**Moving from JavaScript to TypeScript –**

[Andrew Baisden](https://andrewbaisden.medium.com/?source=post_page-----262b23d35f15--------------------------------) Feb 10, 2022

**Introduction**

I have been a JavaScript developer for many years and I did not really have much intention to go outside of my technical stack. There is a lot of safety in sticking with what you already know and trying to learn too many programming languages can be daunting I told myself.

JavaScript is already pretty time consuming to learn and nobody truly masters it because the API keeps getting updated along with the documentation as the language evolves. There are also a lot of frameworks and libraries to learn.

It was outdated thinking and fortunately I saw the light when I was between jobs looking for work. Companies were looking for polyglot developers which essentially means a person who knows and is able to use several programming languages.

**Expand your knowledge**

That is when I realized that JavaScript was not enough if you really want to stand out then you need to be capable of using different programming languages. So back then I decided to learn TypeScript and Python. Ironically, I actually managed to get a job but the company only required me to use JavaScript so unfortunately I forgot most of the TypeScript and Python that I learned because I was just not using it on a daily basis.

All of this happened before I was active on tech Twitter and before I started blogging so I really did not understand the concept of building in public and working on side projects. My justification was that I already had a job so I did not need to do programming on weekends too.

**Finding work during the pandemic**

Fast forward to 2021 and everything changed. We were now in our second year of this global pandemic living with Covid. It took me about 6 months to get a decent job offer and I have been working at this company ever since. During this period, I have worked on projects that had a Python and Kotlin backend. So, I was getting exposed to different languages.

JavaScript is still one of the most popular programming languages in the world and is always going to be in high demand. It came top in the [Stackoverflow 2021 Survey](https://insights.stackoverflow.com/survey/2021" \l "technology-most-popular-technologies" \t "_blank) whereas TypeScript is ranked number 7. So if JavaScript is so popular and highly sought after around the world why bother learning TypeScript?

**Why you should learn TypeScript**

As good as JavaScript is the language still has many flaws when compared to other modern programming languages. And unfortunately there are a lot of people out there who just straight up don’t like JavaScript for various reasons.

TypeScript is basically a modern way to develop JavaScript projects and the language compiles to raw JavaScript so your codebase can still be read by a browser and other developers who might not know TypeScript. Honestly the syntax is JavaScript so even if you are not familiar with TypeScript you can still understand what is happening.

TypeScript aims to solve a lot of the problems that JavaScript has which makes the language a lot closer to other modern programming languages. In my opinion anyone who hates JavaScript is likely to fall in love with TypeScript. Or at the very least find less reasons to complain about it.

**JavaScript vs. TypeScript**

There are quite a lot of differences between the two I will cover some of them here.

**Compilation errors**

TypeScript is able to flag errors in compile time during the development process. This is a really good feature because it means that you are less likely to have errors at runtime when your app has been built and is running. JavaScript is only capable of seeing these errors at runtime so you are highly likely to have much slower debugging because you are now doing more unnecessary checking. The better tooling available in TypeScript provides a far better experience when writing code.

**Static typing vs Dynamic typing**

JavaScript uses dynamic typing whereas TypeScript uses static typing. With dynamic typing you can reassign variables because the data type can change. This is not possible with static typing because the data type is defined meaning that if you try to assign a different data type it will show a compile error. There are pros and cons for each method.

// This is valid JavaScript code  
let num = 10;  
num = "10";// You will get the error Type 'string' is not assignable to type 'number'.  
let num: number = 10;  
num = "10";

**Describing your data using an interface**

TypeScript can use an interface in the code which pretty much describes the structure of an object in the application. It defines the overall syntax that is required for the object so you can use it for documentation and issue tracking inside of your code editor.

It is worth noting that the TypeScript compiler wont convert the interface syntax to JavaScript. It is only used for type checking also known as “duck typing” or “structural subtyping”.

// Describe the shape of objects in your code.  
interface Series {  
id: number;  
seriesName: string;  
releaseDate: number;  
}// Use the interface for type checking in your object.  
const series: Series = {  
// The id needs to be a number  
id: 1,  
// The series name needs to be a string  
seriesName: 'The Book of Boba Fett',  
// The release data needs to be a number  
releaseDate: 2021,  
};console.log(series);

**CommonJS modules vs. ES modules**

Node.js uses CommonJS modules by default and anyone who is familiar with it will know about the require syntax. In comparison when you use Node.js with TypeScript you have the option to use either require or import and export statements. Of course there are ways to get it working in native JavaScript too if you do your research.

**JavaScript CommonJS modules**

const express = require('express');

**TypeScript ES modules**

import express from 'express';

When using TypeScript you get access to a tsconfig.json file which lets you change lots of settings which include the target. This lets you set the JavaScript language version for outputted JavaScript files. For example they can be ES2015, ES2016, ES2017 etc...

**TypeScript Drawbacks**

TypeScript is pretty amazing but it does have a few disadvantages that you should be aware of. Firstly TypeScript does not work in the browser so you have to compile your code to JavaScript before you can use it.

Fortunately TypeScript has a compiler so when you have got it setup it will automatically compile your TypeScript files to JavaScript and luckily it is a fast process. So you don’t have to worry about having to wait around for minutes for your code to compile the process is usually done in seconds.

Another disadvantage is the fact that you are going to be writing a bit more code especially if you want to have static type checking. I don’t really see this as a downside though because you are writing more performant and better code which is going to make it more maintainable.

Something else that you need to know is that you will require some Type Declaration packages alongside some of the normal packages that you use. Type Declaration packages describe built-in objects. Declaration files give you a way to declare types or values so there is no need to provide any sort of implementations for the values.

This is not always going to be the case because some packages already have type definitions but not all of them. It is easier to understand in this Express Node.js example below.

**JavaScript Express App**

npm i expressconst express = require('express');const app = express();app.get('/', (req, res) => {res.send('Home Route');});const port = process.env.PORT || 3000;app.listen(port, () => console.log(`Server running on port ${port}, [http://localhost:${port}`));](http://localhost:$%7bport%7d%60));/)

**TypeScript Express App**

npm i express @types/express @types/nodeimport express, { Response, Request } from 'express';const app = express();app.get('/', (req: Request, res: Response) => {res.send('Home Route');});const port = process.env.PORT || 3000;app.listen(port, () => console.log(`Server running on port ${port}, [http://localhost:${port}`));](http://localhost:$%7bport%7d%60));/)

**TypeScript Support**

TypeScript is well supported and if your code editor of choice is Visual Studio Code then TypeScript is treated like a first class citizen because Microsoft developed the code editor and the language.

Pretty much all of the popular JavaScript frameworks support TypeScript too. So that includes React, Angular, Vue and Svelte. The framework express.js also has compatibility with TypeScript as do other Node.js frameworks. So there really is nothing stopping you from using TypeScript on the front-end and back-end of your application.

Another advantage is the fact that you can now use ES modules on the back-end and front-end natively. So if you are creating an app with a Node back-end and a React front-end for example. You can now use import and export statements for both and you don't need to use CommonJS modules require statements anymore.

**How to learn TypeScript**

I learned TypeScript from Scrimba and I also followed another good TypeScript course on Udemy. If you already know JavaScript then it won’t take you long to get up to speed with TypeScript. Also if you are new to JavaScript or still learning the basics then it is better for you to wait until you have more experience with it before learning TypeScript.

[Learn Typescript for free](https://scrimba.com/learn/typescript)  
[Understanding TypeScript — 2022 Edition](https://www.udemy.com/course/understanding-typescript/)