设计模式实验

刘恒星 2022229044

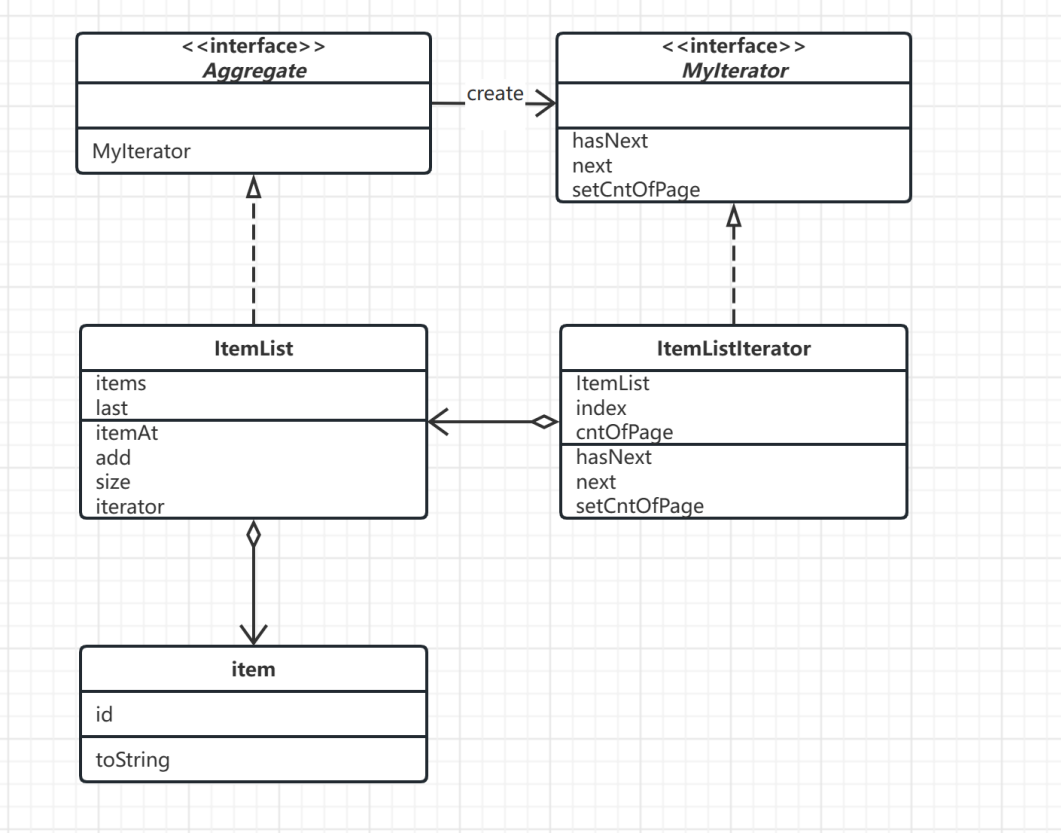
March 20, 2023

1 迭代器模式

实验要求

设计一个逐页迭代器，每次可返回指定个数（一页）元素，并将该迭代器用于对数据进行分页处理。绘制对应的类图并编程模拟实现。

类图设计



代码运行效果

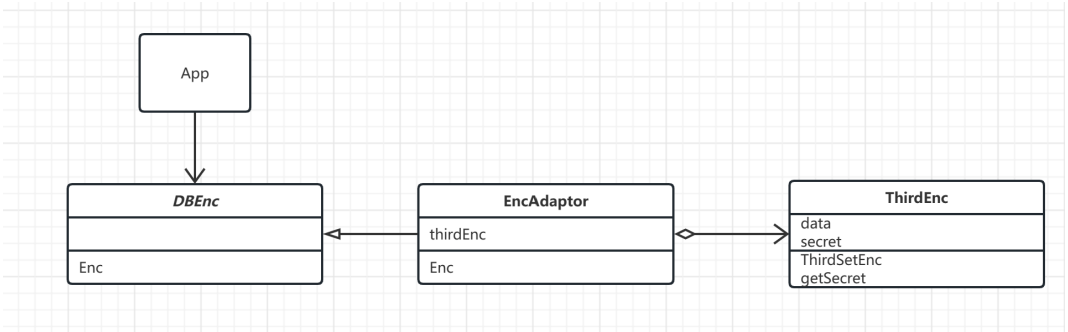
```
在新窗口中打开文件夹 (Ctrl + 单击) pattern\designPatternHW1>  d:; cd 'd:\Desktop\course\hw\design pattern\designPatternHW1'; & 'C:\Program Files\Java\jdk1.8.0_361\bin\java.exe' '-cp' 'C:\Users\CalculuUus\AppData\Roaming\Code\User\workspaceStorage\7aaa8a0753b3c3065b5b310b097d1cfe\redhat.java\jdt_ws\designPatternHW1_edd46259\bin' 'App'
Page start >>>>>>>>>>>>>
item [id=-12]
item [id=57]
Page end <<<<<<<<<<<<<<<<

Page start >>>>>>>>>>>>>
item [id=90]
item [id=-21]
Page end <<<<<<<<<<<<<<<<

Page start >>>>>>>>>>>>>
item [id=-84]
Page end <<<<<<<<<<<<<<<<
```

