简单工厂模式

思想：该模式用于完全隔离封装实现内容。只暴露接口给调用者，完全隔离实现内容。

最终实现：工厂调用方法，生成接口对象实例。

效果：

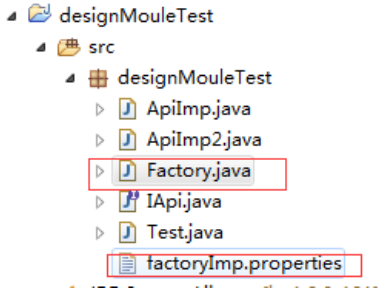
```
3 public class Test {
4
5     public static void main(String[] args) {
6         IApi api=Factory.getApi();
7         String a=api.getIApi();
8         System.out.println(a);
9     }
0 }
1
```

知识点：

- 封装思想的极致运用
- 配置文件参数的调用

如下图：

1.



与调用类放在同一包下（可能与配置中的路径有关）

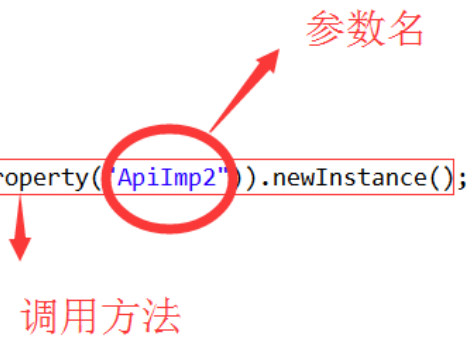
2. 配置文件内容：

```
1 ApiImp2=designMouleTest.ApiImp2
2 ApiImp=designMouleTest.ApiImp
3
```

3. 工厂类调用：

```
public static IApi getApi(){
    //读取properties文件
    Properties properties=new Properties();
    InputStream inputStream=null;
    inputStream=Factory.class.getResourceAsStream("factoryImp.properties");
    try {
        properties.load(inputStream);
    } catch (IOException e) {
       System.out.println("配置信息错误");
        e.printStackTrace();
    }finally {
        try {
            inputStream.close();
        } catch (IOException e) {
            System.out.println("流关闭出错");
            e.printStackTrace();
        }
    }
    IApi api=null;

    try {
        api=(IApi)Class.forName(properties.getProperty("ApiImp2")).newInstance();
    } catch (InstantiationException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (IllegalAccessException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (ClassNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    return api;
}
```



总结:

简单工厂模式使用目的:(封装隔离选择变化)

通过简单工厂模式达到的效果即:根据选择内容,选取实现方式.隔离客户端变化所带来的选择的变化,达到客户端与具体实现的解耦合. 而由于其只暴露同一接口在外部,所以做到了具体实现的封装,隔离.