函数

函数的三种定义方式

1、

```
字面量
    function声明
        function add() {
            // body...
        }
        add();
    var赋值表达式
        var add = function (argument) {
            // body...
        };
        add();
        var add = function fn(argument) {
            // body...
        };
```

```
var add = function fn(argument) {
        // body...
        fn();
};
add();
fn();X
```

这里的fn只能在函数体内部调用

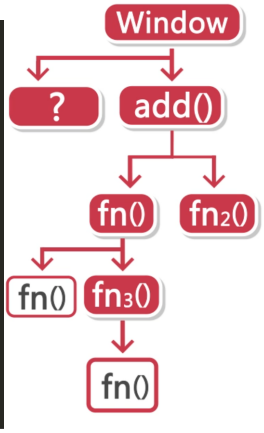
2

```
// 构造函数
    var add = new Function('num1', 'num2', 'return num1 + num2;')
```

赋值语句要加分号

3、关于函数的位置（作用域链）

```
fn()X
function add() {
        fn();
        function fn() {
                fn()
                function fn3(argument) {
                        fn()
                }
                // body...
        }
        function fn2() {
                fn();
                // body...
        }
}
```



4、匿名函数的执行：

```
(function () {
        console.log(1);
})();

(function () {
        console.log(1);
}());
```

```
!+-~function () {
        console.log(1);
}();
```

5、方法的调用，关于属性加不加引号的问题看他是否合法

```
var operation = {
        add: function (num1, num2) {
                return num1 + num2;
        },
        subtract: function (num1, num2) {
                return num1 - num2;
        },
        '@': function () {
                console.log('@');
        }
};
console.log(operation.add(1, 2));
console.log(operation['@'](1, 2));
```

6、arguments的应用，可以在函数的实际参数不确定的时候进行实参的获取

```
function add() {
        if (arguments.length == 0) return;
        var sum = 0;
        for (var i = 0; i < arguments.length; i++) {
                sum += arguments[i];
        }
        return sum;
}
console.log(add());
console.log(add(1, 2, 3, 4, 5));
```

arguments是类数组，不能直接调用数组的方法，是每一个函数都有的。

arguments.callee最常用的地方在于递归

```
function jiecheng(num) {
        if (num <= 1) return 1;
        return num * arguments.callee(num - 1);
}
console.log(jiecheng(5));
console.log(jiecheng(4));
```

arguments.callee就是指代函数本身，如果改变了函数名的话就可以直接用callee进行调用

在严格模式下arguments.callee不能使用

```
var jicheng = function fn(num) {
        if (num <= 1) return 1;
        return num * fn(num - 1);
};
console.log(jicheng(5));
console.log(jicheng(4));
```

所以在严格模式下就可以使用上面这种方法调用

```
function add(num1, num2) {
        if (arguments.length != add.length) throw new Error('请传入' + add.length + '个参数！'
                );
        return num1 + num2;
}
console.log(add(1, 1));
console.log(add(1));
console.log(add(1, 2, 3));
```

每个函数都有length的方法，用于检测形式参数的个数