



JavaScript alternative (TypeScript) and its effectiveness in web development

Md Saiful Islam

BACHELOR'S THESIS
May 2023

Degree Programme
Software Engineering

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Software Engineering

Md Saiful Islam
JavaScript alternative (Typescript) and its effectiveness in web development
Bachelor's thesis 32 pages
May 2023

JavaScript is the most popular language used to build websites and web applications. It has some limitations such as Lack of static typing and limited support for object-oriented programming. TypeScript, a language developed by Microsoft, addresses these issues by providing a statically typed, object-oriented alternative to JavaScript. In this thesis, evaluates how effective TypeScript is for his web development by comparing features such as static typing, interfaces, and modules with those of JavaScript. We also explore how TypeScript impacts productivity, code organization, maintainability, and reliability of large-scale web applications. Conduct a case study of a real web development project to compare results using TypeScript and JavaScript.

CONTENTS

| | | |
|-------|--|----|
| 1 | INTRODUCTION | 5 |
| 2 | THEORETICAL BACKGROUND | 6 |
| 2.1 | History and Evolution of Typescript | 6 |
| 2.2 | Key Features, Including Static Typing, Interfaces, and Modules.... | 7 |
| 2.2.1 | Static Typing | 7 |
| 2.2.2 | Interfaces | 8 |
| 2.2.3 | Modules..... | 9 |
| 2.3 | Comparison With JavaScript..... | 10 |
| 2.3.1 | Processing Time Difference | 13 |
| 3 | USE CASES OF TYPESCRIPT | 15 |
| 3.1 | TypeScript in Action..... | 15 |
| 3.2 | Advantages and disadvantages of using TypeScript | 16 |
| 3.3 | Compatibility with Popular Frameworks (Angular and React)..... | 16 |
| 3.3.1 | Angular with TypeScript | 17 |
| 3.3.2 | React with TypeScript..... | 17 |
| 3.4 | Case Studies of Companies & Organizations Using TypeScript .. | 18 |
| 3.4.1 | Microsoft | 19 |
| 3.4.2 | Slack | 19 |
| 3.4.3 | Airbnb..... | 19 |
| 3.4.4 | Asana..... | 19 |
| 4 | IMPACT OF TYPESCRIPT ON WEB DEVELOPMENT | 20 |
| 4.1 | Improved code maintainability and scalability | 20 |
| 4.2 | Early error detection and debugging..... | 20 |
| 4.3 | Improved Productivity (Visual Code) | 21 |
| 4.4 | Limitations and Challenges of TypeScript | 22 |
| 4.5 | Compatibility Issues with Some Libraries and Frameworks..... | 22 |
| 4.6 | Performance Considerations..... | 23 |
| 4.7 | TypeScript Community and Ecosystem..... | 24 |
| 4.8 | The Importance of TypeScript Popularity | 25 |
| 4.8.1 | Stack Overflow..... | 25 |
| 4.8.2 | GitHub | 27 |
| 4.8.3 | Google Trends | 28 |
| 4.9 | Potential Drawbacks..... | 29 |
| 5 | DISCUSSION | 30 |
| | REFERENCES | 31 |

GLOSSARY or ABBREVIATIONS AND TERMS (choose one or other)

| | |
|----|------------|
| JS | JavaScript |
| TS | TypeScript |

1 INTRODUCTION

JavaScript and TypeScript are two of the most widely used programming languages for web development. While JavaScript has been an essential language for web development since its inception, TypeScript has been gaining popularity in recent years as a "superset" of JavaScript that adds optional static typing and other features.

In this paper, we will explore the features and benefits of TypeScript in web development, examine its impact on the development landscape, and evaluate its effectiveness as an alternative to JavaScript in various use cases. We will also look at the current state of the industry and see how the two languages are being used in real-world projects. Additionally, we will discuss the challenges and limitations of using TypeScript and provide recommendations for developers and organizations considering adopting it as their primary language for web development.

Overall, this thesis aims to provide a comprehensive and objective comparison of JavaScript and TypeScript, and to help developers make informed decisions about which language to use for their projects.

2 THEORETICAL BACKGROUND

Typescript is a programming language introduced by Microsoft in 2012. It's an open-source language that adds features like static typing, classes, and interfaces to JavaScript. This makes it easier for developers to write efficient, maintainable, and scalable code. By catching errors at compile time, you can reduce errors and improve code quality. It is highly compatible with existing JavaScript code, supports all JavaScript features, and can be used in any environment that supports JavaScript. A powerful type system allows developers to specify data types for variables, arguments, and return values, making code easier to document and maintain. It also supports object-oriented programming concepts such as classes, interfaces, and inheritance, making it easy to write modular, reusable code(Hiwarale 2020).

2.1 History and Evolution of Typescript

Typescript was first introduced by Microsoft (after two years of development). This was the Microsoft solution for developing applications at scale using JavaScript, for them and their customers. Steve Lucco and his team of over 50 people, including his lead C# architect Anders Hejlsberg, creator of Delphi and Turbo Pascal, developed his TypeScript at Microsoft. This project was originally known as Strada. Initially, products such as Bing and Office 365 created a need for improving JavaScript so that Microsoft could build scalable products(open AI 2023).

Over time, Typescript has evolved and improved, adding new features and improvements with each new version. In 2015 Typescript 1.5 was released adding support for decorators and his ES6 generator. Typescript 1.6, released in 2015, added support for React and JSX. In 2016, Typescript 2.0 was released, introducing features like optional concatenation, null coalescing, and readonly properties. Typescript 2.1, released later this year, added support for async/await functions and rest/spread properties on objects. Typescript 2.3 was released in 2017 and added support for generic parameter default values and conditional types. Typescript 2.4, released later that year, added support for dynamic import

expressions and string enumerations. In 2018 he released TypeScript 3.0, which introduced features like project references, tuple types, read-only arrays and tuples. TypeScript 3.1, released later that year, added support for mappable tuple and array types. In 2019 TypeScript 3.5 was released adding support for ESNext features like private fields and methods, and in 2020 he introduced features like templated string types, labeled tuple elements and unknown catch clauses (Wikipedia 2023).

