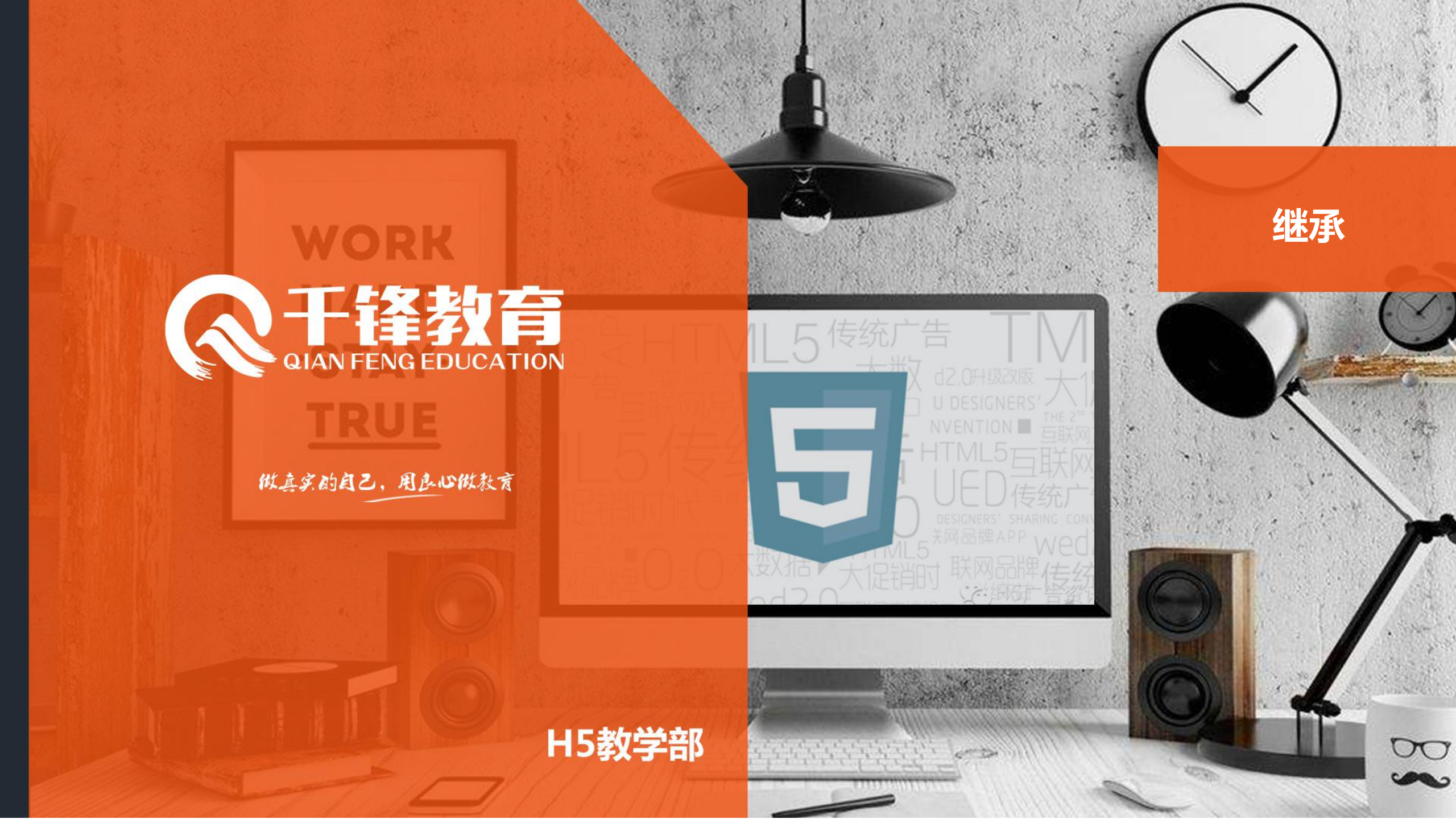


千锋教育
QIAN FENG EDUCATION

做真实的自己，用良心做教育

继承

H5教学部

目录

做真实的自己，用良心做教育

千锋教育
QIAN FENG EDUCATION

继承的介绍

1, 继承的概念

继承是面向对象的一个非常重要的特征. 继承指的是: 子类继承父类的属性和方法.

