JavaScript进阶

---JS语法、表达式及语句

河北师范大学软件学院
Software College of Hebei Normal University

内容提纲

- JS语法、表达式及语句综述
- JS严格模式
- switch详解、for...in

JS语法、表达式及语句综述

- 字面量
  - 对象字面量 var obj = {x:12，y:23};
  - 数组字面量 var arr = [1,2,true,'xyz'];
- 标识符与保留字
  - 标识符用来给变量或函数进行命名，以字母、下划线或$为开始
  - 保留字：arguments、break、case、catch、class、const等（参见教材7.6）
- 表达式（expression）与语句（statement）
  - 表达式代码中基本的单位，它将产生一个值，用于需要值的地方，如：if(a>b){...}
  - 语句表示了一种行为，如：var obj = {x:1,y:b}; //创建对象obj
  - JS期望语句的地方都可以写表达式（表达式语句），如：x+3;

参见实例demo01 语法二义性

JS语法、表达式及语句综述

- 表达式及表达式分类
  - 原始表达式、对象及数组初始化表达式、函数定义表达式、属性访问表达式
  - 函数调用表达式、对象创建表达式、算数表达式、关系表达式、逻辑表达式、赋值表达式
- 语句及语句分类

参见实例demo02 Part1和Part2

  - 表达式语句、复合语句、条件语句(if-else、switch)、循环语句（for、for...in）
- ES5中的块（ES5中没有块作用域，所以带来了很多问题）

```
{
    var a = 20;
}
console.log(a);
```

```
if(true){
    var a = 20;
}
console.log(a);
```

```
if(false){
    var b = 30;
}
console.log(b);
```


参见实例demo03 ES5中关于块的实例

内容提纲

- JS语法、表达式及语句综述
- JS严格模式
- switch详解、for...in

JS严格模式

- ES5中的运行模式
  - 严格模式和非严格模式（松散模式）
- 严格模式的目的
  - 消除Javascript语法的一些不合理、不严谨之处，减少一些怪异行为
  - 消除代码运行的一些不安全之处，保证代码运行的安全
  - 提高编译器效率，增加运行速度
- 启用严格模式的方式
  - 针对整个脚本文件使用 'use strict'
  - 针对函数使用 'use strict'

参见实例demo04 使用严格模式的不同方式

JS严格模式下语法和行为的改变 一（全局变量）

- 严格模式下全局变量需显式声明

```
function  sloppyFunc() {
    sloppyVar = 123;
}
sloppyFunc();
console.log(sloppyVar);
```

```
123
```

```
function  sloppyFunc() {
    'use strict'
    sloppyVar = 123;
}
sloppyFunc();
console.log(sloppyVar);
```

```
▶Uncaught ReferenceError:
    at sloppyFunc (<anonym
    at <anonymous>:5:1
```

参见实例demo05

JS严格模式下语法和行为的改变 二（函数中的this）

- 一般函数中的this（严格模式）为undefined，非严格下为全局变量

```
> function thisTest(){
      'use strict'
      console.log(this);
  }
  thisTest();
undefined
```

- 可以用此特性来判断当前是否为严格模式

```
> //'use strict'
  function isStrictMode(){
      return this===undefined?true:false;
  }
  isStrictMode();
< false
```


参见实例demo06 严格模式下函数中的this

JS严格模式下语法和行为改变 三（属性、变量及函数参数）

- 严格模式下禁止删除不可改变的属性和未定义的变量

```
var str = "abc";
function  sloppyFunc() {
    str.length = 7;
    console.log(str.length);
}
sloppyFunc();
```

```
3
```

```
var str = "abc";
function  strictFunc() {
    'use strict'
    str.length = 7;
    console.log(str.length);
}
strictFunc();
```

```
▶Uncaught TypeError: Cannot assign
```

- 严格模式下禁止函数参数重名

```
function f(a, a, b) {
    return a+b;
}
f(2,3,4);
```

```
7
```

```
"use strict";
function f(a, a, b) {
    return a+b;
}
f(2,3,4);
```

```
Uncaught SyntaxError: Duplicate parameter
```


参见实例demo07 查看更多严格模式案例

内容提纲

- JS语法、表达式及语句综述
- JS严格模式
- switch详解、for...in

switch详解

- switch语句中的case
  - case在比较时使用的是全等操作符比较,因此不会发生隐式类型转换
  - case 后可以是一个表达式（如 i<60）

```
var i = "1";
switch(i){
    case 1:
        console.log("case 1 Number");
        break;
    default:
        console.log("输出: default");
}
```

```
输出: default
```

```
var i = 65;
switch(true){
    case i>=60:
        alert('及格');
        break;
    case i<60:
        alert('不及格');
        break;
    default:
        alert('default');
}
```


参见实例demo08 switch与数据类型案例 回顾数据类型

switch详解

- switch语句中的穿透性及其应用

```
var i = 1;//i=2、3、4
switch(i){
    case 1:
        console.log("case 1");
    case 2:
        console.log("case 2");
        break;
    case 3:
        console.log("case 3");
        //break;
    case 4:
        console.log("case 4");
}
```

```
case 1
```

```
case 2
```

从满足第一case处开始执行，直到遇到break为止，若都没有break则直到default结束为止

利用switch的穿透性:求某月某日是一年中的第几天

参见实例demo09 switch贯穿案例

for...in

- for...in常用来遍历对象

```
var obj = {x:10,y:20,z:"30"};
for(var k in obj){
    console.log(k,obj[k],typeof obj[k]);
}
```

```
x 10 number
y 20 number
z 30 string
```

- for...in遍历数组（忽略空缺）

```
var arr = [2,,"33"];
for(var i in arr){
    console.log(i,arr[i]);
}
```

```
0 2
2 33
```


参见实例demo10 for...in案例

Thank You!

河北師范大学软件学院
Software College of Hebei Normal University