14作用域

通常来说，一段程序代码中所用到的名字并不总是有效和可用的，而限定这个名字的**可用性的代码范围**就是这个名字的**作用域**。作用域的使用提高了程序逻辑的局部性，增强了程序的可靠性，减少了名字冲突。

1、JavaScript作用域：代码名字（变量）在某个范围内起作用和效果，目的是为了提高程序的可靠性，更重要的是减少命名冲突。

2、js作用域（es6）之前：全局作用域　局部作用域

3、全局作用域：整个`script`标签或者是一个单独js文件

4、局部作用域：作用于函数内的代码环境，就是局部作用域。 因为跟函数有关系，所以也称为函数作用域

```
    <script>
        var num = 10;
        var num = 11;   //全局作用域
        function fn() {
            var num = 20;   //局部作用域（函数作用域），只在函数内部起效果。
            console.log(num);
        }
        fn();
        console.log(num);
    </script>
```

全局变量

在全局作用域下声明的变量叫做`全局变量`（**在函数外部定义的变量**）

全局作用域：整个`script`标签或者是一个单独js文件

1. 全局变量在代码的任何位置都可以使用

    ```
        <script>
            var num = 1;
            function fn() {
                console.log(num);
            }
            fn();
        </script>
    ```

2. 在全局作用域下 `var` 声明的变量 是全局变量
3. 特殊情况下，在函数内不使用 var 声明的变量也是全局变量`（不建议使用）`

```
   <script>
        function fn() {
            var num = 1;
            num2 = 2;
        }
        fn();
        // console.log(num);    //报错,已声明的局部变量不可以在外部调用。
        console.log(num2);  //2;如果在函数内部，没有声明，直接赋值的变量也属于全局变量。
    </script>
```

局部变量

在局部作用域下声明的变量叫做局部变量（**在函数内部定义的变量**）

局部变量作用域：作用于函数内的代码环境，就是局部作用域。 因为跟函数有关系，所以也称为函数作用域

1. 局部变量只能在该函数内部使用
2. 在函数内部 var 声明的变量是局部变量

```
    <script>
        function fn() {
            var num = 1;
            num2 = 2;
        }
        fn();
        // console.log(num);    //报错,已声明的局部变量不可以在外部调用。
        console.log(num2);  //2;如果在函数内部，没有声明，直接赋值的变量也属于全局变量。
    </script>
```

3. 函数的**形参**实际上就是**局部变量**

全局变量和局部变量的区别

- 全局变量：在任何一个地方都可以使用，只有在浏览器关闭时才会被销毁，因此比较占内存
- 局部变量：只在函数内部使用，当其所在的代码块被执行时，会被初始化；当代码块运行结束后，就会被销毁，因此更节省内存空间

作用域链

1. 只要是代码，就至少有一个作用域
2. 写在函数内部的叫局部作用域
3. 如果函数中还有函数，那么在这个作用域中就又可以诞生一个作用域
4. 根据在`内部函数可以访问外部函数变量`的这种机制，用链式查找决定哪些数据能被内部函数访问，就称作`作用域链`

5. 作用域链：采取**就近原则**的方式来查找变量最终的值。

```
    <script>
        // 作用域链：内部函数访问外部函数的变量，采取的是链式查找的方式来决定取哪个值，这种结构我们称为作用域链表
        var num = 10;
        function fn() { //外部函数
            var num = 20;

            function fun() { //内部函数
                console.log(num);  // 20 ,一级一级访问
            }
            fun();
        }
        fn();
    </script>
```

案例1：作用域链案例，以下代码的结果是多少？

```
    <script>
        // 案例：根据作用域链，判断以下代码的结果是几？
        function fn1() {
            var num = 123;
            function fn2() {
                var num = 0;
                console.log(num);   //0
            }
            fn2();
        }
        var num = 4567;
        fn1();
    </script>
```

案例2：作用域链案例，以下代码的结果是多少？

```
    <script>
        var a = 1;
        function fn1() {
            var a = 2;
            var b = '22';
            fn2();
            function fn2() {
                var a = 3;
                fn3();
                function fn3() {
                    var a = 4;
                    console.log(a);     //求a的值？
                    console.log(b);     //求b的值？
                }
            }
        }
        fn1();
    </script>
```