Preliminary Report

Jonas Hübotter

December 1, 2020

Abstract

TypeScript is a programming language developed by Microsoft and first published in 2012. TypeScript is a superset of JavaScript adding static typing and transpiling to JavaScript. Today, TypeScript is widely used in many areas and supported by many development environments. TypeScript is open-source and its development takes place on GitHub.

1 History

To understand TypeScript's roots its necessary to first understand the environment in which the initial idea was born. This is especially true as TypeScript is so deeply rooted in JavaScript.

Brendan Eich famously created the first prototype of JavaScript, then named Mocha, in September 1995 in just ten days [Hof19]. The short time it took to develop the first version of Mocha meant that it was inherently unfinished. Originally, JavaScript was envisioned as a scripting language to add small effects to websites. The adoption of JavaScript happened quickly. Initially developed at Netscape, Microsoft released JScript in 1996 which was almost identical. This, however, made the need for standardization of the language apparent. In 1997, the first ECMAScript specification was released based on Netscape's JavaScript.

Through the early adoption by Netscape and Microsoft's Internet Explorer, the most widely used browsers at the time, JavaScript gained traction very quickly. Soon after its inception it had essentially gained a monopoly on client-side code evaluation in browsers.

Already in 1999, with the first draft of ECMAScript version 4, a group led by Mozilla, proposed drastic changes to JavaScript aiming to turn it into a more complete

programming language and leaving behind its roots as a scripting language. This new version of the spec was supposed to include numerous features such as classes, interfaces, a module system and even a static type checker, many of which were advanced for its time. However, Douglas Crockford, creator of the JavaScript Object Notation (JSON) and working at Yahoo! at the time, fought the proposal. He was in favor of keeping JavaScript a more lightweight language that was true to its roots as a scripting language.

Ironically, in hindsight, Microsoft joined Douglas Crockford in his criticism and threatened to ignore the new version of the spec if it was accepted. Microsoft argued that the proposal was poorly designed and simply too big. However, it's criticism was later mostly attributed to its preference for proprietary software. Microsoft sought to keep its market lead by sharing as little as possible with its competitors. At the same time, it was also speculated that Mozilla's proposal was a deliberate attempt to attack Microsoft's firm grip on the market by giving an edge to its competitors who designed the proposal. Still, the reference implementation of the spec, while incomplete, showed that its implementation was conceivable.

Ultimately, Microsoft put forward a proposal for ECMAScript version 3.1 which got accepted, leading to the abandonment of ECMAScript version 4. This controversy, led to a decade-lasting deadlock in further development of the JavaScript spec. The next version of the spec was only released in December 2009.

While almost all widely used browsers used JavaScript, their implementations differed tremendously. Throughout the 2000s, Microsoft in particular, was adding many features developed "in-house" on top of the JavaScript implementation to its Internet Explorer. This led to a very dispersed landscape of JavaScript implementations across browsers. The deadlock in development of the JavaScript language further contributed to this dispersion.

Atwood's law from 2007 stating

> "Any application that can be written in JavaScript, will eventually be written in JavaScript." [K18]

by Jeff Atwood, a cofounder of StackOverflow, exemplified the role JavaScript was perceived to play at the time despite the lack of tooling and development of the language. With the end of the 2000s, however, a new period of investment into the JavaScript ecosystem began.

In 2009, Ryan Dahl created Node.js which was crucial in making JavaScript conceivable as a programming language in its own right and not merely a scripting

language used in browsers. The inception of Node.js also cemented JavaScript's lead as the most popular programming language. At the same time, with the development of ECMAScript version 6 many features from version 4 such as classes and the module system were revived. The now more rapid development of JavaScript also motivated the wider use of transpilers like Babel or CoffeeScript, compiling newer JavaScript to older targets, to allow the use of newer JavaScript with legacy browsers.

1.1 Motivation

While JavaScript was incredibly popular, it wasn't (and is still not) particularly loved. Its roots as a scripting language entailed that it was executed by an interpreter and types were evaluated at runtime.

Anders Hejlsberg was one of the people leading the TypeScript project as it started out. Before beginning to work on TypeScript around 2010, Hejlsberg led the development of Delphi and C#. In a conversation with James Gosling, the creator of Java, in 2019, Hejlsberg framed the motivation behind TypeScript based on the problem of refactoring:

> "You're describing what was the genesis of the TypeScript project, which was these enormous JavaScript code bases that we were seeing customers write, and in-house projects... It turned into write-only code, you know? You dared not touch it, once you've written it because it worked..." [Cas19]

One of the main goals of TypeScript continues to be to make large scale development with JavaScript feasible. Feedback from other Microsoft teams working on more and more web-based software in the late 2000s indicated that they had difficulties in making JavaScript scale [Fol12b]. Heljsberg argued that "a type system is one way you can reason about your code. It's the ability to check your code before you run and deploy it. Without types in a language that's almost impossible." [Tun20]. Soma Somasegar, then Corporate Vice President of Microsoft's Developer Division, said in 2012:

> "At the same time, the reach of JavaScript has continued to expand, going beyond the browser to include native device apps (e.g. Windows Store apps for Windows 8), applications in the cloud (e.g., node.js running on Windows Azure), and more. With these developments, we're starting to

> see applications of unprecedented size written with JavaScript, despite the fact that creating large-scale JavaScript applications is hard. TypeScript makes it easier." [Fol12a]

The general push towards broadening the application domains of JavaScript was a driving force behind TypeScript. With more backend use — a scenario where static types were expected — and the surge in use of transpilers, TypeScript was able to address a number of prevalent problems of JavaScript development in the early 2010s.