2.2 Key Features, Including Static Typing, Interfaces, and Modules

TypeScript begins with JavaScript and ends with JavaScript. TypeScript adopts the basic building blocks of our program from JavaScript. That being so, only need to know JavaScript to use TypeScript. All TypeScript code is converted into its JavaScript analogous for the purpose of execution (Tutorialspoint 2023). It provides a powerful and flexible alternative to JavaScript, offering developers a more structured and efficient approach to web development.

2.2.1 Static Typing

Static typing is a key feature of TypeScript that allows developers to declare the data types of variables, function parameters, and return types at compile-time. This helps catch errors early in the development process and makes your code more reliable (Tutorialspoint 2023).



```
1 // Declaring a variable with a specific data type
2 function calculateArea(width: number, height: number): number {
3     return width * height;
4 }
5
6 let area: number = calculateArea(10, 5);
7 console.log(area); // Output: 50
```

FIGURE 1. Code Example of Static Typing

In figure 1, a function `calculateArea` is defined with two parameters, `width` and `height`, both of which are declared as `number` using static typing. The function calculates the area of a rectangle by multiplying the width and height values and returns the result as a `number`.

When calling the `calculateArea` function, the expected argument types are also specified as `number`. This is because the function expects two numeric arguments and returns a numeric value. The result of the function call is assigned to a variable `area` which is also declared as a `number`.

2.2.2 Interfaces

Another important feature of TypeScript is interfaces, which allow developers to define custom types that can be used throughout their code. Interfaces define the structure and requirements of an object, ensuring that it has certain properties and methods. This enables better code documentation, improves code readability, and makes it easier to maintain and refactor code as projects evolve (Tutorialspoint 2023).

```
1 interface Product {
2   id: number;
3   name: string;
4   price: number;
5   description?: string; // Optional property
6   inStock: boolean;
7   getDiscountedPrice: (discount: number) => number; // Method that takes a discount percentage and returns the discounted price
8 }
9
10 const product: Product = {
11   id: 1,
12   name: "Smartphone",
13   price: 1000,
14   inStock: true,
15   getDiscountedPrice: function(discount: number) {
16     return this.price - (this.price * discount) / 100;
17   }
18 };
19
20 console.log(product.getDiscountedPrice(10));
21
```

FIGURE 2. Code Example of Defined an Interface

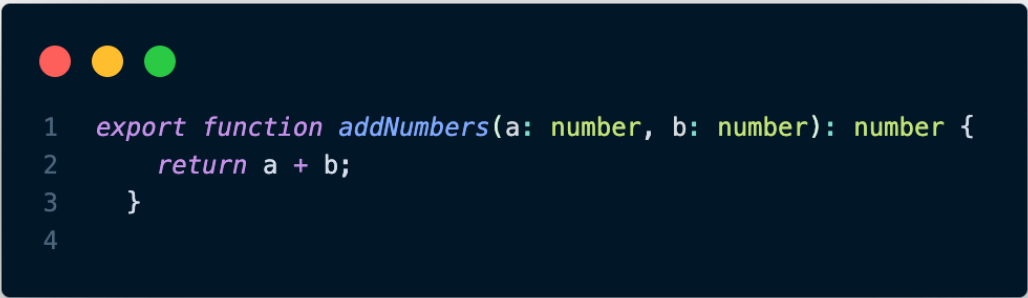
In figure 2, defined an interface `Product` that requires all product objects to have `id`, `name`, `price`, `inStock`, and `getDiscountedPrice` properties. The `description` property is optional, indicated by the question mark "?". Then create

a `product` object that implements the `Product` interface. It has all the required properties and the `getDiscountedPrice` method that takes a discount percentage and returns the discounted price.

2.2.3 Modules

TypeScript also has a modular architecture, allowing developers to use only the features they need, and its syntax is similar to that of JavaScript, making it easy for developers to learn and use. It supports all the features of JavaScript, including the latest ECMAScript standards, and can be used in any environment that supports JavaScript, including web browsers, servers, and mobile devices (Tutorialspoint 2023).

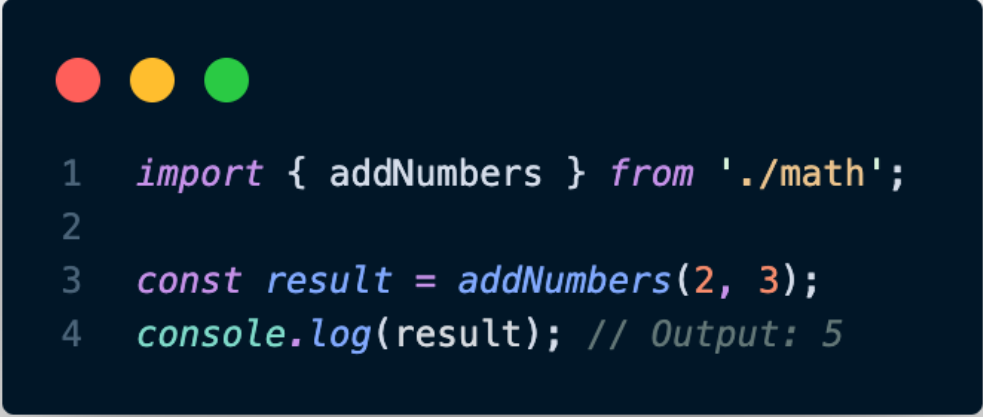
Let's say we have a file called `math.ts` that exports a function for adding two numbers:

A code editor window with a dark background and three colored window control buttons (red, yellow, green) at the top left. The code is as follows:

```
1  export function addNumbers(a: number, b: number): number {  
2      return a + b;  
3  }  
4
```

FIGURE 3. Code Example of Modular Programming Using Ts

Then, in another file called `main.ts`, we can import that function and use it:

A code editor window with a dark background and three colored window control buttons (red, yellow, green) at the top left. The code is as follows:

```
1  import { addNumbers } from './math';  
2  
3  const result = addNumbers(2, 3);  
4  console.log(result); // Output: 5
```

FIGURE 4. Imported Function from Ts File Used in Js File

Using modules in this way can help keep our code organized and make it easier to reuse functionality across multiple files.

2.3 Comparison With JavaScript

TypeScript and JavaScript differ in several key aspects. TypeScript, a superset of JavaScript, provides static typing, advanced features like modules, generics, and interfaces, and detects errors during compile time, making it more suitable for complex projects. On the other hand, JavaScript is a dynamically typed language that is more suitable for simpler web applications due to its lower learning curve, direct browser compatibility, and extensive community support (Krishnan 2022). While TypeScript requires a build setup and a deeper understanding of scripting, JavaScript can be learned on the go and does not require a build setup. The choice between TypeScript and JavaScript depends on project requirements, developer expertise, and the desired level of maintainability and complexity.

