简介

TypeScript 是 JavaScript 的超集，是微软公司开发。主要功能是为 JavaScript 添加类型系统。

类型是人为添加的一种编程约束和用法提示。

在语法上，JavaScript 属于动态类型语言，TypeScript 引入了一个更强大、更严格的类型系统，属于静态类型语言。

1. 静态类型的优点

（1）有利于代码的静态分析。

不必运行代码，就可以确定变量的类型，从而推断代码有没有错误。这就叫做代码的静态分析。在开发阶段运行静态检查，就可以发现很多问题，避免交付有问题的代码，大大降低了线上风险。

（2）有利于发现错误。

每个值、每个变量、每个运算符都有严格的类型约束，TypeScript 就能轻松发现拼写错误、语义错误和方法调用错误，节省时间。

```
let obj = { message: '' };
obj.messege; // Property 'messege' does not exist on type '{ message: string; }'.
Did you mean 'message'?
```

```
const a = 0;
const b = true;
const result = a + b; // Operator '+' cannot be applied to types 'number' and
'boolean'.
```

```
function hello() {
  return 'hello world';
}

hello().find('hello'); // Property 'find' does not exist on type 'string'.
```

（3）更好的 IDE 支持，做到语法提示和自动补全。

IDE（比如 VSCode）一般都会利用类型信息，提供语法提示功能和自动补全功能（只键入一部分的变量名或函数名，编辑器补全后面的部分）。

（4）提供了代码文档。

类型信息可以部分替代代码文档，解释应该如何使用这些代码。

（5）有助于代码重构。

类型信息大大减轻了重构的成本。越是大型的、多人合作的项目，类型信息能够提供的帮助越大。

2. 静态类型的缺点

（1）丧失了动态类型的代码灵活性。

（2）增加了编程工作量。

不仅需要编写功能，还需要编写类型声明。

（3）更高的学习成本。

（4）引入了独立的编译步骤。

将 TypeScript 代码转成 JavaScript 代码，在 JavaScript 引擎运行。

（5）兼容性问题。

过去大部分 JavaScript 项目都没有做 TypeScript 适配。

TypeScript 不一定适合那些小型的、短期的个人项目。