Array.prototype.push()

push() 方法将一个或多个元素添加到数组的末尾，并返回数组的新长度。

```
var numbers = [1, 2, 3];
numbers.push(4);

console.log(numbers);
// [1, 2, 3, 4]

numbers.push(5, 6, 7);

console.log(numbers);
// [1, 2, 3, 4, 5, 6, 7]
```

语法

```
arr.push(element1, ..., elementN)
```

参数

elementN
被添加到数组末尾的元素。

返回值

当调用该方法时，新的 `length` 属性值将被返回。

描述

push方法将值追加到数组中。

`push` 方法有意具有通用性。该方法和 `call()` 或 `apply()` 一起使用时，可应用在类似数组的对象上。 `push` 方法根据 `length` 属性来决定从哪里开始插入给定的值。如果 `length` 不能被转成一个数值，则插入的元素索引为 0，包括 `length` 不存在时。当 `length` 不存在时，将会创建它。

唯一的原生类数组（array-like）对象是 `Strings` ，尽管如此，它们并不适用该方法，因为字符串是不可改变的。

示例

添加元素到数组

下面的代码创建了 `sports` 数组，包含两个元素，然后又把两个元素添加给它。 `total` 变量为数组的新长度值。

```
var sports = ["soccer", "baseball"];
var total = sports.push("football", "swimming");

console.log(sports);
// ["soccer", "baseball", "football", "swimming"]

console.log(total);
// 4
```

合并两个数组

该示例使用 `apply()` 添加第二个数组的所有元素。

注意当第二个数组(如示例中的moreVegs)太大时不要使用这个方法来合并数组，因为事实上一个函数能够接受的参数个数是有限制的。具体可以参考 `apply()` 。

```
var vegetables = ['parsnip', 'potato'];
var moreVegs = ['celery', 'beetroot'];

// 将第二个数组融合进第一个数组
// 相当于 vegetables.push('celery', 'beetroot');
Array.prototype.push.apply(vegetables, moreVegs);

console.log(vegetables);
// ['parsnip', 'potato', 'celery', 'beetroot']
```

像数组一样使用对象

如上所述，push 是特意设计为通用的，我们可以使用它来获得便利。正如下面的例子所示，Array.prototype.push 可以在一个对象上工作。 注意，我们没有创建一个数组来存储对象的集合。 相反，我们将该集合存储在对象本身上，并使用在 Array.prototype.push 上使用的 `call` 来调

用该方法，使其认为我们正在处理数组，而它只是像平常一样运作，这要感谢 JavaScript 允许我们建立任意的执行上下文。

```
var obj = {
    length: 0,

    addElem: function addElem (elem) {
        // obj.length is automatically incremented
        // every time an element is added.
        [].push.call(this, elem);
    }
};

// Let's add some empty objects just to illustrate.
obj.addElem({});
obj.addElem({});
console.log(obj.length);
// → 2
```

注意，尽管 obj 不是数组，但是 push 方法成功地使 obj 的 length 属性增长了，就像我们处理一个实际的数组一样。

规范

Specification	Status	Comment
ECMAScript 3rd Edition (ECMA-262)	ST Standard	Initial definition. Implemented in JavaScript 1.2.
ECMAScript 5.1 (ECMA-262) Array.prototype.push	ST Standard	
ECMAScript 2015 (6th Edition, ECMA-262) Array.prototype.push	ST Standard	
ECMAScript Latest Draft (ECMA-262) Array.prototype.push	LS Living Standard	

浏览器兼容性

Desktop Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari
Basic support	1.0	1.0 (1.7 or earlier)	5.5	(Yes)	(Yes)

相关链接

- `Array.prototype.pop()`
- `Array.prototype.shift()`
- `Array.prototype.unshift()`
- `Array.prototype.concat()`