链滴

列举几种创建对象的方法，并说明每种方法的使用场景

作者：Vanessa

原文链接：https://ld246.com/article/1548772963809

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

2019-01-28

回答

对象字面量

通常用于存储一次性数据。

```
const person = {
  name: "John",
  age: 50,
  birthday() {
    this.age++
  }
}
person.birthday()
console.log(person.age) // 51
```

构造函数

通常用于为一个对象创建多个实例的场景，每个实例都不会受到该对象的其他实例的影响，他们有自独立的数据。new 关键字必须位于构造函数之前，否则下例中的 name 和 age 将会挂载到 window 。

```
function Person(name, age) {
  this.name = name
  this.age = age
}
Person.prototype.birthday = function() {
  this.age++
}
const person1 = new Person("John", 50)
const person2 = new Person("Sally", 20)
person1.birthday()
console.log(person1.name, person1.age) // John 51
person2.birthday()
console.log(person2.name, person2.age) // Sally 21
```

工厂模式

和构造函数类似，都可以创造一个新的实例，但是他可以通过闭包存储私有数据。在函数调用或 this 键字之前不需要使用 new 操作。工厂模式不使用原型链模式，他将所有属性和方法都做为自己的属 。

```
const createPerson = (name, age) => {
  const birthday = () => person.age++
  const person = { name, age, birthday }
  return person
}
const person = createPerson("John", 50)
person.birthday()
```

```
console.log(person.age) // 51
```

Object.create()

可以设置新创建的对象的原型。Object.create() 的第二个参数可以提供新对象自身定义的可枚举属性。

```
const personProto = {
  birthday() {
    this.age++
  }
}
const person = Object.create(personProto, {
  age: {
    value: 50,
    writable: true,
    enumerable: true
  },
 name: {
    value: 'John',
    writable: true,
    enumerable: true
  }
})
person.birthday()
console.log(person.age, person.name) // 51 "John"
```

加分回答

- 原型链相关的方法可以让一个对象从其他对象中继承属性和方法。
- 工厂模式可以通过闭包提供私有属性和方法，但这会增加内存的使用。
- 类没有私有的属性和方法，他通过复用单个原型对象来减少内存的使用。
- 对以上几种基本的模式可以进行组合使用，从而产生动态原型模式、寄生构造模式、稳妥构造模式。

返回总目录

每天 30 秒