**[100-days-of-code](https://github.com/AsharibAli/100-days-of-code/tree/main)** / [day-62](https://github.com/AsharibAli/100-days-of-code/tree/main/day-62) / [TS-Intro & Setup](https://github.com/AsharibAli/100-days-of-code/tree/main/day-62/TS-Intro%20%26%20Setup) / **typescript-intro.md**

"TypeScript is JavaScript with syntax for types. It is all about type safety. Sometimes JavaScript exhibits unexpected behavior. For example, if you write:

2 + "2"

It will return '22'. However, with TypeScript, all of these problems just disappear.

TypeScript is a programming language built on top of JavaScript. It enhances JavaScript by offering several features that aid in creating more robust and maintainable applications while reducing development time. The most significant feature TypeScript provides is static typing.

Static typing, a core concept in TypeScript, is what sets it apart. In programming, we have two main categories: statically typed languages and dynamically typed languages.

* **Statically Typed Languages** (e.g., C++, C#, Java) require declaring a variable's type at compile time or during coding. For example, a variable can be declared as an integer, and it can only store integer values. This fixed type prevents the variable from holding a string or any other data type, making the code safer and less prone to errors.
* **Dynamically Typed Languages** (e.g., JavaScript, Python, Ruby) allow variables to have dynamic types, determined at runtime and changeable during execution. While this dynamic nature provides flexibility, it can lead to errors. For instance, changing a variable from a number to a string later in the code may lead to unexpected behavior or crashes.

This is where TypeScript comes in. TypeScript is JavaScript with type checking. It requires developers to explicitly specify variable types upon declaration, similar to statically typed languages. The TypeScript compiler checks your code for type-related errors at compile time, catching many issues before runtime. This helps you identify and resolve errors early in the development process.

You can use TypeScript in a variety of settings, from frontend web development to backend server-side scripting. It seamlessly integrates with JavaScript, allowing you to apply TypeScript's advantages wherever JavaScript is used. Whether you choose TypeScript depends on your project's needs and your development preferences. If you're eager to explore TypeScript further, the next step is setting up your development environment."

But this is not the only reason to learn TypeScript.

## **Why Should You Learn TypeScript?**

1. High Industry Demand: Many companies seek proficiency in TypeScript, making it a valuable skill in the job market.
2. As you know, JavaScript can be as complex as quantum mechanics, but TypeScript simplifies it and makes it more reliable
3. Efficiency: TypeScript reduces the need for extensive unit testing, saving time and effort.
4. Error Prevention and Quick Feedback: TypeScript detects errors before runtime, providing instant feedback for faster and more reliable development.
5. Seamless Integration: It seamlessly integrates with JavaScript, making adoption painless.
6. Framework Empowerment: TypeScript enhances popular frameworks like React, Vue, and Angular with advanced features such as interfaces, generics, and decorators.

## **How Typescript Works**

* TypeScript Compiler Compiles TS Code into JavaScript Code (This Is Called Transpilation).
* TypeScript provides static code analysis, identifying potential errors before runtime. It analyzes your code to offer early error detection, enhancing your development process. It's all about code analysis.

I'll explain this further with an example. While TypeScript can hint at issues, it's still possible to encounter errors or squiggly lines in your code, but you can still export your code to JavaScript.

## **Prerequisite**

Prerequisite to follow along with this: You only need to have a basic knowledge of JavaScript and a cup of coffee.