01 | 我们为什么要用 TypeScript

周爱民（Aimingoo）

目录

基础环境

1. Node.js + VSCode
   - https://nodejs.org/
   - https://typescriptlang.org/

2. Typescript playground

简单操作

1. TypeScript简单使用

```
# 在全局安装typescript
> npm install -g typescript
# （编译指定.ts文件）
> tsc MyTest.ts
```

2. TypeScript的项目操作

```
# 将当前目录初始化为一个TypeScript项目目录（将生成./tsconfig.json）
> tsc --init --module es2020 --target es2020 --lib es2020
# 编译（使用当前目录中的默认配置./tsconfig.json）
> tsc
# 编译（使用指定配置编译）
> tsc -p ./tsconfig.json  # 适用于多个配置文件的情况
```

3. 在Node.js中支持TypeScript

```
# 将当前目录初始化为Node.js项目目录（生成package.json，如有，则可略过）
> npm init -y
# 安装TypeScript支持
> npm install --save-dev @types/node
```

总结

1. TypeScript 的主要作用

- 在代码上显式标注类型，可以让代码更易读
- 在开发和编译时，通过类型推断和静态类型检查可以显著提高代码质量

2. TypeScript 有两种主要用法

- 编译指定文件（例如`tsc MyTest.ts`）
- 创建和编译TS项目（例如用`tsc --init ...`）

3. 简单的开发环境及其它

- 基础环境：Node.js + VSCode，Or TypeScript playground
- 安装TypeScript
- 安装VSCode插件（chinese language, reload, etc...）
- 简单命令行使用，和VSCode的简单配置
  - tsc, npm, npx, node
- 基本的类型概念：类型安全、类型标注、类型推断、强类型、类型检查等

名词/概念

名词、术语

类型兼容、类型兼容性：	Type Compaction, Type Compatibility
控制台、终端、外壳程序：	Console, Terminal, Shell
弱类型、强类型与静态类型：	Weak vs. strong typing, and Static typing
类型系统：	Type systems
静态检查、静态类型检查、类型检查器：	Static checking, Static type checking, Type checker

概念

- JavaScript是一个弱类型语言（*JavaScript is a weakly typed language*）。
  @see: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures
- TypeScript是一个带有编译期类型检查的JavaScript的运行环境（*TypeScript is JavaScript's runtime with a compile-time type checker*）。
  @see: https://typescriptlang.org/docs/handbook/typescript-from-scratch.html
- 在执行前检查程序是否有错误，称为静态检查（*Detecting errors in code without running it is referred to as static checking*）；在这个阶段根据值的类型进行检查（*based on the kinds of values*），则称为静态类型检查。
  @see: https://typescriptlang.org/docs/handbook/typescript-from-scratch.html
- 类型系统是由规则集构成的逻辑系统，用于描述如何将类型分配给名字、符号等术语（*Is a logical system comprising a set of rules that assigns … type to every term*）。
  @see: https://en.wikipedia.org/wiki/Type_system

Q&A

Q: 我怎知道当前的Node.js可以安装哪个版本的TypeScript

@see https://nx.dev/packages/workspace/documents/nx-nodejs-typescript-version-matrix

Q: 如何安装Bash的控制台

A: 安装一下“Git for Windows”，再启动VSCode就行了。@see https://gitforwindows.org

Q: VSCode的主边栏为什么在右边

A: 在VSCode菜单“View / Appearance / Move Primary Side Bar Left”中切换

Q: 为什么不推荐使用 TypeScript playground

A: 演练场（playground）在某些配置项上不能自如地设置，例如 `--lib`。不过如果你习惯使用它的话，本课程的绝大多数练习也可以在其中完成。此外，由于演练场的结果与生产环境存在差异，本课程中的某些示例在演练场中可能会遇到意外的结果，因此若有必要，请切换回 VSCode。

Q&A

Q: 什么是“隐式的any类型”，它为什么会存在?

A: TypeScript 默认将无法确定的类型标注为 any 类型，以便兼容 JavaScript 中的动态类型和弱类型系统。如果一个 any 类型是由 TypeScript 引擎自行推断出来的，则它称为“隐式的any类型”（反之则是由用户代码显式标注的）。任何隐式的 any 类型都是不安全的，应尽量避免，因此在 tsonfig.json 中有一个 `noImplicitAny` 配置项，用于在遇到隐式的 any 类型时抛出编译错误。

Q: 如何检查开发环境的版本和安装目录

```
# 版本
> node -v
> npm -v
> tsc -v
# 安装路径（在Windows控制台中用where）
> which node   # or npm, tsc, etc.
```

课后作业

1. 将 TypeScript 配置改回默认项

2. 在 Node.js 环境下为示例项目（MyTest.js）完善功能
   - 使用 readline 模块，为 inputSomething() 添加一个输入功能。
   - 简单实现 loadFromDatabase()，例如，从 .json/.js 中装载和返回数据。


极客时间
THANKS