How To Combine Types In Your Node.js Server

Tamar Twena-Stern

Tamar Twena-Stern

- Software Engineer - manager and architect
- Architect @PaloAltoNetworks
- Was a CTO of my own startup
- Passionate about Node.js !
- Twitter: **@SternTwena**

Tamar Twena-Stern

- On Maternity Leave
- Have 3 kids
- Loves to play my violin
- Javascript Israel community leader

Strongly Typed Languages

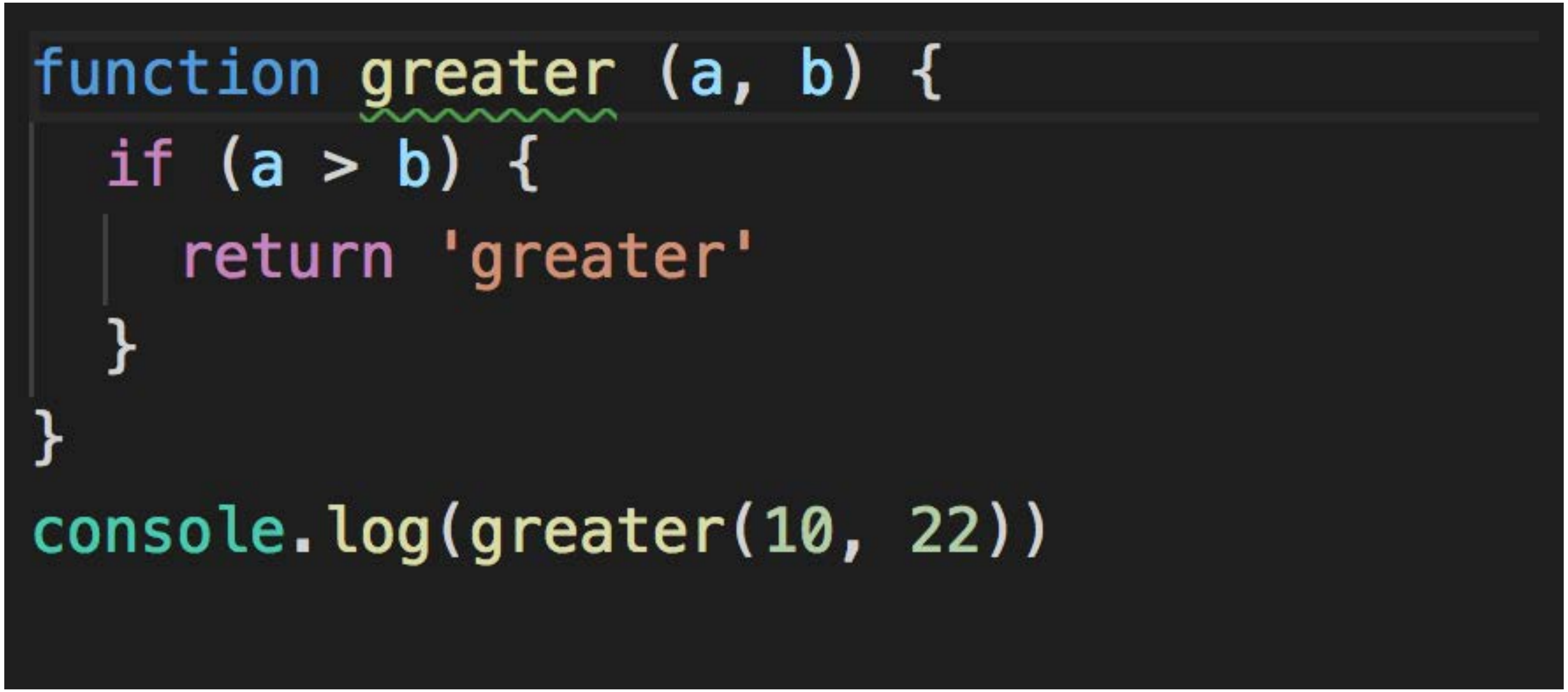
- Use of programming language types in order to
  - Capture invariants of the code
  - Ensure its correctness
  - Definitely exclude certain classes of programming errors.
- Stricter typing rules at compile time
  - Errors and exceptions are more likely to happen during compilation.
  - Rules affect variable assignment, return values and function calling.

