# TypeScript学习笔记

# 什么是 TypeScript

TypeScript 是 JavaScript 的类型的超集，它可以编译成纯 JavaScript。编译出来的 JavaScript 可以运行在任何浏览器上。TypeScript 编译工具可以运行在任何服务器和任何系统上。

## 为什么选择 TypeScript（有点和缺点）

优点：

1. 增加了代码的可读性和可维护性
2. 具有很强的包容性
3. 拥有活跃的社区

缺点：

* 有一定的学习成本，需要理解接口（Interfaces）、泛型（Generics）、类（Classes）、枚举类型（Enums）等前端工程师可能不是很熟悉的概念
* 短期可能会增加一些开发成本，毕竟要多写一些类型的定义，不过对于一个需要长期维护的项目，TypeScript 能够减少其维护成本
* 集成到构建流程需要一些工作量
* 可能和一些库结合的不是很完美

安装：npm install –g typescript

编译命令：tsc hello.ts

编辑器：vscode，sublime，webstorm，Eclipse 等

原始数据类型：

原始数据类型包括：基本数据类型（Boolean，null，undefined，Number,string）和复杂数据类型[Object]（官方文档：原始数据类型和对象类型）

新增==》 空值

JavaScript 没有空值（Void）的概念，在 TypeScript 中，可以用 void 表示没有任何返回值的函数：

function alertName(): void {

alert('My name is Tom');

}

let unusable: void = undefined;

其中null和undefined 是所有类型的类型的子类型（即使变量已经定义了类型，按照ts语法，会报错，但是由于null和undefined是所有类型的子类型，依旧可以正确编译）

新增==》任意值

任意值（Any）用来表示允许赋值为任意类型。

什么事任意类型：==》 如果是一个普通类型，在赋值过程中改变类型是不被允许的，但如果是 any 类型，则允许被赋值为任意类型。

类型推论：在没有定义类型的时候，软件编译过程中会给未被定义的变量推测一个类型，这就是类型推论

联合类型：表示取值可以为多种类型中的一种【let myFavoriteNumber: string | number;】联合类型使用 | 分隔每个类型。这里的 let myFavoriteNumber: string | number 的含义是，允许 myFavoriteNumber 的类型是 string 或者 number，但是不能是其他类型。

对象的类型——接口==》

interface Person {

name: string;

age: number;

}

let tom: Person = {

name: 'Tom',

age: 25

};

可选属性==》 有时我们希望不要完全匹配一个形状，那么可以用可选属性（可以理解为，定义的属性可以没有但是不能增加新的属性）

任意属性：有时候我们希望一个接口允许有任意的属性，可以使用如下方式：