TypeScript

CI passing Azure Pipelines succeeded npm package 4.8.2 downloads 139M/month

TypeScript is a language for application-scale JavaScript. TypeScript adds optional types to JavaScript that support tools for large-scale JavaScript applications for any browser, for any host, on any OS. TypeScript compiles to readable, standards-based JavaScript. Try it out at the playground, and stay up to date via our blog and Twitter account.

Find others who are using TypeScript at our community page.

Installing

For the latest stable version:

```
npm install -g typescript
```

For our nightly builds:

```
npm install -g typescript@next
```

Contribute

There are many ways to contribute to TypeScript.

- Submit bugs and help us verify fixes as they are checked in.
- Review the source code changes.
- Engage with other TypeScript users and developers on StackOverflow.
- Help each other in the TypeScript Community Discord.
- Join the #typescript discussion on Twitter.
- Contribute bug fixes.
- Read the archived language specification (docx, pdf, md).

This project has adopted the Microsoft Open Source Code of Conduct. For more information see the Code of Conduct FAQ or contact opencode@microsoft.com with any additional questions or comments.

Documentation

- TypeScript in 5 minutes
- Programming handbook
- Homepage

Building

In order to build the TypeScript compiler, ensure that you have Git and Node.js installed.

Clone a copy of the repo:

```
git clone https://github.com/microsoft/TypeScript.git
```

Change to the TypeScript directory:

```
cd TypeScript
```

Install [Gulp](https://gulpjs.com/) tools and dev dependencies:

```
npm install -g gulp
npm ci
```

Use one of the following to build and test:

```
gulp local             # Build the compiler into built/local.
gulp clean             # Delete the built compiler.
gulp LKG               # Replace the last known good with the built one.
                       # Bootstrapping step to be executed when the built compiler reaches a stable state.
gulp tests             # Build the test infrastructure using the built compiler.
gulp runtests          # Run tests using the built compiler and test infrastructure.
                       # You can override the specific suite runner used or specify a test for this command.
                       # Use --tests=<testPath> for a specific test and/or --runner=<runnerName> for a specific suite.
                       # Valid runners include conformance, compiler, fourslash, project, user, and docker
                       # The user and docker runners are extended test suite runners - the user runner
                       # works on disk in the tests/cases/user directory, while the docker runner works in containers.
                       # You'll need to have the docker executable in your system path for the docker runner to work.
gulp runtests-parallel # Like runtests, but split across multiple threads. Uses a number of threads equal to the system
                       # core count by default. Use --workers=<number> to adjust this.
gulp baseline-accept   # This replaces the baseline test results with the results obtained from gulp runtests.
gulp lint              # Runs eslint on the TypeScript source.
gulp help              # List the above commands.
```

Usage

```
node built/local/tsc.js hello.ts
```

Roadmap

For details on our planned features and future direction please refer to our [roadmap](https://github.com/microsoft/TypeScript/wiki/Roadmap).