ECMASript和JavaScript

* ECMA是实现，js是标准

兼容性

ES6（ES2015）--IE10+、chrome、firefox、移动端、node.js

编译、转换

1. 在线转换

<script scr=’brower.js’ charset=’utf-8'></script>

<script type=’text/babel’>

……

</script>

1. 提前编译

（babel==browser.js）

ES6:

1. 变量
2. 函数
3. 数组
4. 字符串
5. 面向对象
6. Promise
7. Generator（把同步操作拆为异步操作，对promise的封装）
8. 模块化
9. 变量

var

1. 可以重复声明

var a=5；

var a=23；

1. 无法限制修改
2. 没有块级作用域

If（）{

}

for（）{

}

新特性：

let 变量（不能重复声明，块级作用域）-可以重复修改

const 常量（不能重复声明，块级作用域）-不可以重新修改

1. 箭头函数

标准函数：

function 名字（）{

}

箭头函数：

1. 如果只有一个参数，（）可以省
2. 如果只有一个return，{}可以省

三、函数参数

1. 参数扩展/展开

2. 默认参数

参数扩展：

1. 收集剩余的参数

*function* show(*a*, *b*, ...*args*) { }

\*rest parameter必须是最后一个

2. 展开数组

*let* arr = [1, 2, 3];

show2(...arr);

*function* show2(*a*, *b*, *c*) {

*console*.log(a);

*console*.log(b);

*console*.log(c);

}

*let* arr1 = [1, 2, 3];

*let* arr2 = [4, 5, 6];

*let* arr3 = [...arr1, ...arr2];

展开后的效果，就跟直接把数组的内容拿出来一样。

默认参数

*function* show3(*a*, *b* = 2, *c* = 3) {

*console*.log(a);

*console*.log(b);

*console*.log(c);

}

show3(1);

四、解构赋值

1.左右两边结构一样

2.右边必须是个东西

3.声明和赋值不能分开

*let* [a, b, c] = [1, 2, 3];

*console*.log(a + "/" + b + "/" + c);

*let* { d, e } = { d: 12, e: 23 };

*console*.log(e + "/" + e);

*let* [{ f, g }, [num1, num2, num3], num, str] = [{ f: 45, g: 34 }, [2, 3, 4], 22, 'sssss'];

*console*.log(f, g, num1, num2, num3, num, str);

*let* [json, arrN, num, str] = [{ f: 45, g: 34 }, [2, 3, 4], 22, 'sssss'];

*console*.log(json, arrN, num, str);

五、数组

map----------映射

reduce--------汇总

filter----------过滤器

forEach--------循环（迭代）

map：

*let* arr = [1, 2, 3];

*let* result = arr.map(*element* *=>* element \* 2);

*console*.log(result);

*let* score = [19, 28, 87, 65, 70];

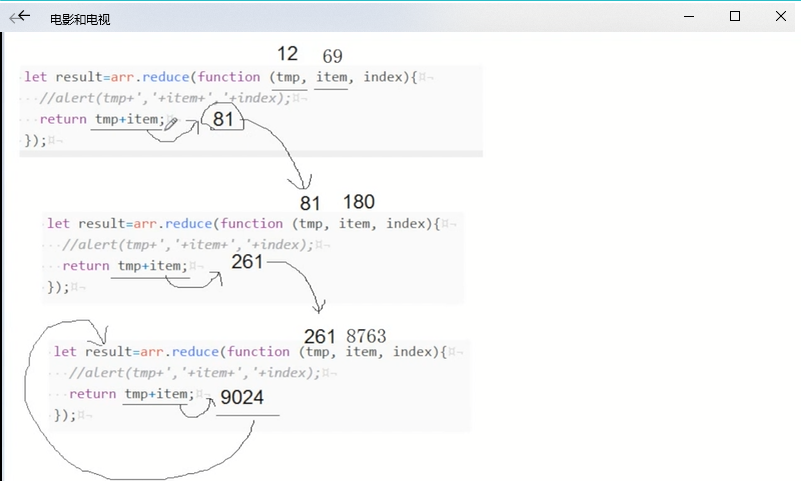
*let* res = score.map(*item* *=>* item >= 60 ? '及格' : '不及格' );

*console*.log(res);

reduce：汇总（一堆出来一个）

求和：[12,8000000,5999999] --🡪 80060011

求品均值：[1,2,3] ---🡪 2



*let* arr = [1, 2, 3];

*let* result = arr.reduce(*function* (*tem*, *item*, *index*) {

return tem + item;

});

*console*.log("和:" + result);

*let* ave = arr.reduce((*tem*, *item*, *index*) *=>* {

if (index != arr.length - 1) {

return tem + item;

} else {

return (tem + item) / arr.length;

}

});

*console*.log("平均值:" + ave);

filter 过滤器

（例子：过滤掉不能被三整除的项）

*let* arr = [12, 5, 8, 34, 55, 23];

*let* result = arr.filer(*item* *=>* {

// if (item % 3 == 0) {

// return true;

// } else {

// return false;

// }

return item % 3 == 0;

});

*console*.log(result);