JavaScript进阶

---JS对象综述

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

内容提纲

- JS对象简介
- JS对象的属性
- JS对象相关操作

JS对象简介

- JS对象是什么
  - JS对象是一种复合值：将很多值复合在一起（包括原始类型值、对象、函数）
  - JS对象是若干无序属性的集合，可以直接通过属性名来访问对象的属性（键值对）
  - 函数作为某一个对象的属性时，称其为该对象的方法

```
var obj = {
    num:10,
    str:"Hi",
    show:function(){
        console.log(this.str);
    }
};
```

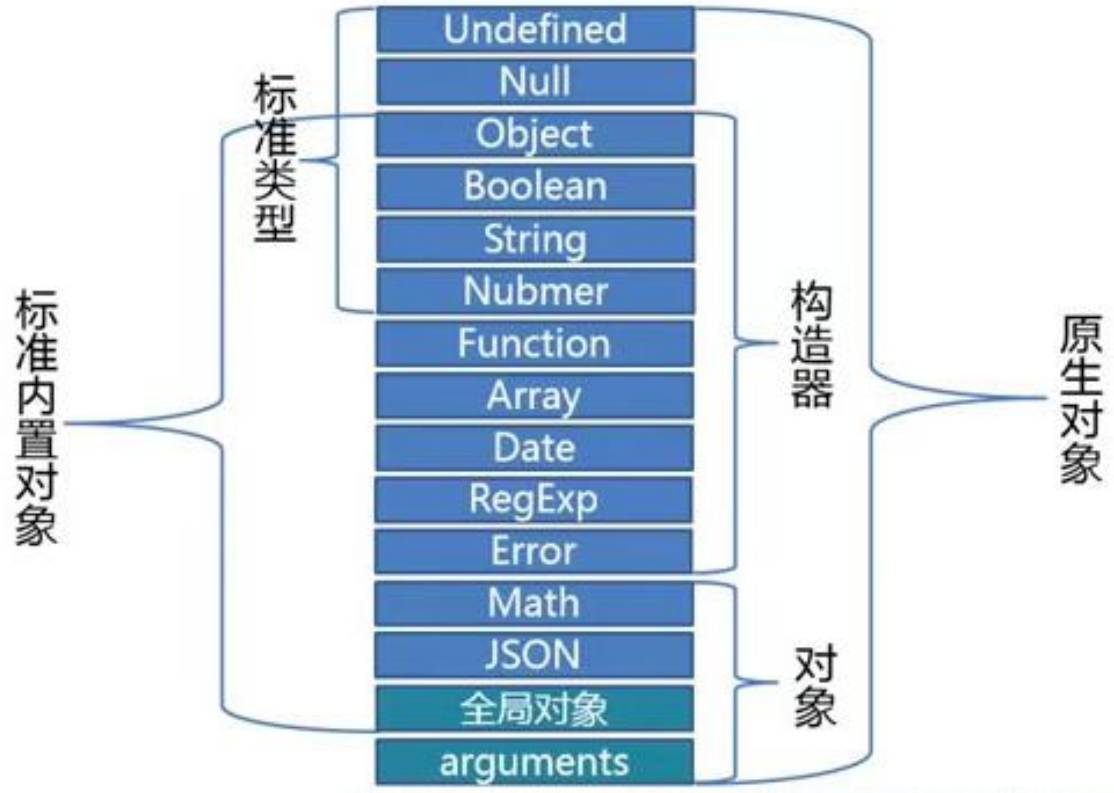
```
console.log(obj.num);//10
console.log(obj.str);//Hi
obj.show();          //Hi
```

```
10
Hi
Hi
```


参见实例demo01 创建JS对象练习

JS对象简介

- JS对象分类
  - 内置对象（native object）由ECMAScript规范定义的对象或构造器对象（数组、函数等）
  - 宿主对象（host object）由JS解析器所嵌入的宿主环境定义的（如：window、document）
  - 自定义对象（user-defined object）运行中的用户自定义JS代码创建的对象



标准内置对象分为两类：
1，构造器函数对象（类对象）
2，非构造器对象

思考 ：typeof Array
：typeof Function
：typeof Date
：typeof Math
：typeof JSON

参见实例demo02 Part1

JS对象简介

- 思考：下述代码都输出什么，并解释原因

```
console.log(Object instanceof Function);
console.log(Object instanceof Object);
console.log(Number instanceof Function);
console.log(Number instanceof Object);
console.log(Function instanceof Function);
console.log(Function instanceof Object);
console.log(Array instanceof Function);
console.log(Array instanceof Object);
console.log(Math instanceof Function);
console.log(Math instanceof Object);
console.log(JSON instanceof Function);
console.log(JSON instanceof Object);
```


参见实例demo02 Part2

内容提纲

- JS对象简介
- JS对象的属性
- JS对象相关操作

JS对象的属性

- JS对象属性的分类
  - 数据属性（property，属性），字符串的键到值的映射（包括基本类型数据、对象、函数）
  - 访问器属性（accessor，或称为访问器），访问属性的方法，注意：访问和设置时不加括号
  - 内置属性（internal property）存在与ECMAScript规范中，不能直接访问

```
var o = {
    _x:1.0,
    get x(){
        return this._x;
    },
    set x(val){
        this._x = val;
    }
};
console.log(o.x);//1
o.x = 2;
console.log(o.x,o._x);//2 2
```

如果只有getter方法那么是只读属性，如果只有setter方法，则是一个只写属性，读取时返回undefined

访问器属性和数据属性不同，存取器不具有可写的属性特性（writable attribute）具体参见属性特性部分

参见实例demo03 访问器属性

JS对象的属性

- JS对象访问器属性实例

```
var p1 = {
    _name:"Jack",
    _age:23,
    set age(val){
        if(val>0&&val<150){
            this._age = val;
        }else{
            console.log("请设置正常年龄");
        }
    },
    get age(){
        return this._age;
    }
};
p1.age = 178;
```

```
请设置正常年龄
```

访问器属性不包含数据值，它包含一对getter和setter函数

实现数据属性的间接访问，可实现数据的验证、过滤、运算等功能

参见实例demo03 访问器属性

内容提纲

- JS对象简介
- JS对象的属性
- JS对象相关操作

JS对象相关操作

- 创建JS对象的方式
  - 通过对象字面量的方式直接创建对象
  - 通过Object的create静态方法创建对象
  - 通过构造函数的方式创建对象

```
var obj = {
    num:10,
    str:"Hi",
    show:function(){
        console.log(this.str);
    }
}

var subObj = Object.create(obj);
subObj.age = 23;
```

```
function Person(age,name){
    this.age = age;
    this.name = name;
}
Person.prototype.sayHi = function(){
    console.log("Hi,I'm "+this.name);
}
var p1 = new Person(20,"Jame");
p1.sayHi();
```

不同方式创建的对象的原型都是什么？

参见实例demo04 生成对象及对象原型链

JS对象相关操作

- 对象属性的增删改查
  - 添加和删除自有属性
  - 访问和修改自有属性
  - 通过点与中括号访问属性的区别（写个访问属性的for循环练习）

```
var obj = {};
obj.x = 2;//直接添加属性
console.log(obj.x);//通过.访问属性
obj.x = 5;//设置属性
console.log(obj["x"]);//通过[]访问属性
delete obj.x;//删除属性
console.log(obj.x);
```


参见实例demo05 对象属性相关操作

Thank You!