我们可以通过继承的方式, 在父类的属性和方法基础上, 在无需修改父类的情况下, 让子类也有用这些属性和方法, 并可以扩展其他属性和方法.

继承的特点:

1, 子类拥有父类的属性和方法；

2, 子类可以有自己新的属性和方法；

3, 子类可以重写父类的方法

继承的优点

1, 代码复用：子类可以继承父类的属性和方法

2, 更灵活：子类可以追加或修改属性和方法

继承的介绍

2, 子类和父类

子类: 通过继承创建的新类, 也称“派生类”

父类: 被继承的类, 也称“基类”.

继承的过程，就是从一般到特殊的过程.

例如:

有一个Person类, Employee是员工类，Manager是管理者类，Employee员工类可以继承Person类,因为员工也是人, Manager管理者类可以继承Employee员工类, 因为管理者也是员工。

有一个Animal动物类, Monkey是猴子类, Dog是狗类, 那么Monkey和Dog都是动物,都可以继承Animal动物类的属性和方法,

原型链继承

1, 原型链继承

因为JS没有类, 所以和其他语言的类继承不一样, 我们需要借助原型来实现继承, 称为原型链继承, 即通过原型进行继承.

例如: Employee继承了Person

Person构造函数(Person类)

```
function Person() {
    this.name = 'Zhang';
}
```

Employee构造函数(Employee类)

```
function Employee() {
    this.age = 100;
}
```

Employee继承了 Person，通过原型，形成链条

```
Employee.prototype = new Person();

var employee= new Employee();     //创建Employee对象, 得到被继承的属性
console.log(employee.age);
console.log(employee.name);
```

原型链继承

1, 原型链继承

因为JS没有类, 所以和其他语言的类继承不一样, 我们需要借助原型来实现继承, 称为原型链继承, 即通过原型进行继承.

例如: Employee继承了Person

Person构造函数(Person类)

```
function Person() {
    this.name = 'Zhang';
}
```

Employee构造函数(Employee类)

```
function Employee() {
    this.age = 100;
}
```

Employee继承了 Person，通过原型，形成链条

```
Employee.prototype = new Person();

var employee= new Employee();     //创建Employee对象, 得到被继承的属性
console.log(employee.age);
console.log(employee.name);
```

原型链继承

```
Manager构造函数(Manager类)
function Manager() {
        this.sex = '男';
}
Manager类继承Employee类
Manager.prototype = new Employee();

var manager= new Manager();      //创建manager对象
console.log(manager.name);       //继承了 Employee和 Person
console.log(manager.age);
console.log(manager.sex);
```

原型链继承

所有的类都有一个默认的父类, 那就是Object。

以下四个打印都是true

```
console.log(manager instanceof Manager);
console.log(manager instanceof Employee);
console.log(manager instanceof Person);
console.log(manager instanceof Object);
```

原型链继承

练习
实现以下类的原型链继承

父类：Cat
方法： eat; miaow
属性： fur
说明： eat吃各种东西

子类1： GarfieldCat
方法：eat; miaow; talk
属性：fur; glasses
说明：eat只吃肉

子类2： TomCat
方法：eat; miaow; catchMouse
属性：fur; friend
说明：eat只吃面包

对象冒充

1, 原型链的问题

创建子类的实例对象时，无法向父类的构造函数中传递参数

为了解决这个问题，我们采用一种叫借用构造函数的技术，或者称为对象冒充的技术来解决

对象冒充

2, 对象冒充: 使用构造函数调用call()或者apply()

父类

```
function Person(name, age){
    this.name = name;
    this.age = age;
}
```

子类

```
function Employee(name, age){
    Person.call(this, name, age); //对象冒充
    //Person.apply(this, [name, age]);
}
```

对象冒充

2, 对象冒充: 使用构造函数调用call()或者apply()