2 适配器模式

实验要求

某 OA 系统需要提供一个加密模块，将用户机密信息（例如口令、邮箱等）加密之后再存储在数据库中，系统已经定义好了数据库操作类。为了提高开发效率，现需要重用已有的加密算法，这些算法封装在一些由第三方提供的类中，有些甚至没有源代码。试使用适配器模式设计该加密模块，实现在不修改现有类的基础上重用第三方加密方法。要求绘制相应的类图并编程模拟实现，需要提供对象适配器和类适配器两套实现方案。

类图设计



代码运行效果

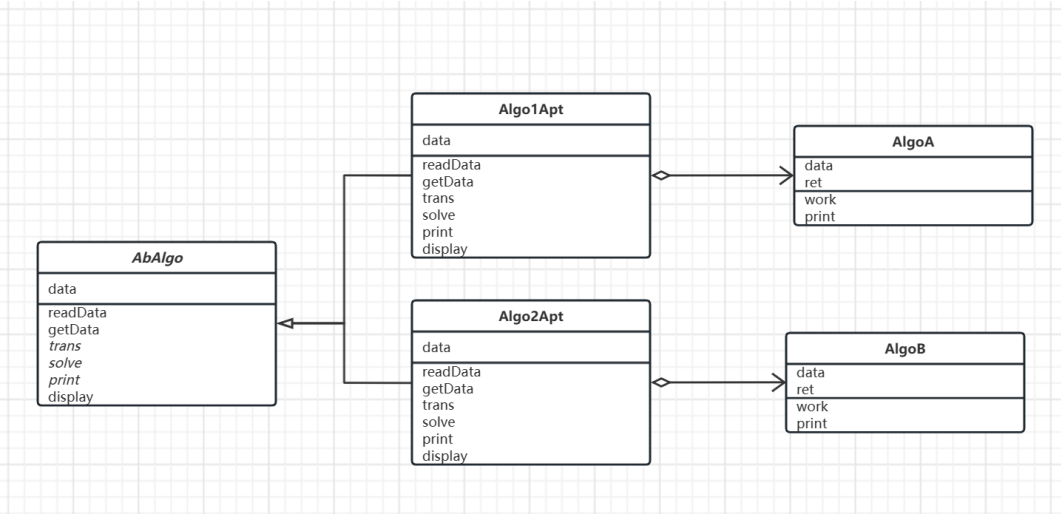
```
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Adaptor>  d:; cd
\Java\jdk1.8.0_361\bin\java.exe' '-cp' 'D:\Desktop\course\hw\design patte
Input string
1234567890
after Enc : 23456789:1
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Adaptor>
```

3　模板方式模式和适配器模式

实验要求

在某数据挖掘工具的数据分类模块中，数据处理流程包括 4 个步骤，分别是： 读取数据；转换数据格式； 调用数据分类算法； 显示数据分类结果。对于不同的分类算法而言，第　步、第　步和第　步是相同的，主要区别在于第　步。第　步将调用算法库中已有的分类算法实现，例如朴素贝叶斯分类（Naive Bayes）算法、决策树（Decision Tree）算法、K 最近邻（K - Nearest Neighbor , KNN）算法等。现采用模板方法模式和适配器模式设计该数据分类模块，绘制对应的类图并编程模拟实现。

