Typescript

只读属性 readonly ; interface point{ readonly x:number }; //x只读

ReadonlyArray<number> = a;

**function** **f**(this: void) { // 确保`this`在这个独立的函数中无法使用 }

一、基本类型：

布尔：let isDone:Boolean=true;

数字：let n1:number=3;//十进制，十六进制，八进制，二进制

字符串：let name:string=’lilei’; let age:string=’my age is ${age + 1}’;

数组：let list:number[]=[1,2,3]; let list:Array<number>=[1,2,3];

元组：let x:[string,number];

x=[‘hello’,10];//OK与x=[10,’hello’];//Error

x[0]substr(1);//OK与x[1].substr(1);//Error

访问一个越界元素，会使用联合型替代：

X[3]=’world’;//OK与x[6]=true;//Error

枚举：1 -- enum Color {Red, Green, Blue};let c:Color=Color.Green;

2 -- enum Color {Red = 1, Green = 3, Blue=8}; let c:Color=Color.Green;默认是从0开始

3 – enum Color {Red=1,Green,Blue}; let colorName:string=Color[2];

任意值：let list:any[]=[1,true,’free’];list[1]=100;

空值：与any类型相反，它表示没有任何类型。

1 – function test():void{ console.log(‘test’) };

2 – let unusable:void=undefined; 当对变量使用时，只能赋值undefined和null

Null和Undefined：与void相同，意义不大

1 – 默认null和undefined是所有类型的子类型,所以它们可以赋值给number类型等

2 – 在使用 –strictNullChecks标记时，let a:null=null;只能是null与let b:undefined=undefined;只能是undeinfed

Never:表示那些用不存在的值的类型。它是所有类型的子类型，但any不能赋值给never

类型断言：1 -- <>;let val:any=’test’;let strlength:number=(<string>val).length;

2 – as; let val:any=’test’;let strlength:number=(val as string).length;

二、变量声明

Var,let,const

解构：1 – 基本 let input=[,2];let [one,two]=input;

2 –作用函数参数,function f([one,two]:[number,number]){}

3—let [one,…rest]=[1,2,3,4]; rest就是[2,3,4];

4 – 忽略 [one]=[1,2,3,4]; one是1;let [,two,,four]=[1,2,3,4];

5 – 对象解构 let o={a:’foo’,b:12,c:’bar’};let {a,b}=o;

6 – 属性重名命let {a:newName1}=o;let {a}:{a:string}=o;

7 – 默认值 function do(obj:{a:tring,b?:number}){let {a:b=1001}=obj;}

函数声明：type C = {a:string,b?:number}; function f({a,b}:C):void{};

三、接口

初探：interface LabelledValue{label:string}

Function printLabel(labelledValue:LabelledValue){ console.log(labelledValue.label); }  
 let myObj={size:10,label:’Size 10 object’};//只要有label属性就行 printLabel(myObj);

可选属性：interfaceSquareConfig { color?:string,width?:number };  
 function createSquare(config:SquareConfig):{color:string;area:number}{

Let newSquare={color:’white’,area:100};

If(config.color) newSquare.color=config.color;

If(config.width) newSquare.area=config.width\*config.width;

return newSquare;

}

Let mySquare= createSquare({color:’black’});

只读属性：interface Point{ readonly x:number,readonly y:number };

let p1:Point={x:10,y:20}; p1.x=5; //