1.2 The TypeScript project

Development on TypeScript began in 2010 leading up the first public release of version 0.8 in 2012. Since then TypeScript has continuously been under active development. The most notable features added in later releases were the addition of generics in 2014, the option to disallow `null` values in 2016 and the support of ECMAScript version 6 in 2018.

From its public announcement TypeScript was made open-source. Initially the development was hosted on CodePlex, an open-source sharing platform developed by Microsoft. In 2014 its code and development was moved to GitHub [Tur14].

TypeScript being a strict syntactical superset of JavaScript meant that any valid JavaScript program would also compile with TypeScript. Moreover, developers didn't have to decide between typing everything or not typing at all. With TypeScript they were also able to type small subsets of their code. While fundamentally, TypeScript is similar to the type system that was originally envisioned for version 4 of ECMSScript, it erases types after type checking rather than using them at runtime. Furthermore, TypeScript is self-hosted, that is, the TypeScript compiler itself is written in TypeScript.

A year before Microsoft publicly announced TypeScript in 2012, Google already presented Dart, a programming language that was intended to replace JavaScript and also included a static type system. Unlike TypeScript, however, Dart was not a superset of JavaScript but an entirely different language. The mixed reception of the project and the general trend towards transpiling to JavaScript ultimately led to the abandonment of the original goal of Dart. Today, TypeScript is by far the most widely used static type system for JavaScript [Hiw20].

1.3 Open-Source at Microsoft

TypeScript also represents a turning point in Microsoft's attitude towards open-source software. Former CEO Steve Ballmer famously called Linux a *"cancer"* and complained that in an open-source model no one is held accountable for flawed software [Pen18]. Still under Steve Ballmer, the TypeScript team decided in 2010 to eventually make TypeScript open-source. Hejlsberg said that at the time Microsoft was "very ambivalent" and even "afraid" of open-source. The success of TypeScript, however, helped tremendously in supporting the acceptance of open-source at Microsoft, ultimately, laying the groundwork of today's open-source efforts of Microsoft [Tun20].

2 Use

TypeScript is used both for client- and server-side development. StackOverflow's developer survey from 2020 found TypeScript to be the ninth most popular programming language, ahead of C++, C, Go, Ruby and Swift. It also identified TypeScript as the second most loved language, trailing only Rust [Sta20].

The TypeScript repository has more than 66 thousand stars and is used by 2.8 million other GitHub repositories. The `typescript` NPM package has over 14 million weekly downloads.

3 Structure, stakeholders and license

While TypeScript is open-source, it is developed by a dedicated team at Microsoft led by Ryan Cavanaugh. The team is accountable for processing incoming issues and feature requests, prioritzing work and spearheading the development of new features. Additionally, the team at Microsoft maintains the website and documentation, runs a blog and does general publicity work. This publicity work includes hosting talks at events and a maintaining dedicated Twitter account [Mic20].

As TypeScript is a large project, responsibilities are shared among members of the team. Some members of the team focus primarily on the compiler while others focus on DefinitelyTyped — a package that includes type declarations for most widely used JavaScript libraries. Product managers run the blog and announce new releases of TypeScript while engineers work on new features both inside the team and with outside collaborators.

TypeScript is designed and funded entirely by Microsoft. Some meaningful contributions come from outside collaborators, yet most development efforts are made by the TypeScript team at Microsoft.

TypeScript is licensed under the Apache License 2.0, a permissive license that restricts trademark use, liability and warranty [Mic14].

4 Communication

As to be expected for a project of this size, TypeScript has numerous channels for communication [Micb]:

- The `typescript` tag on StackOverflow can be used to ask questions.
- Alternatively to StackOverflow, questions can be asked on the TypeScript Community Chat hosted on Discord.
- News are announced on Twitter.
- More detailed announcements and sometimes articles related to TypeScript are published on the TypeScript Blog.

5 Contributing

The TypeScript compiler has very detailed guidelines on where and how to contribute to the project [Mic20].

5.1 Logging issues

Before opening an issue, contributors are asked to refer to the comprehensive FAQ and search the existing issues to prevent duplicates [Mic20].

When reporting a bug, it is requested that you provide an isolated way of reproducing the behavior. When writing-up a suggestion, it is requested that — beyond a description of the problem and the proposed solution — you give examples of how the suggestion would work in certain places [Mic20].

The GitHub repository also makes use of issue templates to direct contributors to these guidelines.

