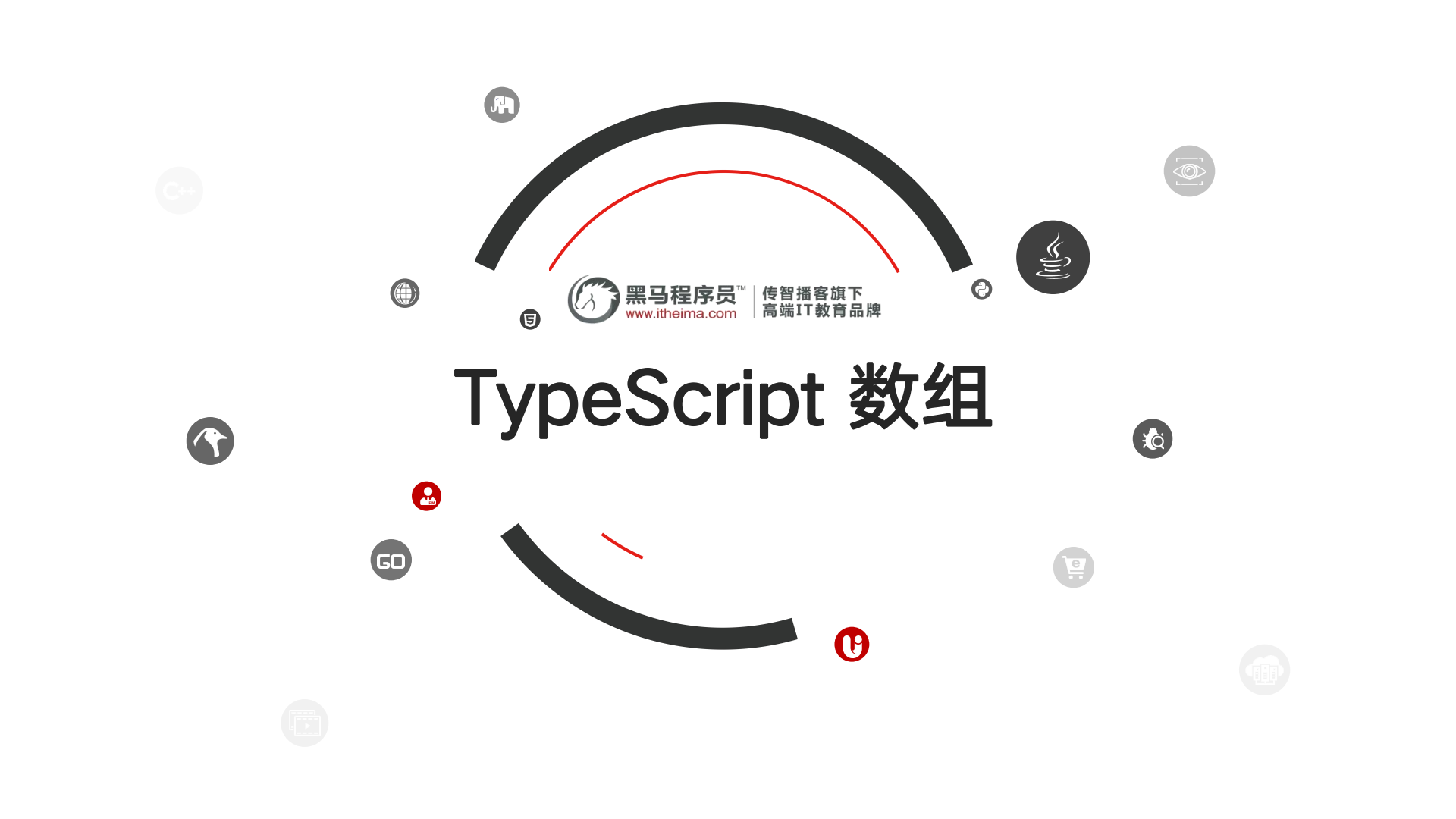

TypeScript 数组

目录 Contents

1. 数组概述

问题1：存储一个人的名字，怎么存？ 声明一个字符串类型的变量

```
let name1: string = '迪丽热巴'
```

问题2：存储三个人的名字，怎么存？ 声明三个字符串类型的变量

```
let name1: string = '迪丽热巴'
let name2: string = '古力娜扎'
let name3: string = '马尔扎哈'
```