TABLE 1: TypeScript vs JavaScript: Key Differences (Raval 2023)

| Feature | TypeScript | JavaScript |
|--|--|---|
| Compile Time Type Checking | Conducts compile-time validation, which reduces runtime overhead | Type verification is performed at runtime |
| Project Size and Development Team | Runs seamlessly for large projects or when many developers are working together | Suitable for small to medium-sized projects or solo development |
| Working with New Libraries or Frameworks | Offers IntelliSense to identify and navigate new interfaces | Offers type definitions, but may require manual configuration |
| Framework Support | May not support certain frameworks, limiting feature usage | Can be used with most frameworks |
| Build Tools | Requires a build step to generate JavaScript code | Can be used without build tools, but it is becoming increasingly uncommon |
| Testing Workflow | Benefits may not justify expenditures if JavaScript developers are already using test-driven development | Can be used with or without test-driven development |

While the differences in this simple example may not seem significant, TypeScript's features become more valuable in larger and more complex applications, where type safety, better error detection, and improved code editor support can significantly enhance the development process.

In figure 5, a code comparison as example of JavaScript and TypeScript code for a quiz app. The example shows how to display a question and its options.

The image shows a side-by-side comparison of JavaScript and TypeScript code in a code editor. The left pane shows the JavaScript version, and the right pane shows the TypeScript version. The TypeScript version includes type annotations for variables and a function interface.

```

JS example.js > ...
1  const questionElement = document.getElementById("
2  const optionsContainer = document.getElementById("
3
4  const question = {
5    text: "What is the capital of France?",
6    options: ["Paris", "London", "Berlin", "Rome"],
7  };
8
9  function displayQuestion() {
10   questionElement.innerHTML = question.text;
11   question.options.forEach(option => {
12     const optionElement = document.createElement(
13       optionElement.innerHTML = option;
14       optionsContainer.appendChild(optionElement);
15   });
16 }
17
18 displayQuestion();
19
TS example.ts > questionT > options
1  const questionElementT = document.getElementById(
2  const optionsContainerT = document.getElementById(
3
4  interface Question {
5    text: string;
6    options: string[];
7  }
8
9  const questionT: Question = {
10   text: "What is the capital of France?",
11   options: ["Paris", "London", "Berlin", "Rome"],
12 };
13
14 function displayQuestionT(): void {
15   questionElementT.innerHTML = question.text;
16   question.options.forEach(option: string => {
17     const optionElement = document.createElement(
18       optionElement.innerHTML = option;
19       optionsContainerT.appendChild(optionElement);
20   });
21 }
22
23 displayQuestionT();
24

```

FIGURE 5. TypeScript vs JavaScript Code Comparison

In Figure 5, we can compare.

Type Annotations: In the TypeScript example, we specify the types for variables and function parameters, which helps catch type-related errors during development and improves code quality.

Interface: Define an interface **Question** to describe the structure of a question object, enhancing readability and making it easier to manage complex data structures.

Type Assertions: Used type assertions (e.g., as `HTMLElement`) to inform TypeScript about the expected type of an element, allowing better type checking and autocomplete in code editors.

2.3.1 Processing Time Difference

When it comes to performance, TypeScript, a compiled language, theoretically runs faster than interpreted languages like JavaScript and Python. However, since TypeScript is compiled into JavaScript, which is an interpreted language, it may not necessarily result in a significant performance boost. TypeScript can still offer performance benefits through its ability to catch errors at compile-time and optimize code based on optional static typing. By catching errors early in the development process, TypeScript can reduce the likelihood of runtime errors and unexpected behaviour, leading to faster execution times. Additionally, the optional static typing in TypeScript can help the compiler optimize the code, resulting in faster execution and smaller output size. While human optimization of JavaScript is also possible, it can be error-prone, especially in large-scale projects. In conclusion, while TypeScript may not always offer a substantial performance boost compared to JavaScript, its compile-time error checking and optional static typing can lead to faster and more efficient code, especially in larger projects (Code A Star 2023).

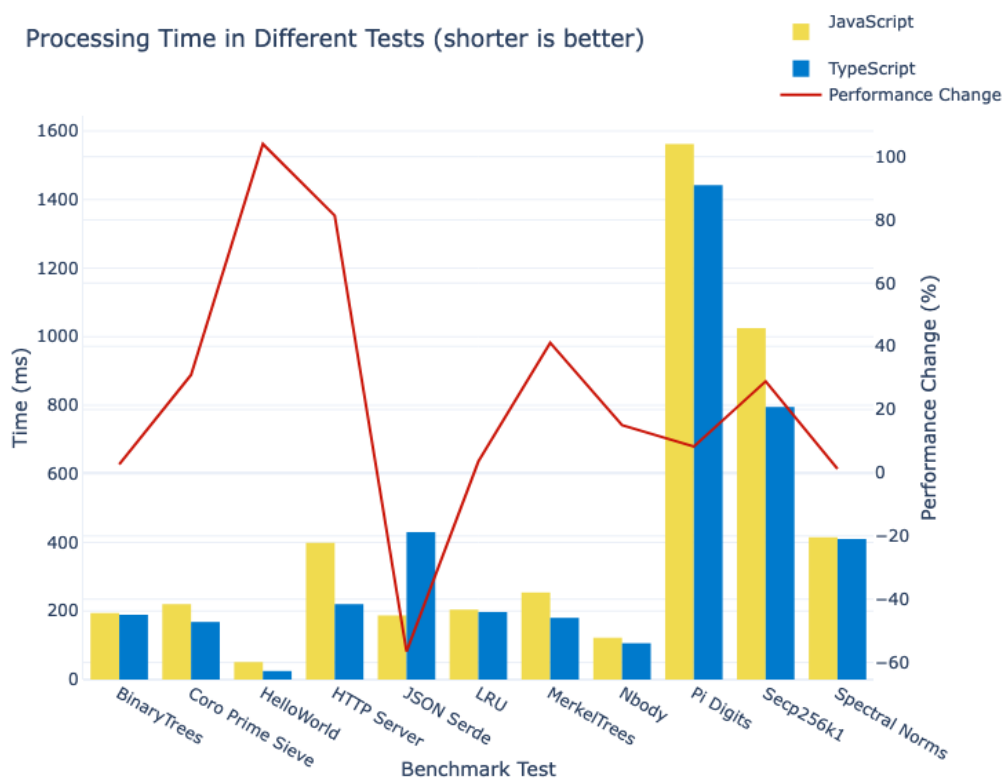


FIGURE 6. TS vs JS Processing Time Differences (Programming Language Benchmarks 2022)

