Javascript数据结构

集合

Skipper

红点工场
redpoint.live

什么是集合?

集合是一种数学中的概念

$$A = \{ 1, 2, 3\}$$

$$B = \{ 7, 2, 9\}$$

集合的特性&概念

1、无重复性　　A = { 1, 2, 2, 3}

A = { 1, 2, 3}

2、空集　　$A = \{\}$

3、子集　　$A = \{1, 2, 3\}$

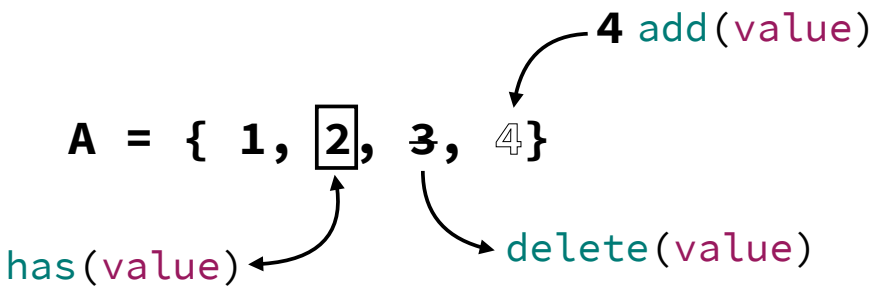
$B = \{1, 2\}$

B是A的子集：$B \subseteq A$

构建集合框架

```
var Set = function(){
    var items = {};
}
```

集合操作方法



集合中查询元素存在：has(value)

```
this.has = function(value){
    return items.hasOwnProperty(value);
}
```

集合中添加元素：add(value)

```
this.add = function(value){
    if(!this.has(value)){
        items[value] = value;
        return value;
    }
    return false;
}
```

集合中移除元素：remove(value)

```
this.add = function(value){
    if(this.has(value)){
        delete items[value];
        return true;
    }
    return false;
}
```

集合其他操作

1、清除集合：

```
this.clear = function(){
    items = {};
}
```

2、获取集合大小：

方法一、

```
this.size = function(){
    return Object.keys(items).length;
}
```

方法一、

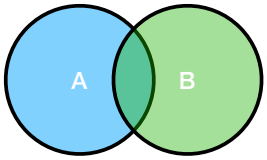
```
this.sizeLegacy = function(){
    var count = 0;
    for(var key in items){
        if(items.hasOwnProperty(key)){
            count++;
        }
    }
    return count;
}
```

3、提取集合全部值并以数组返回：

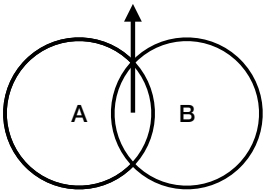
```
this.values = function(){
    var values = [];
    for(var key in items){
        if(items.hasOwnProperty(key)){
            values.push(key);
        }
    }
    return values;
}
```

集合间操作

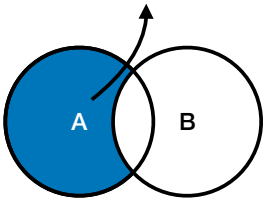
并集：union



交集：intersection

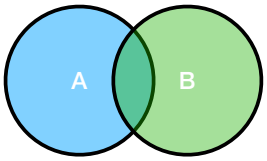


差集：difference



并集操作

并集：union




```
A = {1 ,2};
B = {2 ,3};

A.union(B)={1,2,3};
```

并集操作代码

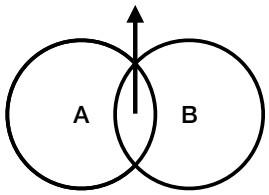
```
this.union = function(otherSet){
    var unionSet = new Set();

    var values = this.values();
    for(let i = 0; i < values.length ; i++){
        unionSet.add(values[i]);
    }

    values = otherSet.values();
    for(let i = 0; i < values.length ; i++){
        unionSet.add(values[i]);
    }
    return unionSet;
}
```

交集操作

交集：intersection




```
A = {1 ,2};
B = {2 ,3};

A.intersection(B)={2};
```

交集操作代码

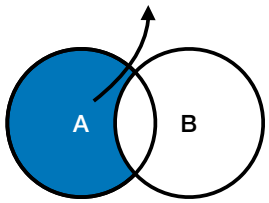
```
this.intersection = function(otherSet){
    var intersectionSet = new Set();

    var values = this.values();
    for(let i = 0; i < values.length ; i++){
        if(otherSet.has(values[i])){
            intersectionSet.add(values[i]);
        }
    }

    return intersectionSet;
}
```

差集操作实现

差集：difference




```
A = {1 ,2};
B = {2 ,3};

A.difference(B)={1};
```

差集操作代码

```
this.intersection = function(otherSet){
    var differenceSet = new Set();

    var values = this.values();
    for(let i = 0; i < values.length ; i++){
        if(!otherSet.has(values[i])){
            differenceSet.add(values[i]);
        }
    }

    return differenceSet;
}
```