类图设计



代码运行效果

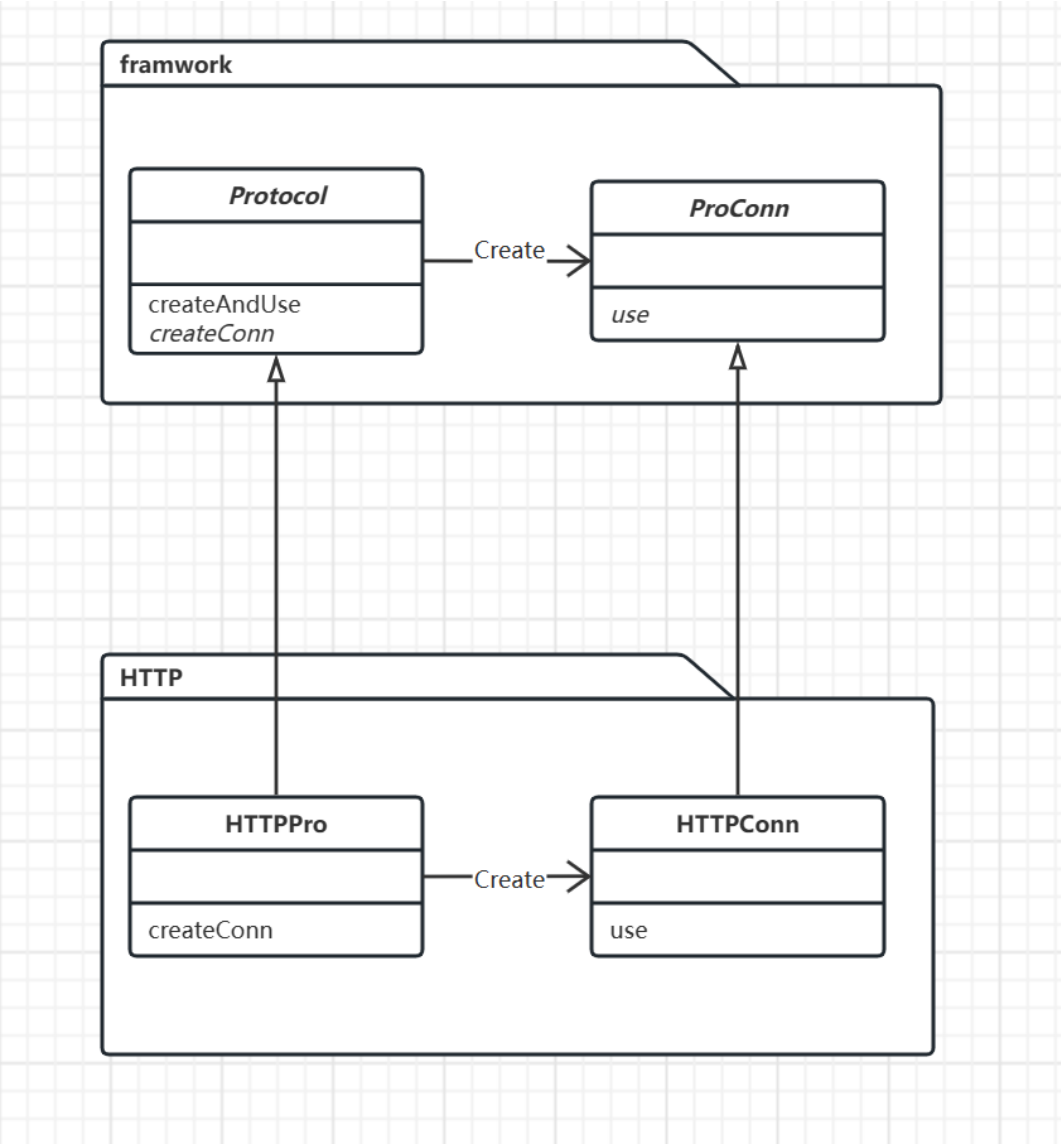
```
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Template>  d:; cd 'd:\Desktc
1.8.0_361\bin\java.exe' '-cp' 'D:\Desktop\course\hw\design pattern\designPatternHW1\
Trans data for Algo 1
ALGO 1
Cat 1:
22
21
61
Cat 2:
76
90
51
61
Trans data for Algo 2
ALGO 2
Cat 1:
16
47
81
87
Cat 2:
65
99
80
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Template>
```