Based on programming language benchmarks as of December 21, 2022, the graph shows that TypeScript outperforms JavaScript in 10 out of 11 benchmark tests. However, there is a benchmark test where TypeScript shows a significant drop in performance compared to JavaScript, which has to do with JSON serialization and deserialization. Even so, the overall performance difference between TypeScript and JavaScript is generally small and probably not large enough to have a significant impact on most real-world applications. It's important to note that performance results may vary depending on the specific use case, the implementation of the code, and the JavaScript engine used to run the code.

3 USE CASES OF TYPESCRIPT

It is designed to tackle large-scale applications by offering static typing and advanced features. Common use cases for TypeScript include building complex web applications, developing server-side applications with Node.js, and creating cross-platform mobile applications. It can also be used with popular frameworks and libraries such as Angular, React, and Vue (Kumar 2021). TypeScript's static typing ensures code quality, maintainability, and a better developer experience, making it a popular choice for various software development scenarios.

3.1 TypeScript in Action

A quiz app built using TypeScript involves creating multiple components that work together to provide a seamless user experience. When building a quiz app with TypeScript, we start by setting up the project structure and organizing our code. We define data structures like questions, answers, and user profiles using TypeScript's interfaces and types to catch errors early. Then, we create reusable components using classes and inheritance for different types of quiz questions. We also use popular front-end frameworks like Angular or React to build the user interface and integrate the app with a back-end server using TypeScript and Node.js. This way, we can handle user authentication and manage quiz data. TypeScript also helps us create custom UI components and reusable libraries if needed.

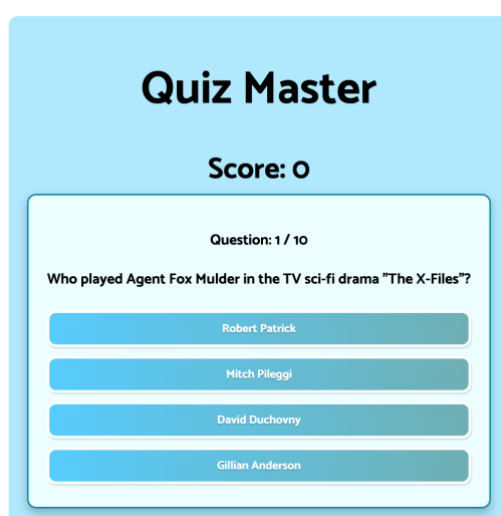


FIGURE 7. Build a TypeScript Based React Quiz Application.

3.2 Advantages and disadvantages of using TypeScript

TypeScript offers several advantages that can improve code quality, maintainability, and developer productivity. However, it also has some drawbacks, such as a learning curve and potential compatibility issues. When considering whether to use TypeScript, it's essential to weigh these pros and cons against the specific needs and goals of the project.

TABLE 2: Advantages and Disadvantages of TypeScript (yashdkadam n.d)

| Advantages of TypeScript | Disadvantages of TypeScript |
|--|--|
| Detects errors at compilation time | Takes longer to compile code |
| Optional static typing | Does not support abstract classes |
| Supports static and inferred typing | Third-party library definition files may not always be available |
| Runs in any browser or JavaScript engine | Quality of type definition files can be a concern |
| Robust tooling and IntelliSense | Requires compilation to JavaScript before running in a browser |
| Improved code organization | Finding TypeScript developers can be difficult |
| Better documentation for APIs | Not fully expressive with JavaScript |

3.3 Compatibility with Popular Frameworks (Angular and React)

TypeScript is highly compatible with popular web frameworks like Angular and React, as both frameworks have built-in support for TypeScript. This compatibility enables developers to take advantage of TypeScript's features when building applications with Angular or React, leading to improved code quality and maintainability.

3.3.1 Angular with TypeScript

Angular has built-in TypeScript support, and the Angular framework itself is written in TypeScript. The Angular CLI, which is used to generate, develop, and manage Angular projects, uses TypeScript by default. TypeScript's static typing and interfaces integrate seamlessly with Angular's component-based architecture, dependency injection, and decorators, resulting in a more robust and maintainable codebase.



FIGURE 8. Angular With TypeScript Code Example (Open AI, ChatGpt 2023)

3.3.2 React with TypeScript

React also supports TypeScript and can easily set up a TypeScript-based React project using Create React App with the TypeScript template. TypeScript works well with React's functional components, hooks, and the overall component architecture. It allows developers to create strongly typed props and state, leading to better error checking during development and making it easier to refactor and maintain the codebase.



```
1  import React from 'react';
2
3  interface Props {
4    name: string;
5  }
6
7  const Greeting: React.FC<Props> = ({ name }) => {
8    return (
9      <div>
10       <h1>Hello {name}!</h1>
11     </div>
12   );
13 };
14
15 export default Greeting;
16
```

FIGURE 9. React With TypeScript Code Example (Open AI, ChatGpt 2023)

3.4 Case Studies of Companies & Organizations Using TypeScript

TypeScript has gained significant traction in the software development community, and several companies and organizations have adopted it to improve their development processes and code quality. These case studies showcase how TypeScript has been beneficial for various organizations, helping them improve code quality, maintainability, and developer productivity (Jessica Clark, n.d). Adopting TypeScript can offer similar advantages to other companies and organizations looking to enhance their software development processes.

3.4.1 Microsoft

Microsoft, the creator of TypeScript, uses it extensively across various products and services. One notable example is Visual Studio Code (VS Code), a popular open-source code editor that is built using TypeScript. TypeScript's strong typing and enhanced tooling have helped Microsoft maintain and scale the VS Code codebase effectively (Jessica Clark, n.d).