JavaScript - Dynamically Typed language

- No typing declarations
- Runtime error checking
- No types information
- No return statement

Problems

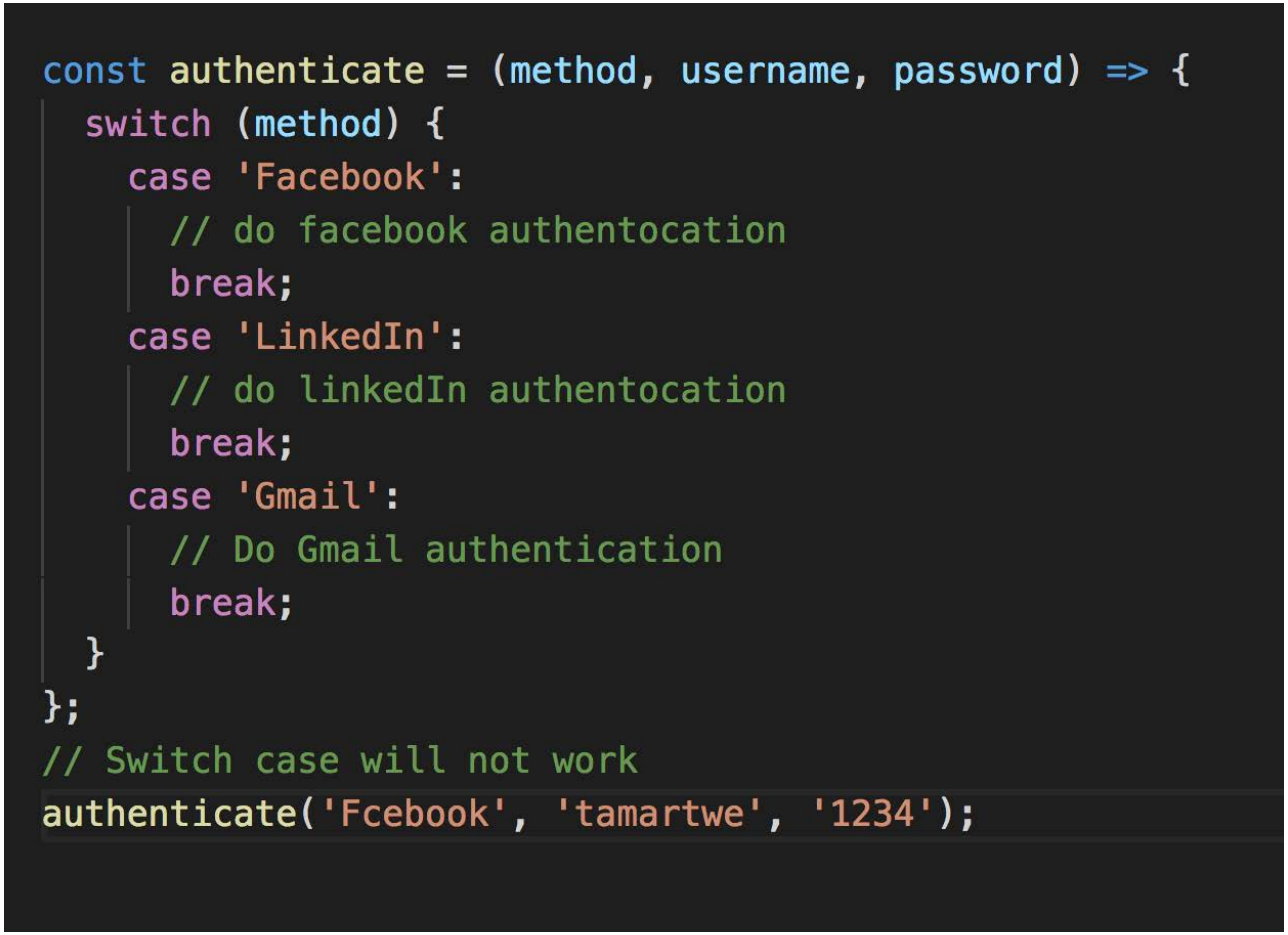
Function Return



```
function greater (a, b) {
  if (a > b) {
    return 'greater'
  }
}
console.log(greater(10, 22))
```