4 工厂方法模式

实验要求

在某网络管理软件中，需要为不同的网络协议提供不同的连接类，例如针对 POP3 协议的连接类 POP3Connection、针对 IMAP 协议的连接类 IMAPConnection 、针对 HTTP 协议的连接类 HTTPConnection 等。由于网络连接对象的创建过程较为复杂，需要将其创建过程封装到专门的类中，该软件还将支持更多类型的网络协议。现采用工厂方法模式进行设计，绘制类图并编程模拟实现。

类图设计



代码运行效果

```
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Factory>  & 'C:\Program Files\Java\jdk1.8.0_361\bin\java.exe' '-cp' 'D:\Desktop\course\hw\design pattern\designPatternHW1\Factory\bin' 'App'
HTTP CONNETING...
HTTP CONNECTING SUCCESS
HTTP Trans Info
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Factory>
```

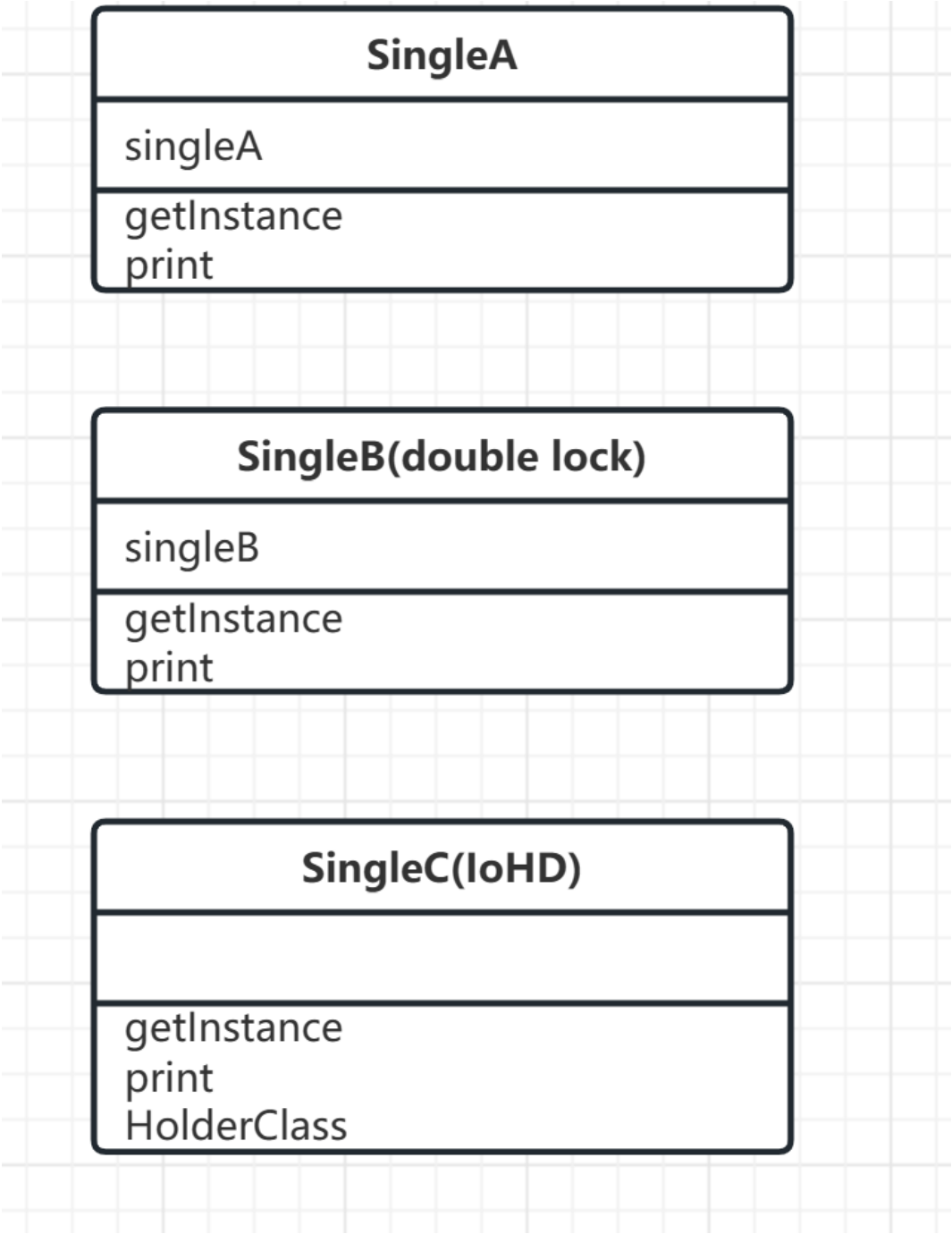
5 单例模式

实验要求

某 Web 性能测试软件中包含一个虚拟用户生成器（Virtual User Generator）。为了避免生成的虚拟用户数量不一致，该测试软件在工作时只允许启动唯一一个虚拟用户生成器。采用单例模