3.4.2 Slack

Slack, a well-known collaboration platform, adopted TypeScript to improve the scalability and maintainability of its desktop application. TypeScript's static typing, better tooling, and seamless integration with the Electron framework enabled the Slack team to catch errors early, improve developer productivity, and maintain a large codebase more efficiently (Jessica Clark, n.d).

3.4.3 Airbnb

Airbnb, a leading online marketplace for lodging, switched to TypeScript to improve its development process and code quality. TypeScript's static typing, superior refactoring capabilities, and strong community support helped Airbnb catch bugs early, reduce technical debt, and enhance collaboration among the development team members (Jessica Clark, n.d).

3.4.4 Asana

Asana, a popular project management and productivity tool, migrated to TypeScript to improve the maintainability and scalability of its application. The adoption of TypeScript allowed Asana to catch bugs before they reached production, enhance developer productivity, and streamline the development process (Jessica Clark, n.d).

4 IMPACT OF TYPESCRIPT ON WEB DEVELOPMENT

TypeScript has had a significant impact on web development since its release in 2012. It has improved code quality by providing developers with a powerful tool to detect errors before they runtime. Its advanced features have also improved productivity and made it easier to write and maintain complex applications. It's static typing and code structure have enabled better collaboration among developers and increased adoption of JavaScript frameworks.

4.1 Improved code maintainability and scalability

In addition to the impacts mentioned earlier, TypeScript also has a significant impact on code maintainability and scalability. By providing strong typing and advanced features like classes and interfaces, TypeScript helps developers write code that is easier to maintain and extend over time.

TypeScript also offers features like code completion and refactoring support, which can help developers quickly navigate code and make changes efficiently. These features can make it easier to refactor code, add new features, or modify existing functionality without introducing errors or breaking the code. The improved maintainability and scalability provided by TypeScript can help developers build more reliable and maintainable web applications. By writing cleaner, more organized, and more reliable code, developers can reduce development time and ensure that their code can be maintained and scaled effectively over time (Dojo 2023).

4.2 Early error detection and debugging

Another significant impact of TypeScript on web development is its ability to enable early error detection and debugging. Since TypeScript is a statically typed language, it can catch errors before runtime, which can significantly reduce the time and effort required for debugging and testing.

To get better understand, suppose we have a function that expects a string parameter, but we accidentally pass a number instead. In a dynamically typed

language like JavaScript, this error would only be caught at runtime, leading to unexpected behavior and potentially difficult-to-debug issues. However, with TypeScript, this error would be caught at compile-time, allowing to fix the issue before it becomes a problem.

```
TS example.ts > ...
1  function multiply(a: number, b: number) {
2  |   return a * b;
3  | }
4
5  const result = multiply(2, '3'); // Error:
6
```

FIGURE 10. Early Error Detection and Debugging (Open AI, ChatGpt 2023).

In Figure9, we have a function called `multiply` that expects two number parameters and returns their product. When we try to call the `multiply` function with a string parameter instead of a number, TypeScript will catch the error at compile-time and prevent the code from running. The error message will indicate that the argument passed is of the wrong type and is not assignable to the parameter's type (Open AI, ChatGpt 2023).

In addition to catching errors early, TypeScript can also improve the debugging experience by providing more informative error messages. The strong typing of TypeScript allows for more detailed error messages, making it easier for developers to identify and fix issues in their code.

4.3 Improved Productivity (Visual Code)

TypeScript has had a positive impact on the productivity of web developers, particularly when combined with development tools like Visual Studio Code (VS Code). By offering strong typing and advanced features such as classes and interfaces, TypeScript can help developers write cleaner and more organized code, making development tasks quicker and more efficient. Additionally, TypeScript provides features such as code completion and refactoring support,

which can further improve productivity by simplifying the development process. Ultimately, TypeScript has enabled developers to write better code more quickly, reducing development time and enhancing the overall productivity of web development projects (Attardi 2022).

4.4 Limitations and Challenges of TypeScript

TypeScript has its limitations and challenges that need to be considered when using it in web development. One of the challenges is that it has a steeper learning curve compared to JavaScript, especially for developers who are not familiar with static typing or advanced features like classes and interfaces. Additionally, not all existing JavaScript libraries and tools have TypeScript support, which can make it difficult to integrate TypeScript into existing projects or use certain libraries and tools (Marius 2021).

Another challenge is that since TypeScript needs to be compiled to JavaScript, there is an additional step in the development process that can slow down iteration cycles and increase build times. While TypeScript's static typing provides many benefits, it can also be limiting in some cases. For example, some dynamic or loosely-typed scenarios may require more flexibility than TypeScript allows. Therefore, it is essential to weigh the benefits against the costs and assess whether TypeScript is the best choice for a particular project. While TypeScript can improve code quality, productivity, and maintainability, developers should consider its limitations and challenges before using it (Gustav n.d).

4.5 Compatibility Issues with Some Libraries and Frameworks

One of the challenges that developers may face when using TypeScript in web development is compatibility issues with some libraries and frameworks. This is because not all existing JavaScript libraries and frameworks have TypeScript support, which can make it difficult to integrate TypeScript into existing projects or use certain libraries and tools.

But still there is good news: many popular JavaScript libraries and frameworks have added TypeScript support in recent years, making it easier for developers to use TypeScript in their projects. Despite this progress, it's still possible that

developers will need to write custom type definitions for libraries that do not have TypeScript support, which can be time-consuming and not always provide a complete solution.

To help bridge the gap, TypeScript provides tools like declaration files and the any type that can help developers work with existing JavaScript code that does not have TypeScript support. While compatibility issues with some libraries and frameworks can be a challenge, the increasing popularity and adoption of TypeScript are driving more libraries and frameworks to support it (TypeScript Handbook 2023)

Moreover, while there may be some challenges when using TypeScript in web development, the benefits it provides in terms of improved code quality, productivity, and maintainability make it worth considering. By keeping an eye on compatibility issues and using the tools available, developers can take advantage of the language's strengths and continue to use it effectively in their projects.

4.6 Performance Considerations

When using TypeScript in web development, there are some performance considerations to keep in mind. TypeScript can improve code quality and maintainability, but it can also add some overhead in terms of performance.