Lets Look At Read File

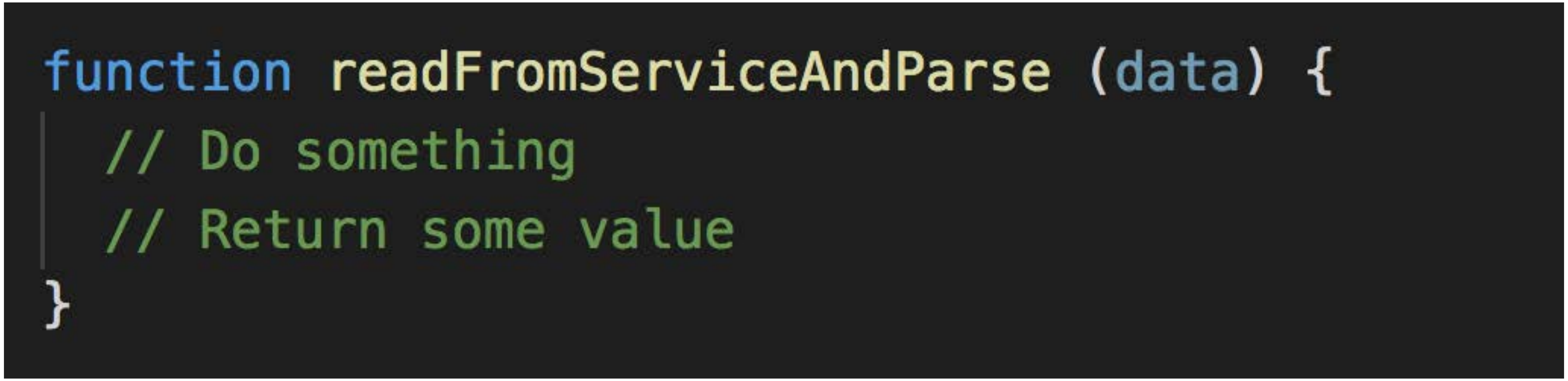
Third Party Authentication With Enum



```
const authenticate = (method, username, password) => {
  switch (method) {
    case 'Facebook':
      // do facebook authentocation
      break;
    case 'LinkedIn':
      // do linkedIn authentocation
      break;
    case 'Gmail':
      // Do Gmail authentication
      break;
  }
};
// Switch case will not work
authenticate('Fcebook', 'tamartwe', '1234');
```