式设计该虚拟用户生成器，绘制类图并分别使用饿汉式单例、双重检测锁和 IoDH 三种方式编程模拟实现。

类图设计



代码运行效果

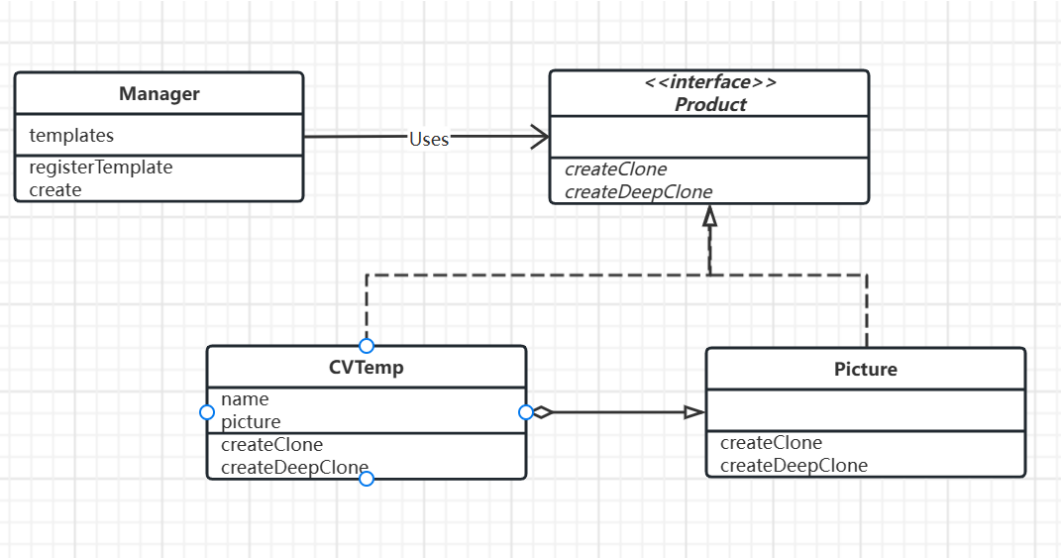
```
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Singleton>  & 'C:\Program Files\Java\jdk1.8.0_361\bin
\java.exe' '-cp' 'D:\Desktop\course\hw\design pattern\designPatternHW1\Singleton\bin' 'App'
饿汉式单例
双重锁单例
IoDH单例
I am singleA
I am singleB
I am singleC
```

6 原型模式

实验要求

在某在线招聘网站中，用户可以创建一个简历模板。针对不同的工作岗位，可以复制该简历模板并进行适当修改后，生成一份新的简历。在复制简历时，用户可以选择是否复制简历中的照片：如果选择“是”，则照片将一同被复制，用户对新简历中的照片进行修改不会影响到简历模板中的照片，对模板进行修改也不会影响到新简历；如果选择“否”，则直接引用简历模板中的照片，修改简历模板中的照片将导致新简历中的照片一同修改，反之亦然。现采用原型模式设计该简历复制功能并提供浅克隆和深克隆两套实现方案，绘制对应的类图并编程模拟实现。

类图设计



代码运行效果

```
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Prototype>  & 'C:\Program Files\Java
gnPatternHW1\Prototype\bin' 'App'
Clone SUCCESS
Deep Clone SUCCESS
PS D:\Desktop\course\hw\design pattern\designPatternHW1\Prototype>
```

7　感想

在这次实验中，学会了这 6 个设计模式的相关知识。初步掌握了这些设计模式的原理，并且能够编写代码。在编写代码的过程中，有时候会有理论和代码设计的脱节，这是在编写代码过程中的困难所在。在逐渐熟练过后能够克服这个困难，更加的了解了这几个设计模式的特点和设计理念。