适配器模式

适配器模式是一种结构型模式，用来将两个不相关的接口在一起工作。将两个不相关的对象连接在一起的对象我们称作适配器。举一个现实生活中的例子，我们把一个移动充电器当做一个适配器，因为移动电池的充电电压为3V，然而我们常见的插座产生的电压为120V或240V。因此，移动充电器在移动充电插座和墙上插座之间作为适配器工作。

在这个教程中，我们将尝试使用适配器模式实现多个适配器。

首先，我们有两个类-`Volt`和`Socket`。

Vlot.java

```
package com.journaldev.design.adapter;

public class Volt {

    private int volts;

    public Volt(int v){
        this.volts=v;
    }

    public int getVolts() {
        return volts;
    }

    public void setVolts(int volts) {
        this.volts = volts;
    }

}
```

Socket.java

```
package com.journaldev.design.adapter;

public class Socket {

    public Volt getVolt(){
        return new Volt(120);
    }
}
```

现在，我们想构建一个适配器，可以产生3V，12V和默认的120V电压。因此，首先我们创建一个适配器接口。

SocketAdapter.java

```
package com.journaldev.design.adapter;

public interface SocketAdapter {

    public Volt get120Volt();

    public Volt get12Volt();

    public Volt get3Volt();
}
```

两种实现方式

当实现适配器模式时，存在两种途径——类适配器和对象适配器，不过这两种方式产生的结果是相同的。

类适配器实现

SocketClassAdapterImpl.java

```
package com.journaldev.design.adapter;

//Using inheritance for adapter pattern
public class SocketClassAdapterImpl extends Socket implements
SocketAdapter{

    @Override
    public Volt get120Volt() {
        return getVolt();
    }

    @Override
    public Volt get12Volt() {
        Volt v= getVolt();
        return convertVolt(v,10);
    }

    @Override
    public Volt get3Volt() {
        Volt v= getVolt();
        return convertVolt(v,40);
    }

    private Volt convertVolt(Volt v, int i) {
        return new Volt(v.getVolts()/i);
    }

}
```

对象适配器实现

SocketObjectAdapterImpl.java

```
package com.journaldev.design.adapter;

public class SocketObjectAdapterImpl implements SocketAdapter{

    //Using Composition for adapter pattern
    private Socket sock = new Socket();

    @Override
    public Volt get120Volt() {
        return sock.getVolt();
    }

    @Override
    public Volt get12Volt() {
        Volt v= sock.getVolt();
        return convertVolt(v,10);
    }

    @Override
    public Volt get3Volt() {
        Volt v= sock.getVolt();
        return convertVolt(v,40);
    }

    private Volt convertVolt(Volt v, int i) {
        return new Volt(v.getVolts()/i);
    }
}
```

注意，这两种适配器实现大致相同，它们都实现了`SocketAdapter`接口。适配器接口也可以是一个抽象类。

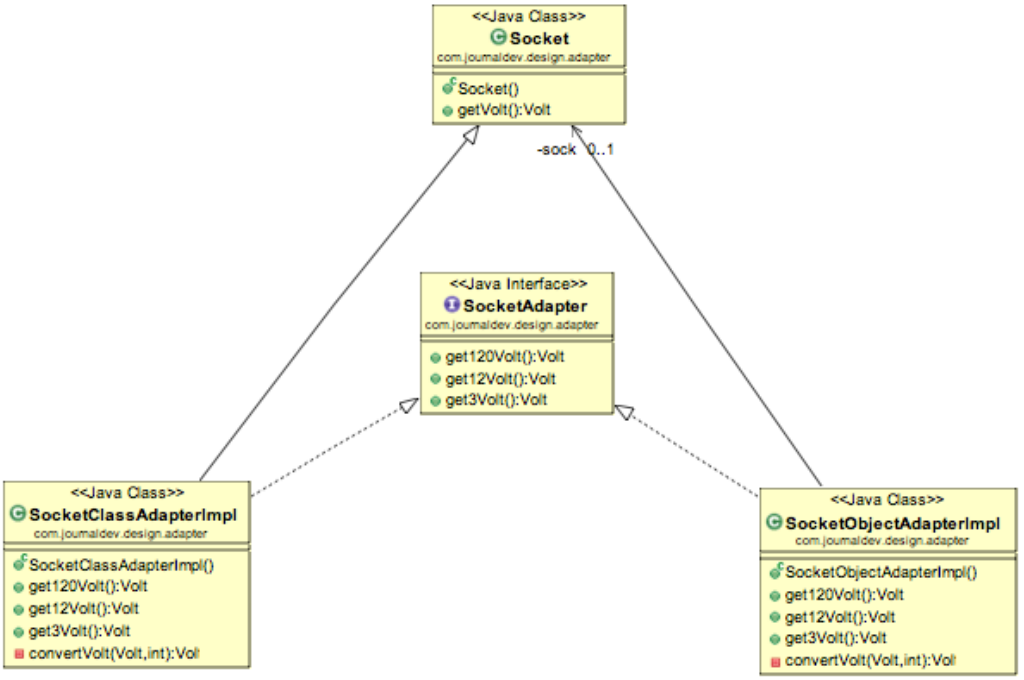
AdapterPatternTest.java

```
package com.journaldev.design.test;

import com.journaldev.design.adapter.SocketAdapter;
import com.journaldev.design.adapter.SocketClassAdapterImpl;
import com.journaldev.design.adapter.SocketObjectAdapterImpl;
import com.journaldev.design.adapter.Volt;

public class AdapterPatternTest {

    public static void main(String[] args) {

        testClassAdapter();
        testObjectAdapter();
    }

    private static void testObjectAdapter() {
        SocketAdapter sockAdapter = new
SocketObjectAdapterImpl();
        Volt v3 = getVolt(sockAdapter,3);
        Volt v12 = getVolt(sockAdapter,12);
        Volt v120 = getVolt(sockAdapter,120);
        System.out.println("v3 volts using Object
Adapter="+v3.getVolts());
        System.out.println("v12 volts using Object
Adapter="+v12.getVolts());
        System.out.println("v120 volts using Object
Adapter="+v120.getVolts());
    }

    private static void testClassAdapter() {
        SocketAdapter sockAdapter = new SocketClassAdapterImpl();
        Volt v3 = getVolt(sockAdapter,3);
        Volt v12 = getVolt(sockAdapter,12);
        Volt v120 = getVolt(sockAdapter,120);
        System.out.println("v3 volts using Class
Adapter="+v3.getVolts());
        System.out.println("v12 volts using Class
Adapter="+v12.getVolts());
        System.out.println("v120 volts using Class
Adapter="+v120.getVolts());
    }

    private static Volt getVolt(SocketAdapter sockAdapter, int i)
{
        switch (i){
        case 3: return sockAdapter.get3Volt();
        case 12: return sockAdapter.get12Volt();
        case 120: return sockAdapter.get120Volt();
        default: return sockAdapter.get120Volt();
        }
    }
}
```

输出：

```
v3 volts using Class Adapter=3
v12 volts using Class Adapter=12
v120 volts using Class Adapter=120
v3 volts using Object Adapter=3
v12 volts using Object Adapter=12
v120 volts using Object Adapter=120
```

适配器模式类图



JDK中适配器模式的例子

- java.util.Arrays#asList();
- java.io.InputStreamReader(InputStream)(returns a Reader)
- java.io.OutputStreamWriter(OutputStream)(returns a Writer)