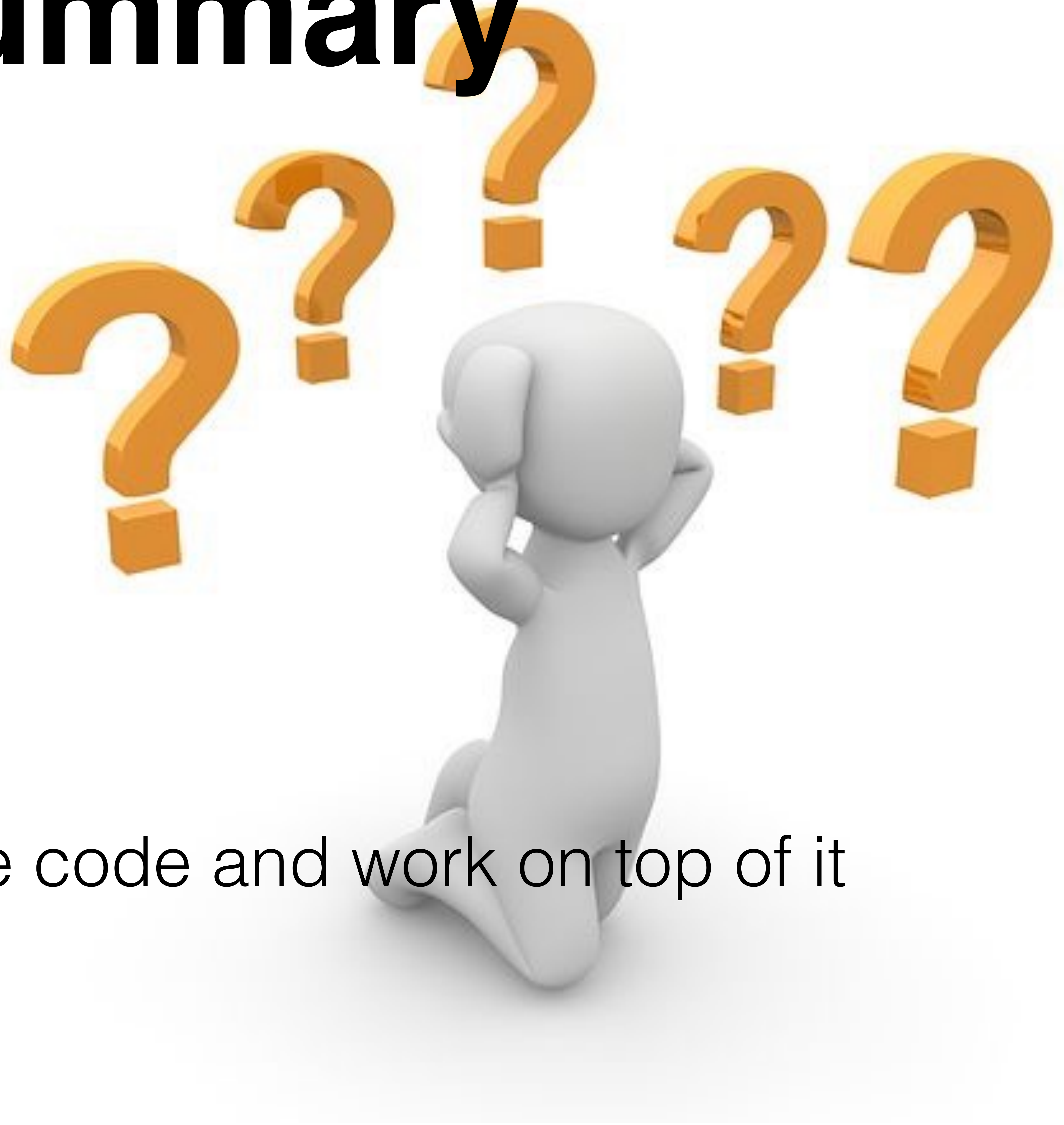
Demo - Simple Function Call, No Types

Can You Use This API ?



```
function readFromServiceAndParse (data) {
  // Do something
  // Return some value
}
```

Problem Summary



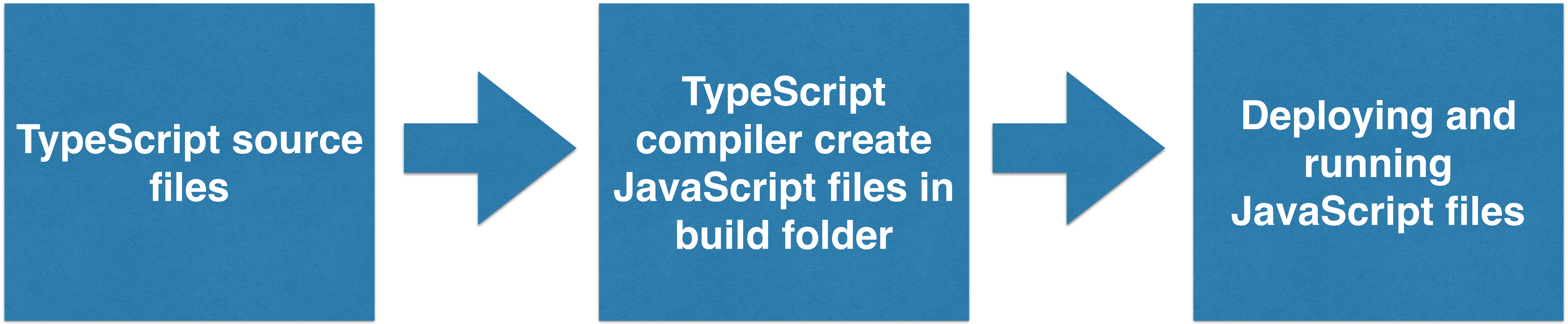
- Hard to use existing APIs
- Inconsistent types in your code
- Type casting is hell
- Hard for other developers to read the code and work on top of it

TypeScript

TypeScript

- Superset of JavaScript
- Transpiles to JavaScript
- Adds optional static typing
- Every JavaScript program is a valid TypeScript program
- Can be used to develop Node.js servers

Transpiles To JavaScript


TypeScript source files
TypeScript compiler create JavaScript files in build folder
Deploying and running JavaScript files

Demo - Simple Node.js TypeScript Server

Working With TypeScript In Node.js

TypeScript Compiler

- Standard Microsoft TypeScript compiler
- Using the command **tsc :**
  - Transpiles all .ts files
  - Create a build folder with JavaScript source files
- Running, debugging and deployment on the build folder