5.2 Contributing code

To contribute code, contributors first have to find a ticket in the issue tracker that requests outside help or is in the `Backlog` milestone. To this end, TypeScript makes use of the `help wanted` issue label. The TypeScript compiler does not accept changes that are proposed without a reference to an issue from the issue tracker that was scheduled by a member of the TypeScript team [Mic20].

Once a ticket has been found, the code has to be forked and cloned and the development dependencies have to be set up. These include Node.js and gulp. Detailed instructions can be found in the contributing guidelines [Mic20].

Submitted pull requests should be well-described, well-tested and adhere to the coding guidelines. The contributing guidelines elaborate further on what *well-tested* means.

When submitting their first pull request, contributors have to complete a Contributor License Agreement that grants Microsoft the permission to use submitted changes according to the terms of the license and testifies that the work is submitted under appropriate copyright [Mic20].

5.3 Contribution history

While over the years many users have contributed to TypeScript, most contributions were made by the core TypeScript team. Since its move to GitHub in 2014, 538 people have contributed to the TypeScript compiler. Throughout the lifespan of the project, contributions have been very steady [Mica].

6 Release management

TypeScript publishes a new full feature release every other month and at least one release every month. Releases are published roughly one week before the monthly release of Visual Studio Code. This usually falls in the last week of a month. Release candidates are published two to three weeks before a scheduled release [Heg18]. Features that are not complete by release time are deferred to the next release [Ros19].

7 Related projects

The TypeScript compiler sits at the center of an entire ecosystem of tools and libraries built around TypeScript. These include

- DefinitelyTyped — a collection of type declarations for widely used JavaScript libraries that do not come with their own type declarations;
- TypeScript-Website — hosting the website of the project; and
- various language packages and linters for VS Code, Sublime and other editors.

8 Conclusion

The circumstance of its inception and the history of Microsoft turn the history of TypeScript into an interesting tale. TypeScript was a culmination of the trends towards large-scale JavaScript apps, a broader application of JavaScript and transpiling to JavaScript leading to its rapid adoption. In addition, TypeScript marked a turning point in Microsoft's relationship with open-source. Today, TypeScript is one of the most widely used open-source projects. Both, development of the project and the project itself, are extremely well documented.

References

[Fol12a] Mary Jo Foley. *Microsoft takes the wraps off TypeScript, a superset of JavaScript*. Oct. 2012. URL: https://www.zdnet.com/article/microsoft-takes-the-wraps-off-typescript-a-superset-of-javascript/.

[Fol12b] Mary Jo Foley. *Who built Microsoft TypeScript and why*. Oct. 2012. URL: https://www.zdnet.com/article/who-built-microsoft-typescript-and-why/.

[Mic14] Microsoft. *License*. July 2014. URL: https://github.com/microsoft/TypeScript/blob/master/LICENSE.txt.

[Tur14] Jonathan Turner. *New Compiler and Moving to GitHub*. July 2014. URL: https://devblogs.microsoft.com/typescript/new-compiler-and-moving-to-github/.

[Heg18] Mohamed Hegazy. *Release process?* July 2018. URL: https://github.com/Microsoft/TypeScript/issues/25639#issuecomment-404889860.

[K18] Jayaprabhakar K. *Rethinking Atwood's Law*. Jan. 2018. URL: https://medium.com/@jayaprabhakar/rethinking-atwoods-law-64a894b54aa4.

[Pen18] Havoc Pennington. *Steve Ballmer was right about open source*. Mar. 2018. URL: https://blog.tidelift.com/steve-ballmer-was-right-about-open-source.

[Cas19] David Cassel. *A Conversation with the Creators Behind Python, Java, TypeScript, and Perl*. Apr. 2019. URL: https://thenewstack.io/a-conversation-with-the-creators-behind-python-java-typescript-and-perl/.

[Hof19] Jay Hoffmann. *The 10-Day Programming Language (Is Kind of a Myth)*. Oct. 2019. URL: https://thehistoryoftheweb.com/the-10-day-programming-language-is-a-myth/.

[Ros19] Daniel Rosenwasser. *TypeScript's New Release Cadence*. Feb. 2019. URL: https://devblogs.microsoft.com/typescript/typescripts-new-release-cadence/.

[Hiw20] Uday Hiwarale. *A beginner's guide to TypeScript (with history)*. Sept. 2020. URL: https://medium.com/jspoint/typescript-a-beginners-guide-6956fe8bcf9e.

[Mic20] Microsoft. *Contributing guidelines*. Aug. 2020. URL: https://github.com/microsoft/TypeScript/blob/master/CONTRIBUTING.md.

[Sta20] StackOverflow. *Stack Overflow Developer Survey 2020*. Feb. 2020. URL: https://insights.stackoverflow.com/survey/2020.

[Tun20] Liam Tung. *TypeScript creator: How the programming language beat Microsoft's open-source fears*. Sept. 2020. URL: https://www.zdnet.com/article/typescript-creator-how-the-programming-language-beat-microsofts-open-source-fears/.

[Mica] Microsoft. *Contributors*. URL: https://github.com/microsoft/TypeScript/graphs/contributors.

[Micb] Microsoft. *TypeScript Community*. URL: https://www.typescriptlang.org/community.