将一个复杂对象的构建与它的表示分离，使得同样的构建过程可以创建不同的表示。屏蔽一个复杂的类的建造过程，通过建造者去实现。

产品类：一般是一个较为复杂的对象，也就是说创建对象的过程比较复杂，一般会有比较多的代码量。在本类图中，产品类是一个具体的类，而非抽象类。实际编程中，产品类可以是由一个抽象类与它的不同实现组成，也可以是由多个抽象类与他们的实现组成。

抽象建造者：引入抽象建造者的目的，是为了将建造的具体过程交与它的子类来实现。这样更容易扩展。一般至少会有两个抽象方法，一个用来建造产品，一个是用来返回产品。

**建造者**：实现抽象类的所有未实现的方法，具体来说一般是两项任务：组建产品；返回组建好的产品。

**导演类**：负责调用适当的建造者来组建产品，导演类一般不与产品类发生依赖关系，与导演类直接交互的是建造者类。一般来说，导演类被用来封装程序中易变的部分

class Product {

private String name;

private String type;

public void showProduct(){

System.out.println("名称："+name);

System.out.println("型号："+type);

}

public void setName(String name) {

this.name = name;

}

public void setType(String type) {

this.type = type;

}

}

abstract class Builder {

public abstract void setPart(String arg1, String arg2);

public abstract Product getProduct();

}

class ConcreteBuilder extends Builder {

private Product product = new Product();

public Product getProduct() {

return product;

}

public void setPart(String arg1, String arg2) {

product.setName(arg1);

product.setType(arg2);

}

}

public class Director {

private Builder builder = new ConcreteBuilder();

public Product getAProduct(){

builder.setPart("宝马汽车","X7");

return builder.getProduct();

}

public Product getBProduct(){

builder.setPart("奥迪汽车","Q5");

return builder.getProduct();

}

}

public class Client {

public static void main(String[] args){

Director director = new Director();

Product product1 = director.getAProduct();

product1.showProduct();

Product product2 = director.getBProduct();

product2.showProduct();

}

}

一个广告有诸多属性，(素材、链接)

一个构造类创建一个广告并赋值相关的属性

具体的director生产具体的广告

Ps层调用