Object.prototype.isPrototypeOf()

isPrototypeOf() 方法用于测试一个对象是否存在于另一个对象的原型链上。

> isPrototypeOf() 与 instanceof 运算符不同。在表达式 "object instanceof AFunction"中，object 的原型链是针对 AFunction.prototype 进行检查的，而不是针对 AFunction 本身。

语法

```
prototypeObj.isPrototypeOf(object)
```

参数

object
在该对象的原型链上搜寻

返回值

Boolean，表示调用对象是否在另一个对象的原型链上。

报错

TypeError
如果 prototypeObj 为 undefined 或 null，会抛出 TypeError 。

描述

isPrototypeOf() 方法允许你检查一个对象是否存在于另一个对象的原型链上。

示例

本示例展示了 Baz.prototype，Bar.prototype，Foo.prototype 和 Object.prototype 在 baz 对象的原型链上：

```
function Foo() {}
function Bar() {}
function Baz() {}

Bar.prototype = Object.create(Foo.prototype);
Baz.prototype = Object.create(Bar.prototype);

var baz = new Baz();

console.log(Baz.prototype.isPrototypeOf(baz)); // true
console.log(Bar.prototype.isPrototypeOf(baz)); // true
console.log(Foo.prototype.isPrototypeOf(baz)); // true
console.log(Object.prototype.isPrototypeOf(baz)); // true
```

如果你有段代码只在需要操作继承自一个特定的原型链的对象的情况下执行，同 `instanceof` 操作符一样 `isPrototypeOf()` 方法就会派上用场，例如，为了确保某些方法或属性将位于对象上。

例如，检查 `baz` 对象是否继承自 `Foo.prototype`：

```
if (Foo.prototype.isPrototypeOf(baz)) {
  // do something safe
}
```

规范

规范版本	规范状态	注解
ECMAScript 3rd Edition (ECMA-262)	ST Standard	初始定义
ECMAScript 5.1 (ECMA-262) Object.prototype.hasOwnProperty	ST Standard	
ECMAScript 2015 (6th Edition, ECMA-262) Object.prototype.hasOwnProperty	ST Standard	
ECMAScript Latest Draft (ECMA-262) Object.prototype.hasOwnProperty	LS Living Standard	

浏览器兼容性

Desktop Mobile

特性	Chrome	Edge	Firefox (Gecko)	Internet Explorer	Opera	Safari
基础支持	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	

相关链接

- `instanceof`
- `Object.getPrototypeOf()`
- `Object.setPrototypeOf()`
- `Object.prototype.__proto__`