Ts-node

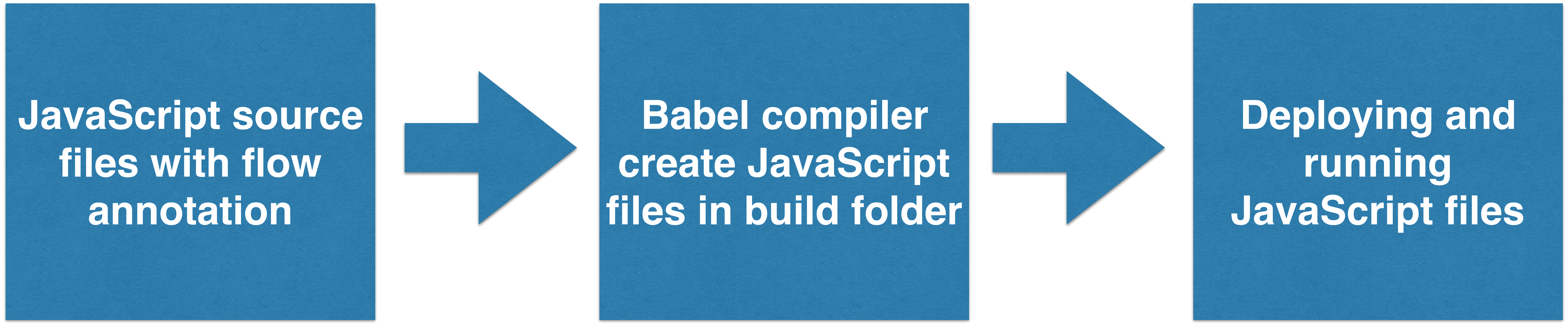
- An executable that we are running
- Registers the TypeScript compiler for the relevant TypeScript file extensions.
- Transpiling relevant extensions on the fly

Flow

Flow

- Open source package
- developed by Facebook
- Static type checker for JavaScript
- Works with babel compiler
- JavaScript files with flow annotations

Flow Compiles With Babel


JavaScript source files with flow annotation
Babel compiler create JavaScript files in build folder
Deploying and running JavaScript files

Simple Node Server With Flow

Lets Vote For Using Types In Node!

Basic Usage

- Annotate your code with types
- Catch errors during “compile” time
- No types = plane JavaScript.

