对象拓展

1. Object.keys、values、entries
   1. Object.keys
   2. values
      1. 成员顺序
      2. 可遍历属性
      3. 参数是字符串
   3. entries
      1. 与Object.values基本一致
2. 对象方法简写，计算属性
3. 展开运算符

【Object.keys、values、entries】

1. Object.keys：将对象成员（不包含继承）返回成一个数组，返回键名

var obj = { foo: 'bar', baz: 42 };

Object.keys(obj)

// ["foo", "baz"]

1. values：将对象成员（不包含继承）返回成一个数组，返回键值

const obj = { foo: 'bar', baz: 42 };

Object.values(obj)

// ["bar", 42]

* 1. 成员顺序：属性名为数值的属性，是按照数值大小，从小到大遍历

const obj = { 100: 'a', 2: 'b', 7: 'c' };

Object.values(obj)

// ["b", "c", "a"]

* 1. 只返回对象自身的可遍历属性。

const obj = Object.create({}, {p: {value: 42}});

Object.values(obj) // []

上面代码中，Object.create方法的第二个参数添加的对象属性（属性p），如果不显式声明，默认是不可遍历的，因为p的属性描述对象的enumerable默认是false，Object.values不会返回这个属性。只要把enumerable改成true，Object.values就会返回属性p的值。

const obj = Object.create({}, {p:

{

value: 42,

enumerable: true

}});

Object.values(obj) // [42]

Object.values会过滤属性名为 Symbol 值的属性。

Object.values({ [Symbol()]: 123, foo: 'abc' });

// ['abc']

* 1. 参数是一个字符串，会返回各个字符组成的一个数组

Object.values('foo')

// ['f', 'o', 'o']

1. Object. entries：将对象成员（不包含继承）返回成一个数组，返回键值对数组

const obj = { foo: 'bar', baz: 42 };

Object.entries(obj)

// [ ["foo", "bar"], ["baz", 42] ]

除了返回值不一样，该方法的行为与Object.values基本一致。

【对象方法简写、计算属性】

老：

const key = ‘job’

const obj = {

num:1,

str:’darbuly’,

work:function(){

},

//key:’fe’

}

obj[key]=’fe’

简写：

1. 函数去掉function和冒号

work(){

}

1. key使用中括号，还可以加

[key+’work’]:’fe’,

3.直接写变量进去做属性，