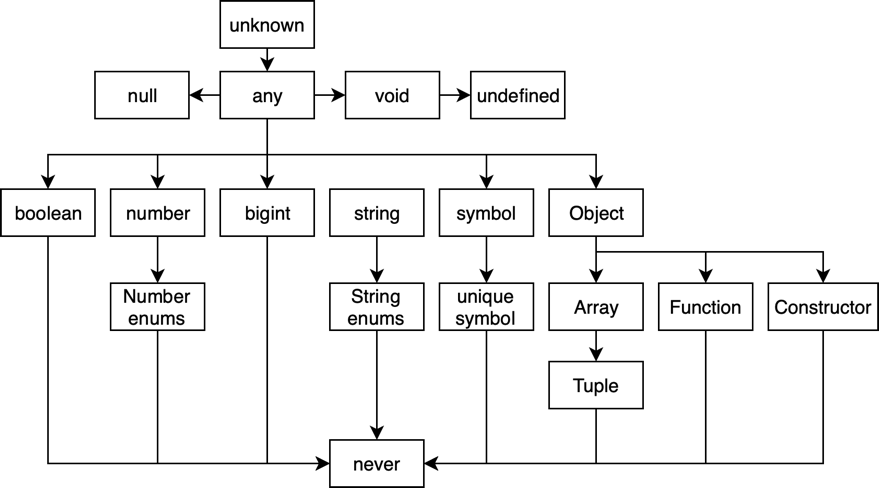


1. Open the terminal from the VSCode by hitting the control + ~ key, check your current directory using the following command, if you are in Module 1, you are in the right place:

>>> pwd

**Data Types and TypeScript Syntax**

Almost all TypeScript data types inherit from *any*. However, you want to avoid using *any* so that the typechecker can help you typecheck your variables and save you from undesired variable manipulation. The following figure shows the type hierarchy of TypeScript.



*Figure 1.* TypeScript’s type hierarchy. Adapted from *Programming TypeScript* (Kindle Locations 452-453), by B. Cherny, 2019. Sebastopol, CA: O'Reilly Media.

1. Create a **hello\_world.ts** file under the Module1 and type the following code in the file.



**Note:** The semicolon is not mandatory. The code will work without semicolon too.

1. In the VS code terminal, compile your TypeScript code into a JavaScript file. This command will create a JavaScript file under the **Module1**.

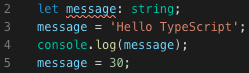
>>> tsc hello\_world.ts

1. Run the Javascript file with node.

>>> node hello\_world.js

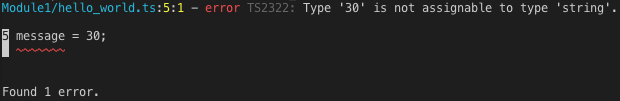


1. Add the following code into the **hello\_world.ts** file.

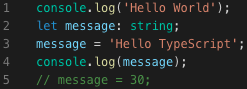


1. Compile your TypeScript code into a JavaScript file. **You will get an error message.** Read the error message and think about why you get it. [1] (With VS Code, you can easily get hints by moving your mouse to the error)

>>> tsc hello\_world.ts



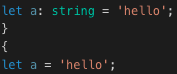
1. Comment the line that caused the problem.

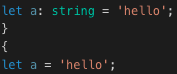


1. Add two more lines to the file.

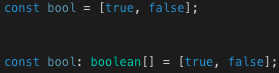


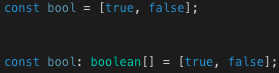
Note: If you initialize a variable when declaring it, you don't have to declare its data type again. Also, use "const" when you are not planning to alter the variable in the future. For example, the following two lines of code are exactly the same (TypeScript will give a string type to the variable “a” based on the value you assign to it).





The following two lines of code are also the same.





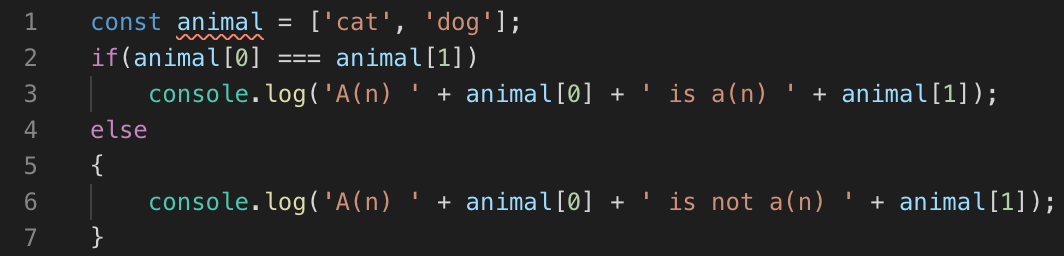
1. Compile your TypeScript code and run the JavaScript file.

>>> tsc hello\_world.ts & node hello\_world.js



**Branching and Looping**

1. Create a file called **branching\_looping.ts** in this module’s folder.



Note: You need curly brackets for blocks (loops, branches, functions, ...), but if it only contains one line of code, you can omit it. The preceding picture has both types as an example.

1. Compile your TypeScript code and run the JavaScript file.

>>> tsc branching\_looping.ts

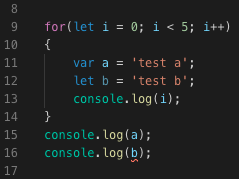
>>> node branching\_looping.js

Try doing

>>> tsc branching\_looping.ts & node branching\_looping.js

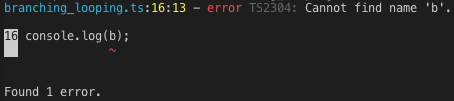


1. Add the following code to the **branching\_looping.ts** file.

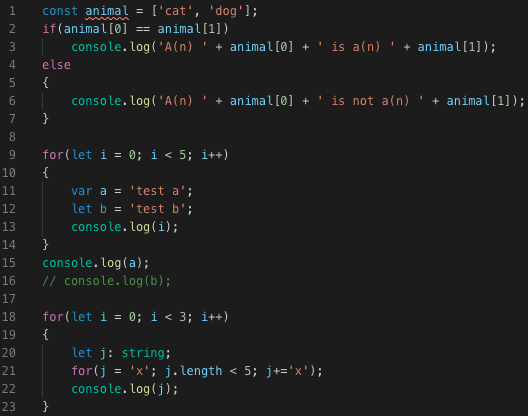


1. Compile the file. **You will get an error message.** Why do you think it causes the error?

>>> tsc branching\_looping.ts

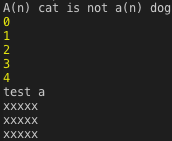


1. Comment the line that causes the trouble and finish the file by matching your script with the following picture.



1. Compile your TypeScript code and run the JavaScript file.

>>> tsc branching\_looping.ts & node branching\_looping.js



**Explanations**

[1] TypeScript Compiler helps you typecheck your code, which means it verifies that your code is typesafe. It keeps you from changing the datatypes accidentally so your program can work as you expect.

[2] Declarations with keywords “let” and “const” are scoped locally to the block. On the other hand, the keyword “var” scopes the variable to the whole function (or the whole script if they are not declared in a function).

**Push your work to GitHub**

Run the following commands to push your work to the GitHub repository:

Open the terminal from the VSCode by hitting the control + ~ key and type the following command:

>>> git add .

>>> git commit -m “Submission for Module 1”

>>> git push origin master