

JavaScript进阶

---JS对象综述

---JS对象属性特性



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

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

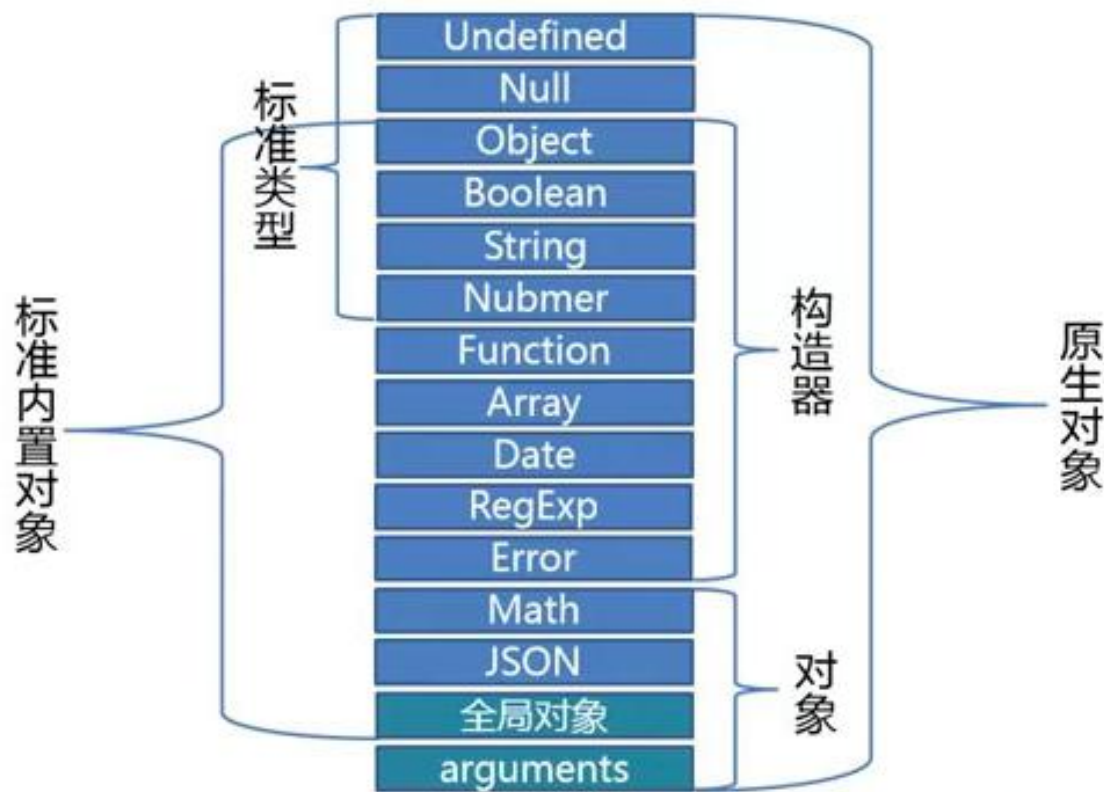
```
var obj = {  
  num:10,  
  str:"Hi",  
  show:function(){  
    console.log(this.str);  
  }  
};
```

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

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);
```


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

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);
```

The background of the slide is decorated with various abstract shapes in shades of green and yellow. These shapes, which include circles, ovals, and irregular blobs, are scattered across the top and right sides of the slide, creating a modern, organic feel.

Have a Break!

JavaScript进阶

---JS对象属性特性



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

内容提纲

- **对象属性特性简介**
- **对象属性（数据属性）的特性**
- **对象访问器（访问器属性）的特性**
- **属性特性描述符及属性特性补充部分**



JS对象及对象属性

• JS对象知识回顾

- JS对象是若干**无序属性的集合**（数据属性、访问器属性、内部属性）
- 生成对象的3种方式：**字面量**直接生成、**Object静态方法**、**构造函数**实例化对象

