

SA第十一次作业

1、请举例说明克隆模式的其他应用。

(1)数据库中的对象复制：在数据库中，有时候需要创建与现有对象相似的新对象，但是不希望影响到原对象的数据。这时可以使用克隆模式进行对象的克隆，以便在数据库中创建新的对象实例。

(2)文档编辑器中的复制：在文档编辑器中，用户可能需要复制粘贴一些内容，以创建新的文档或段落。克隆模式可以用于复制已有的文档或段落对象，以方便用户创建新的内容。

(3)网络通信中的消息复制：在网络通信中，有时候需要发送与现有消息相似的新消息，但是又不希望影响到原消息的数据。使用克隆模式可以克隆已有的消息对象，以便发送新的消息。

2、试描述浅克隆和深克隆。

(1)浅克隆：在浅克隆中，只复制对象本身及其所有的基本属性，而不会复制对象所引用的其他对象。如果原对象包含引用类型的属性，则浅克隆后的对象与原对象会共享这些引用类型的属性，即它们指向相同的对象。因此，如果修改了其中一个对象所引用的对象，另一个对象也会受到影响。

(2)深克隆：在深克隆中，不仅复制对象本身及其所有的基本属性，还会递归地复制对象所引用的其他对象，包括它们的属性对象，以确保克隆后的对象与原对象完全独立，互不影响。

源码：

Address类：

```
public class Address implements Cloneable{
    private String city;

    public Address(String city) {
        this.city = city;
    }

    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }

    public Object clone() throws CloneNotSupportedException {
        return super.clone();
    }
}
```

Person类：

其中包含一个引用类型Address

浅克隆的代码中直接使用父类的clone()只会复制对象的所有属性，不会复制其中的引用类型，因此克隆后的Address和原来的Address是同一个。

深克隆的代码中手动将引用类型修改为一个克隆后的对象，那么克隆后的Address只有内容和原来一样，地址和原来不同。

```
public class Person implements Cloneable {
    private String name;
    private Address address;

    public Person(String name, Address address) {
        this.name = name;
        this.address = address;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Address getAddress() {
        return address;
    }

    public void setAddress(Address address) {
        this.address = address;
    }

    @Override
    protected Object clone() throws CloneNotSupportedException {
        //浅克隆代码
        //return super.clone();

        //深克隆代码
        Person clonedPerson = (Person) super.clone();
        clonedPerson.address = (Address) this.address.clone();
        return clonedPerson;
    }
}
```

Main函数

```
public class Main {
    public static void main(String[] args) {

        Address address = new Address("New York");

        Person person = new Person("Alice", address);

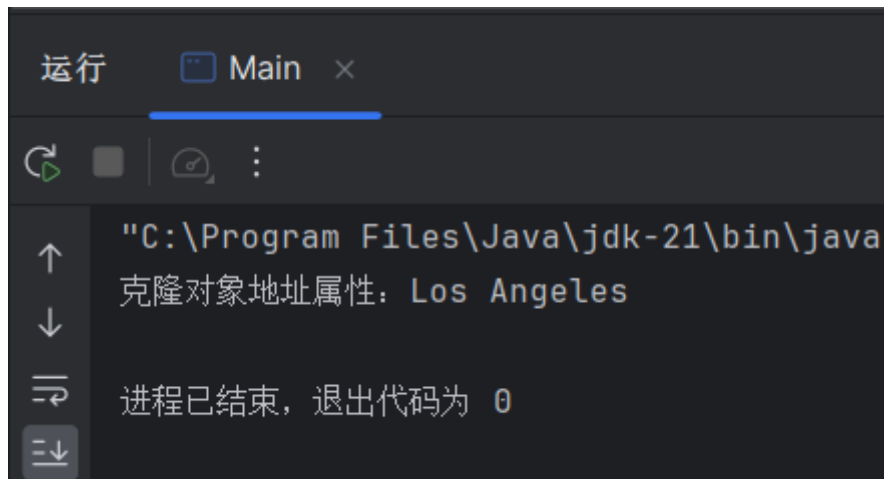
        try {
            Person clonePerson = (Person) person.clone();

            // 修改原对象的地址属性
            person.getAddress().setCity("Los Angeles");
        }
    }
}
```

```
// 输出地址属性
System.out.println("克隆对象地址属性: " + clonePerson.getAddress().getCity()); // 输出
Los Angeles
    } catch (CloneNotSupportedException e) {
        e.printStackTrace();
    }
}
}
```

运行结果:

浅克隆



```
运行 Main x
"C:\Program Files\Java\jdk-21\bin\java
克隆对象地址属性: Los Angeles
进程已结束，退出代码为 0
```

深克隆



```
"C:\Program Files\Java\jdk-21\bin\j
克隆对象地址属性: New York
进程已结束，退出代码为 0
```