TypeScript 押题

TS 和 JS 的区别是什么？有什么优势?

any、unknown、never 的区别是什么?

type 和 interface 的区别是什么?

TS 工具类型 Partial、Required 等的作用和实现?

TS 和 JS 的区别是什么？有什么优势?

1. 语法层面：TypeScript = JavaScript + Type（TS 是 JS 的超集）
2. 执行环境层面：浏览器、Node.js 可以直接执行 JS，但不能执行 TS（Deno 可以执行 TS）
3. 编译层面：TS 有编译阶段，JS 没有编译阶段（只有转译阶段和 lint 阶段）
4. 编写层面：TS 更难写一点，但是**类型更安全**
5. 文档层面：TS 的代码写出来就是文档，IDE 可以完美**提示**。JS 的提示主要靠 TS

其他……自己搜一下博客

any、unknown、never 的区别是什么?

any V.S. unknown

二者都是顶级类型（top type），任何类型的值都可以赋值给顶级类型变量：

```
let foo: any = 123; // 不报错
let bar: unknown = 123; // 不报错
```

但是 unknown 比 any 的类型检查更严格，any 什么检查都不做，unknown 要求先收窄类型：

```
const value: unknown = "Hello World";
const someString: string = value;
// 报错: Type 'unknown' is not assignable to type 'string'.(2322)
```

```
const value: unknown = "Hello World";
const someString: string = value as string; // 不报错
```

如果改成 any，基本在哪都不报错。所以能用 unknown 就优先用 unknown，类型更安全一点。

never

never 是底类型，表示不应该出现的类型，这里有一个尤雨溪给出的例子：

```ts
interface A {
  type: 'a'
}

interface B {
  type: 'b'
}

type All = A | B

function handleValue(val: All) {
  switch (val.type) {
    case 'a':
      // 这里 val 被收窄为 A
      break
    case 'b':
      // val 在这里是 B
      break
    default:
      // val 在这里是 never
      const exhaustiveCheck: never = val
      break
  }
}
```

现在你应该理解什么是「不应该出现的类型」了吧。

type 和 interface 的区别是什么?

官方给出的文档说明:

1. 组合方式：interface 使用 extends 来实现继承，type 使用 & 来实现联合类型。
2. 扩展方式：interface 可以重复声明用来扩展，type 一个类型只能声明一次
3. 范围不同：type 适用于基本类型，interface 一般不行。
4. 命名方式：interface 会创建新的类型名，type 只是创建类型别名，并没有新创建类型。

其他……建议搜一下博客。

TS 工具类型 Partial、Required 等的作用和实现?

1. 将英文翻译为中文。
    a. Partial 部分类型
    b. Required 必填类型
    c. Readonly 只读类型
    d. Exclude 排除类型
    e. Extract 提取类型
    f. Pick/Omit 排除 key 类型
    g. ReturnType 返回值类型
2. 举例说明每个工具类型的用法。