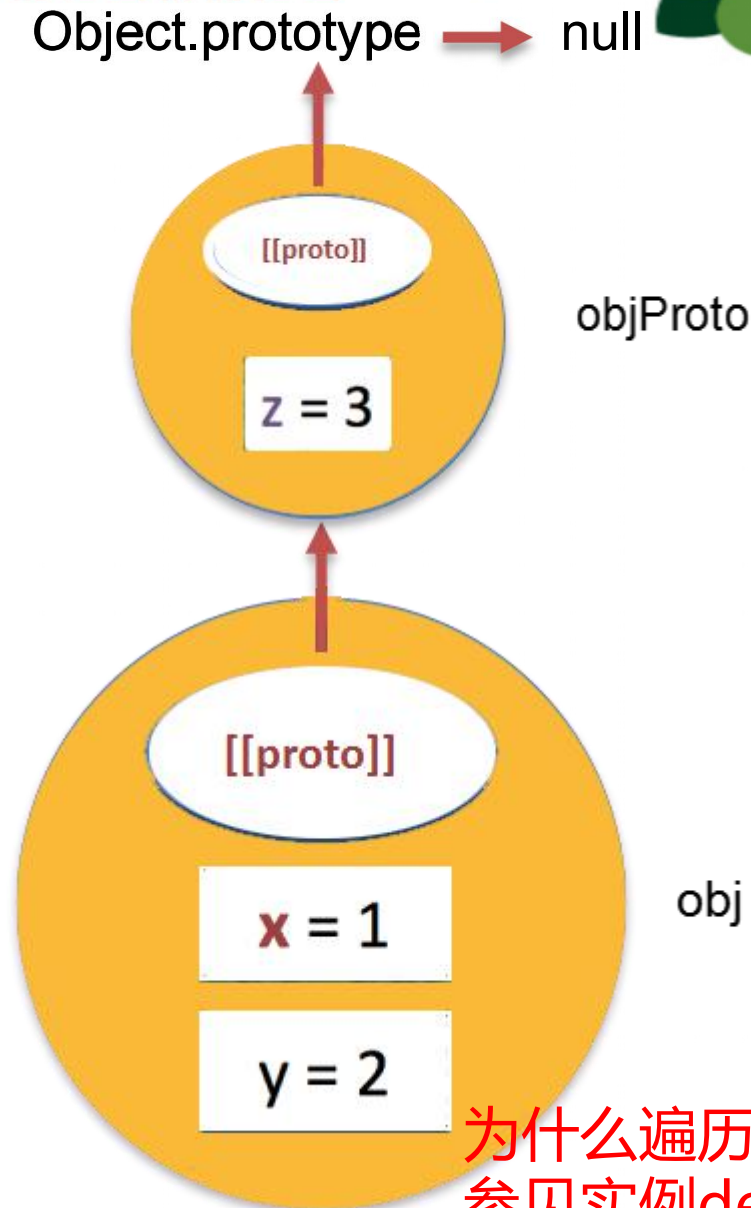
```
var obj = {  
    num:10,  
    str:"Hi",  
    show:function(){  
        console.log(this.str);  
    }  
}
```

```
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();
```

问题：为什么原型链上有些属性遍历不到

```
var objProto = {  
  z:3  
};  
var obj = Object.create(objProto);  
obj.x = 1;  
obj.y = 2;  
console.log(obj.x); //1  
console.log(obj.y); //2  
console.log(obj.z); //3  
console.log(obj.toString); //原型链上有toString  
for(var k in obj){ //可以通过for...in遍历所有属性  
  console.log(k,obj[k]); //是否能遍历到toString?
```



为什么遍历不到某些属性
参见实例demo06

内容提纲

- 对象属性特性简介
- **对象属性（数据属性）的特性**
- 对象访问器（访问器属性）的特性
- 属性特性描述符及属性特性补充部分



JS对象属性特性及其设置

• JS对象属性（数据属性）的特性

- 属性的值（[[value]]），对应属性的值
- 可写特性（[[writable]]），确定属性是否可写性
- 可配置特性（[[configurable]]），确定属性是否能删除和其他特性是否可配置
- 可枚举特性（[[enumerable]]），属性是否可枚举

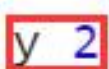
• 设置属性的特性（defineProperty方法设置enumerable）

```
var obj = {  
  x:1,  
  y:2  
};  
for(var k in obj){  
  console.log(k,obj[k]);  
}
```



```
x 1  
y 2
```

```
var obj = {  
  x:1,  
  y:2  
};  
Object.defineProperty(obj,"x",{enumerable:false});  
for(var k in obj){  
  console.log(k,obj[k]);  
}
```



```
y 2
```

参见实例demo07 设置属性的可枚举特性

JS对象属性特性及其设置

- 设置属性特性实例 (writable与configurable)

```
var person = {name:"Jack"};
Object.defineProperty(person, "name", {
    writable:false,
    configurable:false,
    enumerable:true,
    value:"Mike"
});
console.log(person.name); //Mike
person.name = "Lucy";
console.log(person.name); //Still Mike
delete person.name;
console.log(person.name); //Still Mike
```


给对象添加属性（方式1：直接添加）

- 直接给对象添加属性（属性特性默认都为true）

```
var obj = {  
    x:1,  
    y:2  
};
```

```
//直接添加的属性，其所有特性默认都是true  
obj.z = 3;
```

```
for(var k in obj){  
    console.log(k,obj[k]);  
}
```