One thing to consider is compilation time. Since TypeScript needs to be compiled to JavaScript, this can slow down the development process and increase build times. However, developers can use tools like Webpack or Gulp to optimize the build process and minimize compilation time. Another thing to consider is runtime performance. While TypeScript code can be optimized by the compiler, the overhead introduced by type checking and other language features can impact runtime performance. Developers can use techniques like lazy loading to improve the initial load time of web applications and minimize the impact of runtime performance (Nikita 2022).

Memory usage is another consideration. The use of classes and other advanced features in TypeScript can increase memory usage compared to traditional JavaScript code. To optimize memory usage, developers can use techniques like object pooling and minimize unnecessary object creation.

Finally, code size can be impacted by TypeScript due to the additional typing information and language features. This can impact page load times and overall performance. To minimize code size, developers can use techniques like tree shaking and minification to remove unused code and compress the codebase. Overall, while TypeScript can introduce some performance overhead, these issues can be minimized by using best practices and optimization techniques. By using TypeScript effectively, developers can improve code quality and maintainability while ensuring that their applications are performant and efficient (JSdairies 2019).

4.7 TypeScript Community and Ecosystem

TypeScript has a growing and active community of developers, contributors, and enthusiasts. According to the TypeScript website, there are more than 2 million TypeScript downloads per month, and the language is used by companies such as Microsoft, Google, Asana, Slack, and Airbnb.

Documentation and Resources: The TypeScript community has a wealth of resources available for learning and using the language. The official TypeScript website provides extensive documentation, guides, and tutorials, as well as a playground to experiment with TypeScript code. There are also many third-party resources, such as blogs, videos, books, and courses, that cover TypeScript in depth.

Frameworks and Libraries: Many popular web frameworks and libraries support TypeScript, including Angular, React, Vue, Express, Nest.js, and more. These frameworks provide TypeScript-specific features, such as decorators, type annotations, and tooling integration, that make TypeScript development more productive and efficient.

Tools and Editors: There are many tools and editors that support TypeScript, such as Visual Studio Code, WebStorm, Atom, Sublime Text, and more. These tools provide features such as autocomplete, syntax highlighting, code formatting, and debugging, that make TypeScript development easier and more enjoyable.

Community Contributions: The TypeScript community is actively contributing to the language and its ecosystem. There are many open-source projects and

repositories on platforms like GitHub that provide TypeScript definitions, plugins, and utilities. The community also reports bugs, suggests improvements, and proposes new features through the official TypeScript GitHub repository.

Conferences and Events: There are many conferences and events that focus on TypeScript and related technologies. These events provide opportunities for developers to learn, network, and share their experiences with TypeScript. Some of the notable conferences include ng-conf, React Conf, Vue.js Summit, and TypeScript Day (TS official, TS documentation 2023)

The TypeScript community and ecosystem offer a vibrant and supportive environment for web developers who want to use a more robust and structured language than JavaScript. Whether we are a beginner or an experienced developer, there are many resources and tools available to help we learn and use TypeScript effectively.

4.8 The Importance of TypeScript Popularity

The importance of TypeScript's popularity and thorough documentation in web development cannot be overstated. The language's increasing popularity and widespread adoption are driving more libraries and frameworks to support it, making it easier for developers to use TypeScript in their projects. In addition, TypeScript's comprehensive documentation provides developers with clear explanations of its features and syntax, helping them write efficient and organized code that is easier to maintain and scale. By leveraging the language's popularity and thorough documentation, developers can take advantage of the many benefits Typescripts offers, such as improved code quality, productivity, and maintainability, and build more reliable and efficient web applications (Open AI, ChatGpt 2023).

4.8.1 Stack Overflow

TypeScript's popularity is driven by its ability to improve code quality, maintainability, and scalability, making it an attractive option for web developers. Its strong typing and advanced features like classes and interfaces help developers write cleaner, more organized code, reducing the likelihood of bugs and making code easier to maintain and scale. TypeScript's growing popularity

has also led to more libraries and frameworks adding TypeScript support, making it easier for developers to integrate TypeScript into their projects

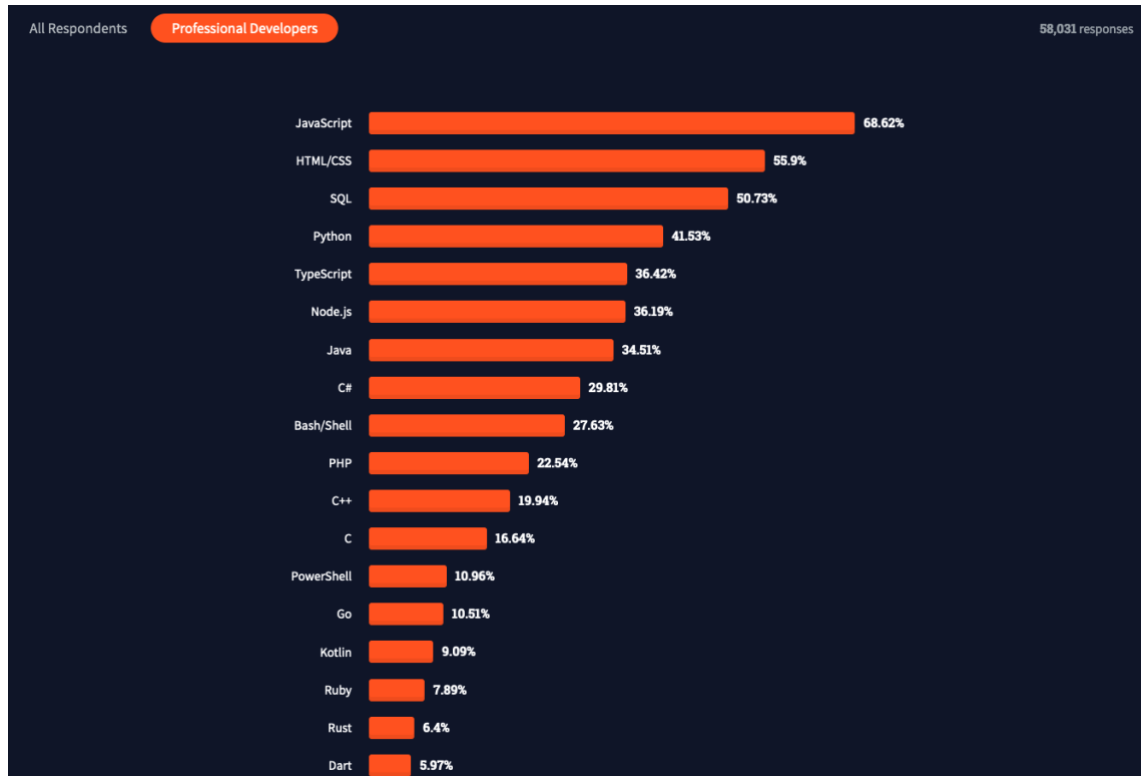


FIGURE 11. Stack Overflow Survey list 2021 (Stack Overflow Developer Survey n.d)

TypeScript has been steadily growing in popularity among developers over the past few years. According to the Stack Overflow Developer Survey 2021, It was the 5th most commonly used technology among professional developers worldwide, with 36.2%. This places TypeScript ahead of other popular web development technologies such as Vue.js, Angular, and Ruby on Rails.