问题3：如何是存储一个班级中所有人的名字呢？

存储多个数据时，声明多个变量就太繁琐了。

1. 数组概述

数组，是用于存放多个数据的集合。

有数组：只需要使用一个数组（[]），就可以存储任意多个数据。

```
let names: string[] = ['迪丽热巴', '古力娜扎', '马尔扎哈']
```

没有数组：存储三个人的名字，就需要三个字符串类型的变量。

```
let name1: string = '迪丽热巴'
let name2: string = '古力娜扎'
let name3: string = '马尔扎哈'
```

注意：数组中，通常是相同类型的数据。

目录 Contents

2. 创建数组

创建数组有两种语法形式。

- 语法一（推荐）：

```
let names: string[] = []
```

[]（中括号）表示数组。如果数组中没有内容，就是一个空数组。

2. 创建数组

创建数组有两种语法形式。

- 语法一（推荐）：

```
let names: string[] = []
```

[]（中括号）表示数组。如果数组中没有内容，就是一个空数组。

数组的类型注解由两部分组成：类型+[]。此处表示字符串类型的数组（只能出现字符串类型）。

```
let names: string[] = ['迪丽热巴']
```

2. 创建数组

创建数组有两种语法形式。

- 语法一（推荐）：

```
let names: string[] = []
```

[]（中括号）表示数组。如果数组中没有内容，就是一个空数组。

数组的类型注解由两部分组成：类型+[]。此处表示字符串类型的数组（只能出现字符串类型）。

```
let names: string[] = ['迪丽热巴', '古力娜扎', '马尔扎哈']
```

数组，多个元素之间使用逗号（,）分隔。

数组中的每一项内容称为：元素。

2. 创建数组

创建数组有两种语法形式。

- 语法二（不推荐）：

```
let names: string[] = new Array()
```

功能与 `[]` 相同，但是更加繁琐：

```
let names: string[] = []
```

数组中有数据时：

```
let names: string[] = new Array('迪丽热巴', '古力娜扎', '马尔扎哈')
// 相当于：
let names: string[] = ['迪丽热巴', '古力娜扎', '马尔扎哈']
```

目录 Contents

3. 数组长度和索引

3.1 概述

生活中，我们经常会排队（比如：排队吃饭）。

队伍的特征：1 长度　2 顺序和序号（队伍中的每个人）。

我们可以把数组想象成这个队伍，因为数组也有长度，也有顺序并且数组中的每个元素也有序号。

3. 数组长度和索引

3.2 数组长度

数组长度：表示数组中元素的个数，通过数组的 `length` 属性来获取。

```
let foods: string[] = ['煎饼', '馒头', '米饭']
```

获取数组长度：

```
console.log(foods.length) // 3
```

3. 数组长度和索引

3.2 数组索引

数组中的每个元素都有自己的序号。

我们把数组中元素的序号，称为：索引（下标），数组中的元素与索引一一对应。

注意：数组索引是从 0 开始的。

```
let foods: string[] = ['煎饼', '馒头', '米饭']
// 数组的索引分别为:        0      1      2
```

问题：该数组的长度（length）和最大索引之间有什么关系？ 最大索引为：length - 1

3. 数组长度和索引

总结：

数组是有序的集合，用来存储多个数据。

问题1：如何获取数组长度？ `foods.length`

问题2：数组索引是从几开始的？ 索引从 0 开始

目录 Contents

4. 取值和存值

4.1 取值

从数组中，获取到某一个元素的值，就是从数组中取值。（比如，获取最爱的食物 – 煎饼）

```
let foods: string[] = ['煎饼', '馒头', '米饭']
// 数组的索引分别为:        0      1      2
```

数组中的元素与索引是一一对应的，通过索引获取到某一个元素的值。

语法：

```
数组名称[索引]
```

比如，获取到最爱的食物 – 煎饼：

```
console.log(foods[0]) // 煎饼
```

4. 取值和存值

4.2 存值

如果要修改数组中某个元素的值，就要使用数组存值。（比如，不爱吃馒头，将馒头替换为包子）

```
let foods: string[] = ['煎饼', '馒头', '米饭']
// 数组的索引分别为:        0      1      2
```

