JS函数（function）的定义和使用

函数是一组执行特定任务（具有特定功能）的，可以重复使用的代码块，前面几节中用到的 alert()、write() 就是 JavaScript 中内置的函数。

除了使用内置函数外，我们也可以自行创建函数（自定义函数），然后在需要的地方调用这个函数，这样不仅可以避免编写重复的代码，还有利于代码的后期维护。本节我们主要来介绍一下如何使用 JavaScript 编写一个自己的函数。

JS 定义函数

JS 函数声明需要以 function 关键字开头，之后为要创建的函数名称，function 关键字与函数名称之间使用空格分开，函数名之后为一个括号 `()`，括号中用来定义函数中要使用的参数（多个参数之间使用逗号 `,` 分隔开），一个函数最多可以有 255 个参数，最后为一个花括号 `{}`，花括号中用来定义函数的函数体（即实现函数的代码），如下所示：

```
function functionName(parameter_list) {
   // 函数中的代码
}
```

示例代码如下：

```
01.  function sayHello(name){
02.      document.write("Hello " + name);
03.  }
```

上面示例中定义了一个函数 sayHello()，该函数需要接收一个参数 name，调用该函数会在页面中输出“Hello ...”。

JS 调用函数

一旦定义好了一个函数，我们就可以在当前文档的任意位置来调用它。调用函数非常简单，只需要函数名后面加上一个括号即可，例如 alert()、write()。注意，如果在定义函数时函数名后面的括号中指定了参数，那么在调用函数时也需要在括号中提供对应的参数。

示例代码如下：

```
01.  function sayHello(name){
02.      document.write("Hello " + name);
03.  }
04.  // 调用 sayHello() 函数
05.  sayHello('C语言中文网');
```

> 提示：JavaScript 对于大小写敏感，所以在定义函数时 function 关键字一定要使用小写，而且调用函数时必须使用与声明时相同的大小写来调用函数。

参数的默认值

在定义函数时，您可以为函数的参数设置一个默认值，这样当我们在调用这个函数时，如果没有提供参数，就会使用这个默认值作为参数值，如下例所示：

```
01.  function sayHello(name = "World"){
02.      document.write("Hello " + name);
03.  }
04.
05.  sayHello();                  // 输出：Hello World
06.  sayHello('c.biancheng.net');      // 输出：Hello c.biancheng.net
```

JS 函数返回值

在函数中可以使用 return 语句将一个值（函数的运行结果）返回给调用函数的程序，这个值可以是任何类型，例如数组、对象、字符串等。对于有返回值的函数，我们可以会使用一个变量来接收这个函数的返回值，示例代码如下：

```
01.  function getSum(num1, num2){
02.      return num1 + num2;
03.  }
04.
05.  var sum1 = getSum(7, 12);       // 函数返回值为：19
06.  var sum2 = getSum(-5, 33);      // 函数返回值为：28
```

> 提示：return 语句通常在函数的末尾定义，当函数运行到 return 语句时会立即停止运行，并返回到调用函数的地方继续执行。

另外，一个函数只能有一个返回值，若要返回多个值则，则可以将值放入一个数组中，然后返回这个数组即可，如下例所示：

```
01.  function division(dividend, divisor){
02.      var quotient = dividend / divisor;
03.      var arr = [dividend, divisor, quotient]
04.      return arr;
05.  }
06.
07.  var res = division(100, 4)
08.  document.write(res[0]);          // 输出：100
09.  document.write(res[1]);          // 输出：4
10.  document.write(res[2]);          // 输出：25
```

JS 函数表达式

函数表达式与声明变量非常相似，是另外一种声明函数的形式，语法格式如下：

```
var myfunction = function name(parameter_list){
   // 函数中的代码
};
```

参数说明如下：

- myfunction：变量名，可以通过它来调用等号之后的函数；
- name：函数名，可以省略（一般情况下我们也会将其省略），如果省略那么该函数就会成为一个匿名函数；
- parameter_list：为参数列表，一个函数最多可以有 255 个参数。

示例代码如下：

```
01.  // 函数声明
02.  function getSum(num1, num2) {
03.      var total = num1 + num2;
04.      return total;
05.  }
06.
07.  // 函数表达式
08.  var getSum = function(num1, num2) {
09.      var total = num1 + num2;
10.      return total;
11.  };
```

上面示例中的两个函数是等价的，它们的功能、返回值、调用方法都是相同的。

> 注意：在函数声明中，不需要在右花括号后放置分号，但若使用函数表达式就应该在表达式的最后以分号结尾。

函数声明和函数表达式虽然看起来非常相似，但它们的运行方式是不同的，如下例所示：

```
01.  declaration();          // 输出: function declaration
02.  function declaration() {
03.      document.write("function declaration");
04.  }
05.
06.  expression();           // 报错: Uncaught TypeError: undefined is not a
     function
07.  var expression = function() {
08.      document.write("function expression");
09.  };
```

如上例所示，如果函数表达式在定义之前被调用，会抛出异常（报错），但函数声明则可以成功运行。这是因为在程序执行前，JavaScript 会先对函数声明进行解析，因此无论是在函数声明前还是声明后调用函数都是可行的。而函数表达式则是将一个匿名函数赋值给一个变量，所以在程序还没有执行到该表达式之前，相当于函数还未定义，因此无法调用。