给对象添加属性（方式2：通过Object.defineProperty添加）

- 通过defineProperty方法添加（属性特性默认为false）

```
var obj = {  
    x:1,  
    y:2  
};  
//直接添加的属性，其所有特性默认都是true  
obj.z = 3;  
//通过defineProperty方法添加的属性，除了手动修改的之外，其他默认都是false  
Object.defineProperty(obj, "w", {  
    value:456, configurable:true  
}); //writable, enumerable没有指定，所以默认为false  
for(var k in obj){  
    console.log(k, obj[k]); //遍历不到"w"  
}
```

内容提纲

- 对象属性特性简介
- 对象属性（数据属性）的特性
- 对象访问器（访问器属性）的特性
- 属性特性描述符及属性特性补充部分



JS对象访问器属性特性及其设置

• JS对象访问器（访问器属性）的特性

- 可配置特性（[[configurable]]），确定属性是否能删除和其他特性是否可配置
- 可枚举特性（[[enumerable]]），属性是否可枚举
- 读取属性特性（[[get]]），在读取属性时调用的函数，默认是undefined
- 写入属性特性（[[Set]]），在写入属性时调用的函数，默认是undefined

```
var obj1={
  _name:"Daisy"
};
Object.defineProperty(obj1,"name",{
  get:function (){//只定义了get 特性，因此只能读不能写
    return this._name;
  }
});
```

使用Object.defineProperty来添加和设置访问器属性特性，注意：通过字面量添加访问器和通过Object.defineProperty方法添加的写法的区别

JS对象访问器属性特性及其设置

• 设置访问器属性特性实例

```
var person = {_name:"jack"};
Object.defineProperty(person, "name", {
    configurable:false, //若为true会怎样?
    enumerable:true,
    set:function(val){this._name = val;},
    get:function(){return this._name;}
});
console.log(person.name);
person.name = "Lucy";
console.log(person.name);
delete person.name; //
console.log(person.name);
```

内容提纲

- 对象属性特性简介
- 对象属性（数据属性）的特性
- 对象访问器（访问器属性）的特性
- 属性特性描述符及属性特性补充部分



属性特性描述符 (Descriptor)

• 什么是属性特性描述符

- 属性特性描述符是一个对象，用来查看对象属性的特性
- 该对象包含4个属性，对应4个特性，通过getOwnPropertyDescriptor方法获得

```
> var obj = {x:5};
    Object.defineProperty(obj,"y",{
      configurable : false,
      writable : false,
      enumerable : true,
      value : 6
    });
    Object.getOwnPropertyDescriptor(obj,"x");
< ▶ Object {value: 5, writable: true, enumerable: true, configurable: true}
> Object.getOwnPropertyDescriptor(obj,"y");
< ▶ Object {value: 6, writable: false, enumerable: true, configurable: false}
```

对象属性特性（补充部分）

- 给多个属性设置特性的方法（Object.defineProperty）

```
var obj = {_x:1};
Object.defineProperty(obj,{
  y:{value:2,writable:true,enumerable:true},
  z:{value:2,writable:true,enumerable:true},
  x:{
    get:function(){return this._x;},
    set:function (val) {
      this._x = val;
    }
  }
});
```

参见实例demo13 给多个属性设置特性和Object.create方法的第二个参数

对象属性特性（补充部分）

- 关于属性特性的继承

```
var o1 = {};  
Object.defineProperty(o1, "x", {  
    value: 12,  
    //writable:true  
}); //思考configurable和writable是true还是false  
o1.x = 34;  
console.log(o1.x);
```

```
var o2 = Object.create(o1);  
o2.x = 56;  
console.log(o2.x); //输出多少？ 打开writable: true后输出多少？
```

对象属性特性（补充部分）

• Object与属性和属性特性相关的方法

- Object.keys(...)、Object.getOwnPropertyNames(...) 区别：是否包含可遍历的属性
- Object.prototype.hasOwnProperty(...) 可结合Object.keys一起使用
- Object.prototype.propertyIsEnumerable(...) (hasOwnProperty的升级版)
- in、for...in (两者关于enumerable的区别) 参见实例demo15 Object相关方法

• JS 对象之扩展、密封及冻结（级别逐渐升高）

- Extensible (Object.isExtensible(), Object.preventExtensions()) 限制添加新属性
- seal (Object.isSealed(), Object.seal()) 在extend的限制基础上，增加限制可配置属性特性
- freeze (Object.isFrozen(), Object.freeze()) 在seal的限制基础上，增加限制可写属性特性

参考链接：<https://segmentfault.com/a/1190000003894119>



The background of the slide is decorated with various abstract shapes in shades of green and yellow. These shapes, which include circles, ovals, and irregular blobs, are scattered across the top and right sides of the slide, creating a modern and artistic feel.

Thank You!