Furthermore, TypeScript was also among the top 5 most loved programming languages, with 73.1% of respondents saying they loved working with it. This highlights the popularity of TypeScript among developers and its growing adoption in the web development community

4.8.2 GitHub

According to the 2021 GitHub State of the Octoverse report, TypeScript was the 4th most popular language on GitHub, based on the number of repositories created in 2021, with over 2.3 million repositories. Certainly! The Hashicorp Configuration Language (HCL) has become more popular over the past year, largely due to the growing popularity of the Terraform tool and Infrastructure as Code (IaC) practices that automate deployments. As a result, other languages like Go. TypeScript have also seen significant increases in usage.

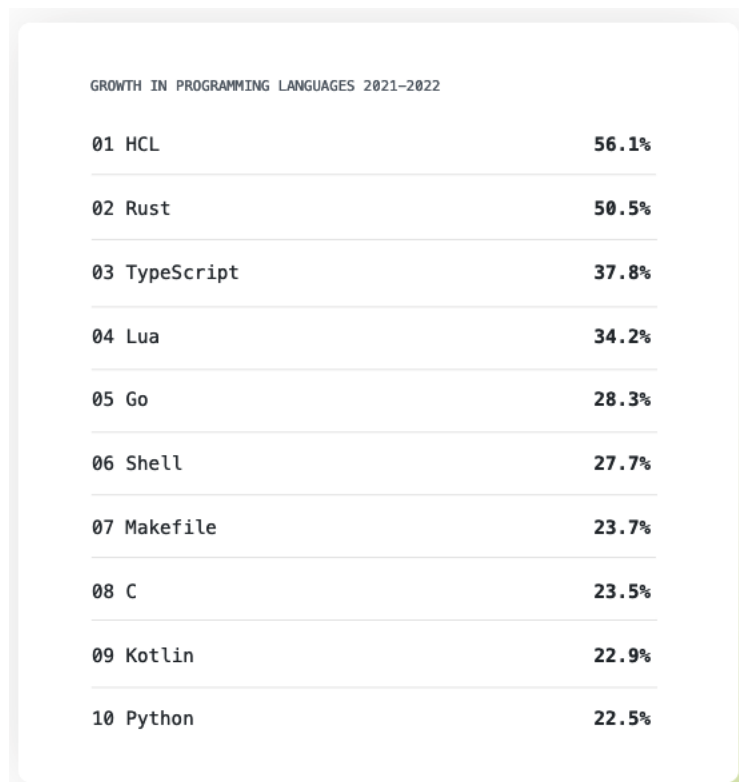


FIGURE 12. Octoverse Report 2021-2022 (Octoverse 2022)

According to GitHub's analysis of programming language growth from 2021 to 2022, TypeScript has moved up to the 3rd position with a growth rate of 37.8%. This indicates a significant increase in the adoption and popularity of TypeScript in the programming community.

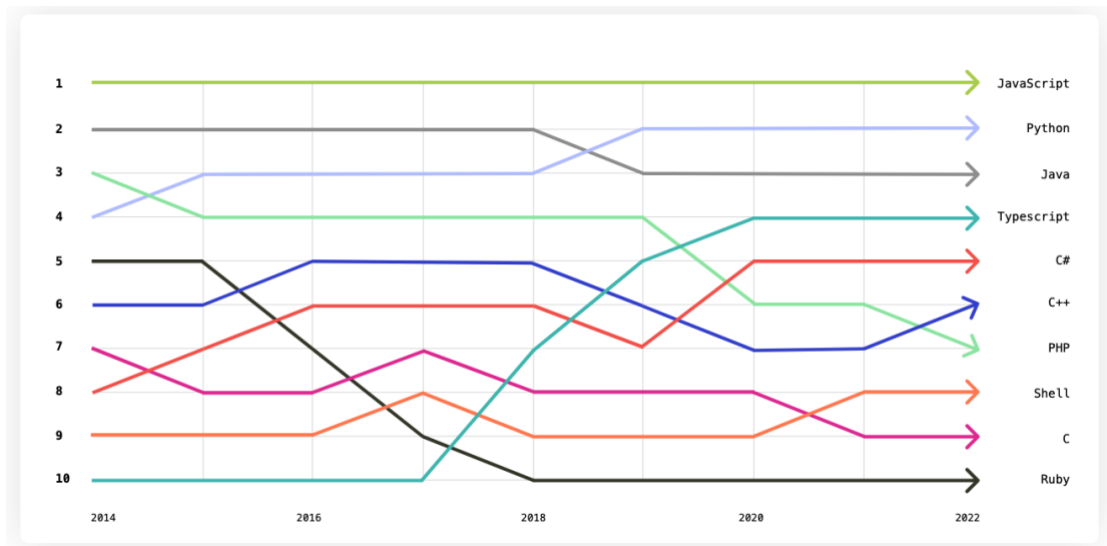


FIGURE 13. Octoverse Report 2022 (Octoverse 2022)

JavaScript remains the most widely used programming language, and Python has maintained its second place position due to its versatility across various domains such as development, education, machine learning, and data science. TypeScript, a superset of JavaScript that adds features like static typing, has also maintained its position as the fourth most popular language year-over-year. Interestingly, PHP dropped from sixth to seventh place in the 2022 rankings.

4.8.3 Google Trends

According to Google Trends, the popularity of TypeScript has been on the rise over the past 5 years. There has been a steady increase in interest in TypeScript, with a significant spike in early 2021. This suggests that TypeScript is becoming more widely used and adopted by developers around the world.

