10、Set和Map数据结构

- es6 提供了两种新的数据结构 Set 和 Map

Set:

- Set 是一个构造函数，用来生成 Set 数据结构，它类似于数组，但是成员的值都是唯一的，没有重复的值.
- 初始化 Set 可以接受一个数组或类数组对象作为参数，也可以创建一个空的 Set

```
var s1 = new Set();
var s2 = new Set([1, 2, 3]);
console.log(s1);// Set(0) {}
console.log(s2);// Set(3) {1, 2, 3}
```

- 在 Set 中成员的值是唯一的，重复的值自动被过滤掉

```
var s1 = new Set([1, 2, 2, 3, 1, 4]);
console.log(s1);// Set(4) {1, 2, 3, 4}
```

- Set 的一些属性方法：
  - size：返回成员总数
  - add(value)：添加某个值，返回Set结构本身
  - delete(value)：删除某个值，返回一个布尔值，表示删除是否成功
  - has(value)：返回一个布尔值，表示该值是否为Set的成员
  - clear()：清除所有成员，没有返回值

```
var set = new Set([1,2]);
set.add(3);// 添加成员
console.log(set.size);// 3 成员总数
console.log(set);// Set(3) {1, 2, 3}
set.add([4,5]);// 添加成员
console.log(set.size);// 4 成员总数
console.log(set.has(2));// true 有该成员
console.log(set);// Set(4) {1, 2, 3, [4, 5]}
set.delete(2);// 删除成员
console.log(set);// Set(3) {1, 3, [4, 5]}
console.log(set.has(2));// false 没有该成员
set.clear();// 清除所有成员
console.log(set);// Set(0) {}
```

- 得益于数据结构 Set 查找更快速高效，但也因为数据结构的内部数据是无序的，无法实现按下标改查，排序等操作

```
var arr = [1,2,3,'a',4,'b'];
var set = new Set(arr);
console.log(set[0]);// undefined
console.log(set['a']);// undefined
```

Map:

- Map 是一个构造函数，用来生成 Map 数据结构，它类似于对象，也是键值对的集合.
- 但是“键”可以是非字符串, 初始化 Map 需要一个二维数组，或者直接初始化一个空的 Map

```
var m1 = new Map();
var m2 = new Map([['a', 123], ['b', 456], [3, 'abc']]);
console.log(m1);// Map(0) {}
console.log(m2);// Map(3) {"a" => 123, "b" => 456, 3 => "abc"}
```

- Map 的一些操作方法：
  - set(key, value)：设置键值对
  - get(key)：获取键对应的值
  - has(key)：是否存在某个键
  - delete(key)：删除某个键值对，返回一个布尔值，表示删除是否成功

```
var map = new Map([['a', 123], ['b', 456], [3, 'abc']]);
map.set('c',789);
console.log(map.get('c')); // 789
console.log(map.has('b')); // true 此key存在
map.delete(3); // true 成功删除key
console.log(map); // Map(3) {"a" => 123, "b" => 456, "c" => 789}
```

- 传统 Object 只能用字符串作键，Object 结构提供了“字符串 — 值”的对应
- Map 结构提供了“键 — 值”的对应，是一种更完善的 Hash 结构实现
- 如果你需要“键值对”的数据结构，Map 比 Object 更快速 更高效 更合适
- 遍历Map和Set对象

```
//遍历Map
var map = new Map([['a', 123], ['b', 456], [3, 'abc'],['c', 789]]);
map.forEach((val,key,obj)=>{
    console.log('val: '+val);
    console.log('key: '+key);
    console.log(obj);
}); // 结果如下图
```

```
//遍历Set
var set = new Set(['a','b','c','d']);
set.forEach((val,key,obj)=>{
    console.log('val: '+val);
    console.log('key: '+key);
    console.log(obj);
});// 结果如下图
```

- 当然，还可以使用es6中的 for of 遍历

```
var map = new Map([['a', 123], ['b', 456], [3, 'abc'],['c', 789]]);
for (const [key,val] of map) {
    console.log(key+' : '+val);
}
// a : 123
// b : 456
// 3 : abc
// c : 789
```

```
var set = new Set(['a','b','c','d']);
for (const val of set) {
    console.log(val);// a b c d
}
```