# 在ts中函数需要约束参数的类型和返回值的类型

# ts中定义函数的格式

## 函数名称（参数1:数据类型，参数2:数据类型）: 返回值类型{

## 函数体

## }

## 如

|  |
| --- |
| function getBigger(a:number,b:number):number {  return a<b ? b:a }  console.log(getBigger(30,50)) //输出50 |

## 如果没有返回值需要写void

|  |
| --- |
| function sayHello():void {  console.log('hello ts') }  sayHello() //输出hello ts |

# ts函数中使用可变参数，使用...变量名称:数据类型[]的格式

|  |
| --- |
| //ts中使用...变量名称:数据类型[]的格式来定义可变参数 function sumUp(...nums:number[]):number {  let sum = 0  nums.forEach(item=>{  sum+=item  })  return sum }  // console.log(sumUp(10,20,30)) console.log(sumUp(1.1,2.2,3.3,4.4,5,6,7)) |

## 上面这个函数可以就是任何数量的数字之和，类似的有下面的代码

|  |
| --- |
| function sayHelloToPeople(...names:string[]) {  for (const name of names) {  console.log("Hello " + name)  } }  // sayHelloToPeople('Jack','Mary','James','Franky','Tracy')  function anyMax(...nums:number[]):number {  let max = 0  for (const num of nums) {  if (num > max) {  max = num  }  }  return max }  console.log("max is :"+ anyMax(1,20,200,20,300,6)) |

### 注意：可变参数一定要放在参数列表的最后

# ts函数中的可选参数或者可空参数

## 这种参数的后面跟一个?,

## 注：可选参数一定要放在所以必选参数之后，除非在他后面的参数有默认值

|  |
| --- |
| function eatFood(mainfood:string,sidefood?:string) {  if(sidefood){  console.log(`very good,eating ${mainfood} with ${sidefood}`)  } else{  console.log(`my god! only eating ${mainfood}?!!`)  } }  // eatFood('rice','fish') // very good,eating rice with fish eatFood('potato')// my god! only eating potato?!! |

### 如：可选参数可以放在有默认值的参数前面

|  |
| --- |
| function testOptional(x:number,y?:number,z:number=10) {  if(y){  return x+y+z  }else{  return x+z  } }  console.log(testOptional(10)) |

### 这样子也是可以的

|  |
| --- |
| function testOptional(x:number,y:number=10,z?:number) {  if(z){  return x+y+z  }else{  return x+y  } }  console.log(testOptional(10)) |

# 对象作为函数参数

## 如下面的写法

|  |
| --- |
| function test(params:{x:number,y:number}) {  return params.x + params.y }  console.log(test({x: 10, y: 20})); //输出30 |

## 也可以约束返回值

|  |
| --- |
| function test(params:{x:number,y:number}):number {  return params.x + params.y }  console.log(test({x: 10, y: 20})); |

# ts中函数的有默认值从参数

## 可以给函数的某些参数设置默认值，但是带有默认值的参数一定要放在后面声明

|  |
| --- |
| //函数的默认值参数 function defaultparam(a:number,b:number=20):number {  return a+b }  // console.log(defaultparam(100)) //输出120 console.log(defaultparam(100,200)) //输出300 |

### 当然如果函数只有一个参数，给这个参数赋默认值，那么你调用函数的时候就可以不传递参数

|  |
| --- |
|  |

# ts中的函数类型

## 当我们用一个变量来接受一个函数的时候，我们也可以约束这个变量的类型，如下

|  |
| --- |
|  |

### 上面的例子说明这个变量可以接受一个没有返回值的函数void全面的箭头并不是箭头函数，void后面的是箭头函数

## 如果需要约束参数，可以怎么写

|  |
| --- |
|  |

### 格式如下

|  |
| --- |
|  |

### 如：

|  |
| --- |
|  |