Demo - Basic Usage

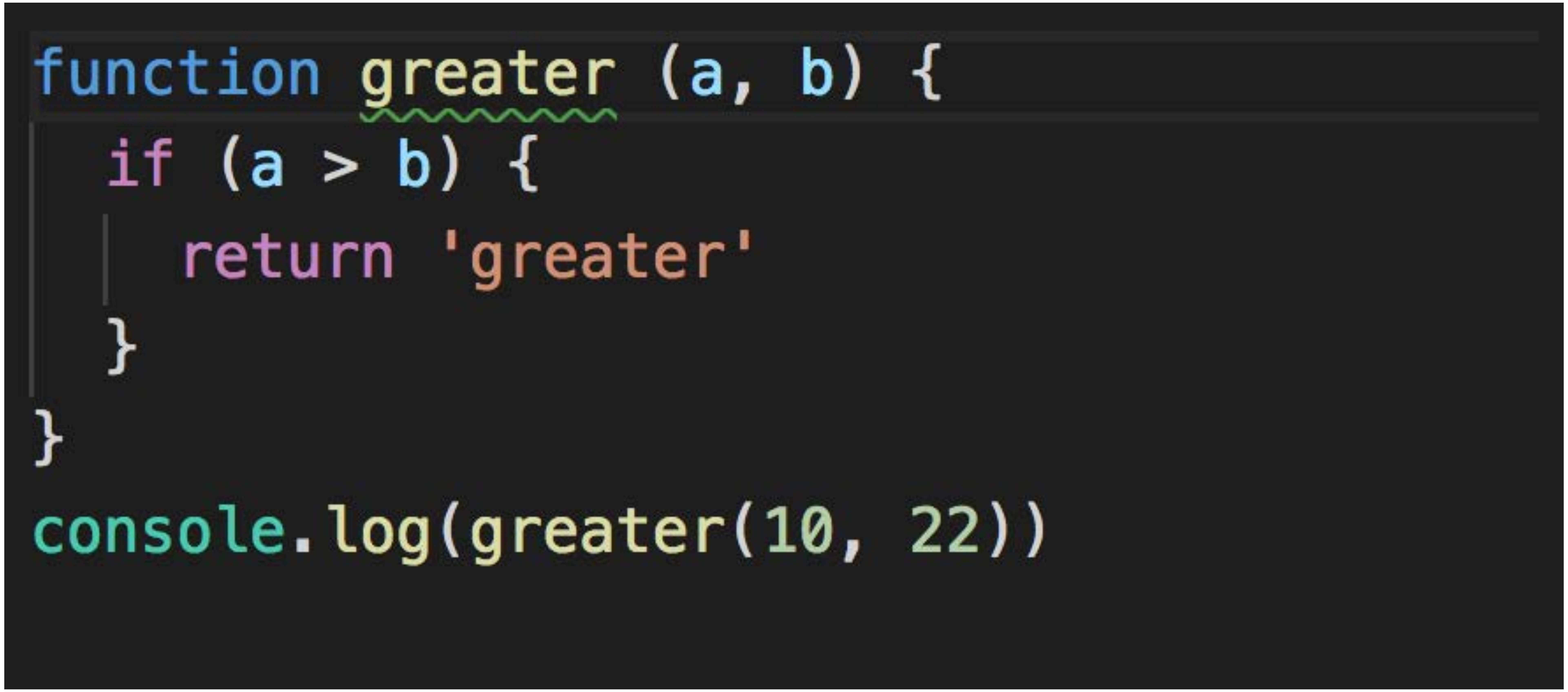
Ease Of Development

- Ability to Relate objects to their type origin in the files
- Ability to auto complete according to the type origin
- Ability to relate type to the file origin
- Speeding up development in complex projects

Demo - Type Relating In The IDE

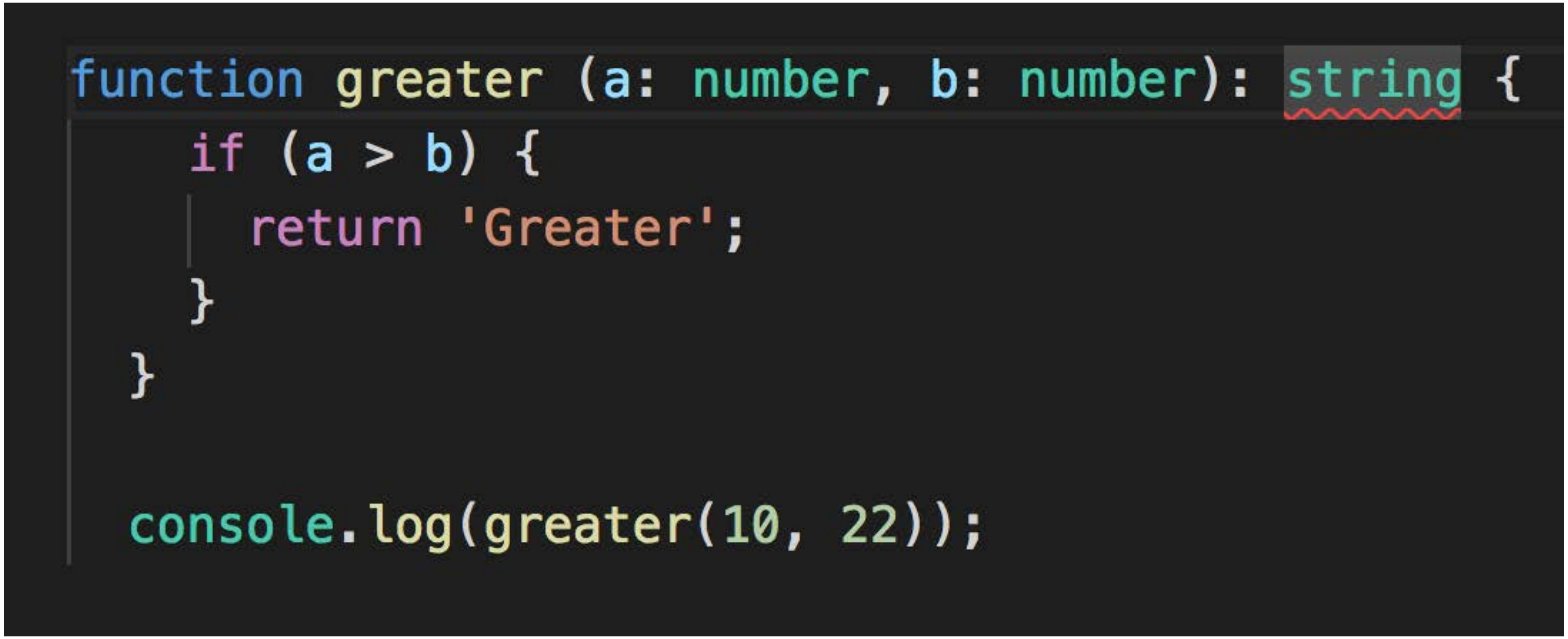
Are You Ready To Go Back To Our Problems ?

