箭头函数-非原创

基本定义

① 简单写法，一个变量一个表达式返回单值

```
var f = v => v;

// 等同于
var f = function (v) {
  return v;
};
```

② 空参数

```
var f = () => 5;
// 等同于
var f = function () { return 5 };
```

③ 多参数

```
var sum = (num1, num2) => num1 + num2;
// 等同于
var sum = function(num1, num2) {
  return num1 + num2;
};

```

④ 函数体中多条语句，使用return语句返回并使用大括号将函数体包围起来。

```
var sum = (num1, num2) => {
        num1+=1;
        return num1 + num2; }
```

⑤ 函数结构体也可以不返回

```
//一 单行不返回使用void关键字标识
let fn = () => void doesNotReturn();
// 二 多行语句不使用return
var sum = (num1, num2) => {
num1+=1;
num1 + num2; }
```

⑥ 如果箭头函数直接返回一个对象，由于大括号被解释为代码块，必须在对象外面加上括号，否则会报错。

```
// 报错
let getTempItem = id => { id: id, name: "Temp" };

// 不报错
let getTempItem = id => ({ id: id, name: "Temp" });
```

箭头函数与变量解构结合使用

箭头函数可以与变量解构结合使用。

```
const full = ({ first, last }) => first + ' ' + last;

// 等同于
function full(person) {
  return person.first + ' ' + person.last;
}

```

箭头函数的使用范例

箭头函数使得表达更加简洁。

```
const isEven = n => n % 2 === 0;
const square = n => n * n;
```

简化回调函数

```
// 正常函数写法
[1,2,3].map(function (x) {
  return x * x;
});

// 箭头函数写法
[1,2,3].map(x => x * x);
```

```
// 正常函数写法
var result = values.sort(function (a, b) {
  return a - b;
});

// 箭头函数写法
var result = values.sort((a, b) => a - b);
```

rest 参数与箭头函数结合的例子

```
const numbers = (...nums) => nums;

numbers(1, 2, 3, 4, 5)
// [1,2,3,4,5]

const headAndTail = (head, ...tail) => [head, tail];

headAndTail(1, 2, 3, 4, 5)
// [1,[2,3,4,5]]
```