# TS简介

## TS是谁



## 为什么要学TS

[为什么要用TypeScript？- html中文网](https://www.html.cn/qa/javascript/11118.html)

TypeScript的设计目的应该是解决JavaScript的“痛点”：弱类型和没有命名空间，导致很难模块化，不适合开发大型程序。

TypeScript:

1. 引入强类型
2. 可以支持接口，泛型等

## Hello World

# yarn global add typescript

Hello.ts

console.log("Hello World!");

# tsc hello.ts

# node hello.js

# 类型

## 申明方式

## 类型种类

共12种类型

|  |  |  |
| --- | --- | --- |
| **类型** | **例子** | **描述** |
| Number |  |  |
| String |  |  |
| Boolean |  |  |
| 字面量 | 值本身  let d: 'male' | 'female' | 1 | 0 | Ts新增类型 |
| Any |  | Ts新增类型 |
| unknown |  | 安全类型的any  Ts新增类型 |
| void |  | Ts新增类型 |
| never |  | Ts新增类型 |
| object | Js里面typeof function可以返回function，  在ts里面function就归类的object吧 |  |
| array |  |  |
| tuple | 对数组类型，长度进行限制 | Ts新增类型 |
| enum |  | Ts新增类型 |

感觉js的基本类型归纳有点乱：

基础类型：字符串值，数值，布尔值，数组，对象。

三个特殊的：undefined, null, function

而且typeof用起来不好用，而且是非常困惑, 只能强行记忆的那种：

typeof undefined // undefined

typeof null // object

typeof {name:'Bill', age:62} // 返回 "object"

typeof [1,2,3,4] // 返回 "object"

typeof null // 返回 "object"

typeof function myFunc(){} // 返回 "function"

# 编译选项

上面的例子中，可以编译ts；得一个个文件去指定，很烦的

# tsc hello.ts

# 检测改变并自动编译为js，但是窗口不能停

# tsc hello.ts -w

## Tsconfig.json

# webpack打包ts

实际项目，肯定离不开webpack, 那么就需要在webpack中修改：

# 脚手架支持ts

。。。 待补充

# Reference

[尚硅谷2021版TypeScript教程（李立超老师TS新课）——Bilibili](https://www.bilibili.com/video/BV1Xy4y1v7S2?p=1)

这个教程感觉很不好；

1. Ts类型讲的不错
2. Webpack部分太罗嗦了，实际项目肯定不直接操作webpack的
3. 实战例子也没通过对比体现出ts的优势所在，反而是让我用力于贪吃蛇的逻辑实现上去了。

6分评价吧；