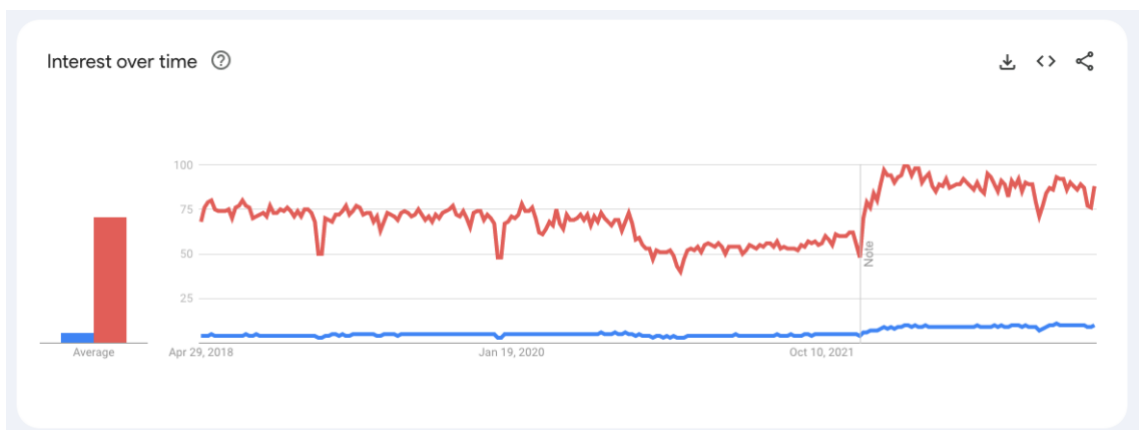


FIGURE 14. Google Trends View 2021 (Google trends 2023)

4.9 Potential Drawbacks

Despite it provides many benefits to developers, such as improved code readability, reduced errors, and better maintainability it also has some potential drawbacks that developers should be aware of. One such drawback is the learning curve, as developers who are new to TypeScript may need some time to learn the new concepts and syntax introduced by the language. Another potential drawback is limited tooling support, which can lead to issues such as code completion not working correctly or debugging being difficult. Over-engineering limited third-party library support, compilation overhead, and increased file sizes are also potential drawbacks of using TypeScript. However, with careful consideration and use of the language's benefits, developers can mitigate these potential drawbacks and effectively use TypeScript to improve their code's quality and maintainability. Ultimately, weighing the benefits and drawbacks of using TypeScript can help developers make an informed decision about whether to use it for a project (Alexsoft 2020)

5 DISCUSSION

As web development evolves, developers are constantly looking for better ways to write efficient and maintainable code. That's where TypeScript comes in - it's a superset of JavaScript that offers several benefits over its prior. TypeScript provides early error detection, better tooling support, and cleaner syntax, which helps developers improve the quality of their code and increase productivity.

However, it's worth observing that switching to TypeScript from JavaScript may come with some challenges. Developers who are already comfortable with JavaScript may need to put in some effort to learn TypeScript. Additionally, transpiling TypeScript to JavaScript adds an extra step to the development process, which could slow things down. Despite these challenges, TypeScript's benefits make it a worthwhile investment for developers, especially for larger-scale projects where writing efficient and maintainable code is critical. It's no surprise that major companies like Microsoft and Google are adopting TypeScript in their development workflows, further upgrowth its effectiveness in web development.

In conclusion, while there may be a learning curve associated with TypeScript, it provides several advantages over JavaScript, making it a powerful alternative for web development. Developers should evaluate their project needs and invest time in learning TypeScript to determine if it's the right choice for them.

REFERENCES

Official TypeScript Documentation. n.d. TypeScript basics and context.
Read 01.01.2023 <https://www.typescriptlang.org/docs/>

Wikipedia. n.d. TypeScript: Theoretical Background. Read 01.02.2023
<https://en.wikipedia.org/wiki/TypeScript>

TypeScript Handbook. n.d. TS for the New Programmer, TypeScript for JS Programmers, TS for Java/C# Programmers, TS for Functional Programmers, TypeScript Tooling in 5 minutes.
Read 01.02.2023 <https://www.typescriptlang.org/docs/>

Gaurav Khanna. n.d. Useful React APIs For Building Flexible Components with TypeScript. Read 01.02.2023
<https://www.smashingmagazine.com/2021/10/react-apis-building-flexible-components-typescript/>

Giuseppe Maggiore. n.d. Category theory for TypeScript programmers. Read 01.02.2023 <https://medium.com/@giuseppemaggiore/category-theory-for-typescript-programmers-a2372f771b20>

Mad Devs. n.d. Quick start with Typescript and React. Read 05.02.2023
<https://www.smashingmagazine.com/2021/10/react-apis-building-flexible-components-typescript/>

Tutorialspoint. n.d. Learn TypeScript absolute beginners. Read 01.01.2023
https://www.tutorialspoint.com/typescript/typescript_quick_guide.htm

Abhimanyu Krishnan. n.d. TypeScript vs JavaScript: Which is Best in 2023. Read 01.03.2023 <https://hackr.io/blog/typescript-vs-javascript>

Janeth Kent. n.d. TypeScript: The evolution of JavaScript.

Read 02.03.2023

<https://www.mano.org/en/programming/javascript/typescript-the-evolution-of-javascript>

Tutorialspoint. n.d. Interfaces, classes, modules.

Read 15.01.2023

https://www.tutorialspoint.com/typescript/typescript_quick_guide.htm

Stack Overflow Developer Survey n.d. Stack overflow survey check

2021-2022 Read 16.04.2023

<https://insights.stackoverflow.com/survey/2021#most-loved-dreaded-and-wanted-languages-loved>

Octoverse n.d. Octoverse survey check 2021-2022. Read 17.04.2023

<https://octoverse.github.com/>

Google Trends n.d. Google Trends check 2021-2022 Read 17.04.2023.

<https://trends.google.com/trends/explore?date=today%205-y&q=typescript>

Code A Star. n.d. TypeScript vs JavaScript: Which is Best in 2023

Read 15.04.2023 <https://www.codeastar.com/take-it-easy-lets-find-out-which-one-is-better-javascript-or-typescript/>

Alexsoft. 2020. The Good and the Bad of TypeScript Read 16.04.2023

<https://www.altexsoft.com/blog/typescript-pros-and-cons/>