技巧：先获取到要修改的元素，然后，再存值。

语法：

```
数组名称[索引] = 新值
```

比如，将馒头替换为包子：

```
foods[1] = '包子'
console.log(foods) // ['煎饼', '包子', '米饭']
```

4. 取值和存值

4.3 添加元素

存值的语法是：数组名称[**索引**] = 新值，根据索引是否存在，有两种功能：1 修改元素　2 添加元素。

```
let foods: string[] = ['煎饼', '馒头', '米饭']
// 数组的索引分别为:        0      1      2
```

1. 如果索引存在，就表示：修改元素。

```
foods[1] = '包子'
```

2. 如果索引不存在，就表示：添加元素。

```
foods[3] = '油泼面'
console.log(foods) // ['煎饼', '馒头', '米饭', '油泼面']
```

添加元素的通用写法：数组名称[数组长度] = 新值

目录 Contents

5. 遍历数组

遍历数组，也就是把数组中的所有元素挨个获取一次（比如，计算数组中所有数字的和）。

```
let nums: number[] = [100, 200, 300]
// 索引分别为：          0    1    2
```

通过数组取值的方式，就可以一个个取出来：

```
console.log(nums[0]) // 100
console.log(nums[1]) // 200
console.log(nums[2]) // 300
```

存在问题：太繁琐，相似的代码重复多次。

重复做某件事情，可以使用 `for` 循环。

重复取值的规律：索引号自增（每次加1），而 `for` 循环的计数器`i`也是自增的。

5. 遍历数组

遍历数组，也就是把数组中的所有元素挨个获取一次（比如，计算数组中所有数字的和）。

```
let nums: number[] = [100, 200, 300]
// 索引分别为:          0    1    2
```

推荐，使用for循环遍历数组：

```
for (let i: number = ?; i <= ?              ; i++) {
  console.log(nums[i])
}
```

注意1：因为数组索引是从0开始的，所以计数器i的默认值为0。

5. 遍历数组

遍历数组，也就是把数组中的所有元素挨个获取一次（比如，计算数组中所有数字的和）。

```
let nums: number[] = [100, 200, 300]
// 索引分别为:          0    1    2
```

推荐，使用for循环遍历数组：

```
for (let i: number = 0; i <= ?              ; i++) {
  console.log(nums[i])
}
```

注意1：因为数组索引是从0开始的，所以计数器i的默认值为0。

注意2：应该根据数组长度来计算，公式为数组长度减一，也就是：nums.length - 1（最大索引）。

5. 遍历数组

遍历数组，也就是把数组中的所有元素挨个获取一次（比如，计算数组中所有数字的和）。

```
let nums: number[] = [100, 200, 300]
// 索引分别为:          0    1    2
```

推荐，使用for循环遍历数组：

```
for (let i: number = 0; i <= nums.length - 1; i++) {
  console.log(nums[i])
}
```

注意1：因为数组索引是从0开始的，所以计数器i的默认值为0。

注意2：应该根据数组长度来计算，公式为数组长度减一，也就是：nums.length - 1（最大索引）。

优势：不管数组中元素的数量怎么变化，for循环的判断条件不需要改动。

5. 遍历数组

遍历数组，也就是把数组中的所有元素挨个获取一次（比如，计算数组中所有数字的和）。

```
let nums: number[] = [100, 200, 300]
// 索引分别为:          0    1    2
```

推荐，使用for循环遍历数组：

```
for (let i: number = 0; i <= nums.length - 1; i++) {
  console.log(nums[i])
}
```

简化判断条件（计数器i的值为整数，所以，i<=2 与 i<3 作用相同）：

```
for (let i: number = 0; i < nums.length; i++) {
  console.log(nums[i])
}
```

5. 遍历数组

总结：

遍历数组，也就是把数组中的所有元素挨个获取一次。

问题1：如果要遍历数组应该使用什么语句？ `for`循环语句

问题2：for循环计数器的默认值是多少？ 默认值为：`0`

问题3：for循环的判断条件是什么？ `i < nums.length`

传智播客旗下高端IT教育品牌