**TypeScript 基础类型**

|  |  |  |
| --- | --- | --- |
| **数据类型** | **关键字** | **描述** |
| 任意类型 | any | 声明为 any 的变量可以赋予任意类型的值。 |
| 数字类型 | number | 双精度 64 位浮点值。它可以用来表示整数和分数。  let binaryLiteral: number = 0b1010; // 二进制  let octalLiteral: number = 0o744; // 八进制  let decLiteral: number = 6; // 十进制  let hexLiteral: number = 0xf00d; // 十六进制 |
| 字符串类型 | string | 一个字符系列，使用单引号（**'**）或双引号（**"**）来表示字符串类型。反引号（**`**）来定义多行文本和内嵌表达式。  let name: string = "Runoob";  let years: number = 5;  let words: string = `您好，今年是 ${ name } 发布 ${ years + 1} 周年`; |
| 布尔类型 | boolean | 表示逻辑值：true 和 false。  let flag: boolean = true; |
| 数组类型 | 无 | 声明变量为数组。  // 在元素类型后面加上[]  let arr: number[] = [1, 2];  // 或者使用数组泛型  let arr: Array<number> = [1, 2]; |
| 元组 | 无 | 元组类型用来表示已知元素数量和类型的数组，各元素的类型不必相同，对应位置的类型需要相同。  let x: [string, number];  x = ['Runoob', 1]; // 运行正常  x = [1, 'Runoob']; // 报错  console.log(x[0]); // 输出 Runoob |
| 枚举 | enum | 枚举类型用于定义数值集合。  enum Color {Red, Green, Blue};  let c: Color = Color.Blue;  console.log(c); // 输出 2 |
| void | void | 用于标识方法返回值的类型，表示该方法没有返回值。  function hello(): void {  alert("Hello Runoob");  } |
| null | null | 表示对象值缺失。 |
| undefined | undefined | 用于初始化变量为一个未定义的值 |
| never | never | never 是其它类型（包括 null 和 undefined）的子类型，代表从不会出现的值。 |

**Any 类型**

任意值是 TypeScript 针对编程时类型不明确的变量使用的一种数据类型

let x: any = 1;    // 数字类型

x = 'I am who I am';    // 字符串类型

x = false;    // 布尔类型

**Null 和 Undefined**

**null**

在 JavaScript 中 null 表示 "什么都没有"。

null是一个只有一个值的特殊类型。表示一个空对象引用。

用 typeof 检测 null 返回是 object。

**undefined**

在 JavaScript 中, undefined 是一个没有设置值的变量。

typeof 一个没有值的变量会返回 undefined。

**never 类型**

never 是其它类型（包括 null 和 undefined）的子类型，代表从不会出现的值。这意味着声明为 never 类型的变量只能被 never 类型所赋值，在函数中它通常表现为抛出异常或无法执行到终止点（例如无限循环），示例代码如下：

let x: never;

let y: number;

// 运行错误，数字类型不能转为 never 类型

x = 123;

// 运行正确，never 类型可以赋值给 never类型

x = (()=>{ throw new Error('exception')})();

// 运行正确，never 类型可以赋值给 数字类型

y = (()=>{ throw new Error('exception')})();

// 返回值为 never 的函数可以是抛出异常的情况

function error(message: string): never {

    throw new Error(message);

}

// 返回值为 never 的函数可以是无法被执行到的终止点的情况

function loop(): never {

    while (true) {}

}