父类

```
function Person(name, age){
    this.name = name;
    this.age = age;
}
```

子类

```
function Employee(name, age){
    Person.call(this, name, age); //对象冒充
    //Person.apply(this, [name, age]);
}
```

组合继承

1, 组合继承: 原型链+借用构造函数(对象冒充)的模式

父类

```
function Person(name, age){
    this.name = name;
    this.age = age;
};
Person.prototype.run = function(){
    return this.name + this.age;
};
```

子类

```
function Employee(name, age){
    Person.call(this, name, age);
};
Employee.prototype = new Person(); //原型链继承Person原型中的方法
var employee = new Employee("zhangsan", 100);
var employee2 = new Employee("lisi", 200);
console.log(employee.run());
console.log(employee2.run());
```

组合继承

练习：使用组合继承的方式实现以下类的继承

父类：Cat
方法： eat;
属性： fur
说明： eat吃各种东西

子类： TomCat
方法：eat; catchMouse
属性：fur; friend
说明：eat只吃面包

组合继承

练习

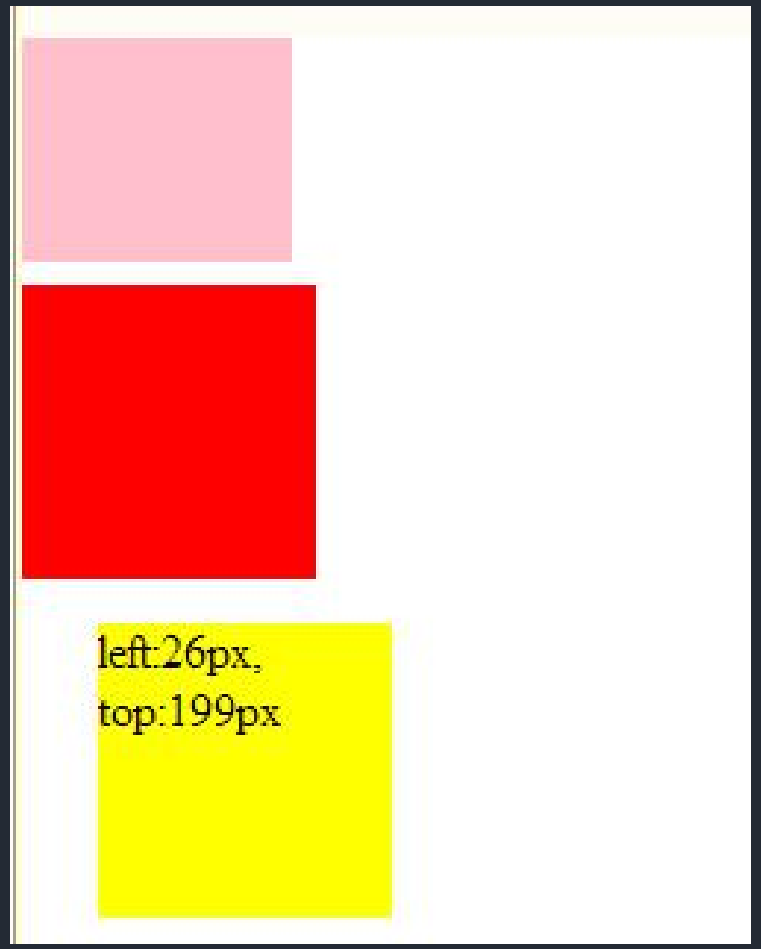
1, 使用组合继承(构造函数+原型链继承+借用构造函数)实现拖拽

有三个div

➢粉色的div可以随意拖拽DragBox;

➢红色的div可以拖拽,但是不能超出左边界和上边界DragBoxLimit;

➢黄色的div在红色div基础上可以显示当前坐标位置DragBoxLimitText;



作业

1, 使用组合继承(构造函数+原型链继承+借用构造函数)实现萤火虫+雪花

这里涉及到两个构造函数: 萤火虫Fireworm和雪花Snow

THANK YOU

千锋教育
QIAN FENG EDUCATION

做真实的自己，用良心做教育