Function Return



```
function greater (a, b) {
  if (a > b) {
    return 'greater'
  }
}
console.log(greater(10, 22))
```

Function Return With TypeScript



```
function greater (a: number, b: number): string {
    if (a > b) {
      return 'Greater';
    }
  }

  console.log(greater(10, 22));
```

Read File With Types

Third Party Authentication

Simple Function Call With Types

Now, Lets Talk About The Challenging Parts

Code Transpiler

- Source to source compiler
- Both TypeScript And Flow require using a code transpilar
    - Typescript - tsc - from Typescript To JavaScript
    - Flow - babel - traspiles .js files to .js files with no types
- The code that you run is not the code that you write

Adding A Build Step

- Can slow development
- Every code change requires new build
- Need to work with ‘npm watch’ to synchronise every code change
- Can cause debugging problems in the IDE

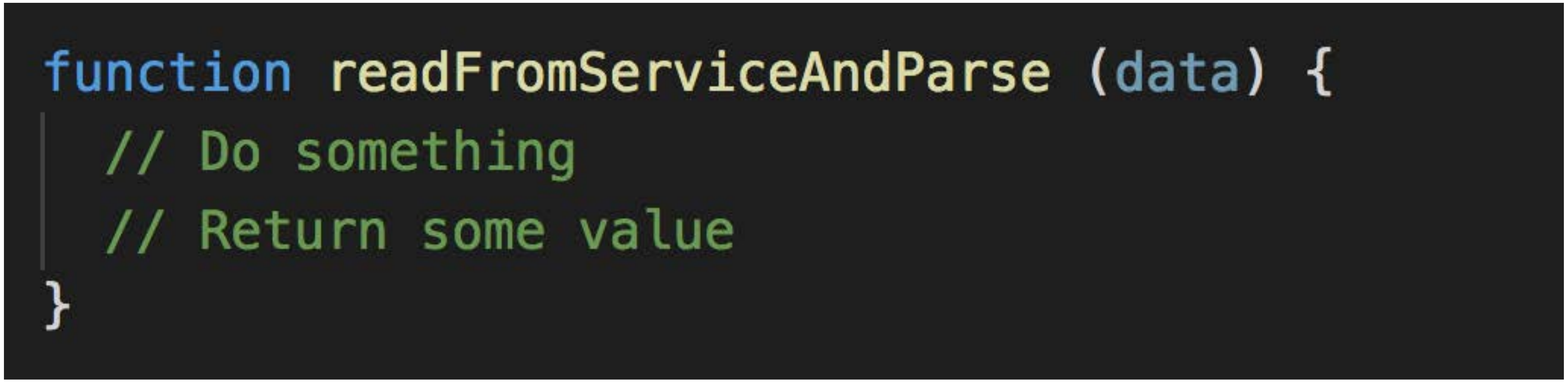
Production - Using Source Maps

- Adding a compilation step -> Source code is different from deployed code
- Stack traces from production will appear on the deployed code
- Without them - Impossible to debug
- Need to use source maps to display stack traces from the original code
- Crucial to analyse bugs on different environments (dev/ prod)

SourceMaps - No Unified Standard

- TypeScript:
  - source-map-support package
  - Add source map support in tsconfig
- Flow + Babel
  - babel-plugin-source-map-support
- Make sure your cloud logging service supports source maps

Remember This ? It Is Still A Problem



```
function readFromServiceAndParse (data) {
  // Do something
  // Return some value
}
```

Tamar Twena-Stern

- Twitter: **@SternTwena**