设计模式

目录

[设计模式 1](#_Toc524070891)

[1 设计模式概述 2](#_Toc524070892)

[1.1 设计模式的概念 3](#_Toc524070893)

[1.2 设计模式的组成 3](#_Toc524070894)

[1.3 GoF 设计模式 3](#_Toc524070895)

[1.4 其他设计模式 3](#_Toc524070896)

[1.5 设计模式与软件架构 3](#_Toc524070897)

[1.6 设计模式分类 3](#_Toc524070898)

[2 设计模式及实现 3](#_Toc524070899)

[2.1 Abstract Factory 模式 3](#_Toc524070900)

[2.2 Singleton 模式 3](#_Toc524070901)

[2.3 Decorator 模式 3](#_Toc524070902)

[2.4 Facade/Session Facade 模式 3](#_Toc524070903)

[2.5 Mediator 模式 3](#_Toc524070904)

[2.6 Observer 模式 3](#_Toc524070905)

[2.7 Intercepting Filter 模式 4](#_Toc524070906)

[3 设计模式总结 21](#_Toc524070907)

# 设计模式概述

* 设计模式基于面向对象技术，是人们在长期的开发实践中良好经验的结晶，提供了一个简单、统一的描述方法，使得人们可以复用这些软件设计办法、过程管理经验。按照设计模式的目的进行划分，现有的设计模式可以分为创建型、（54）和行为型三种类型。其中（55）属于创建型模式，（56）属于行为型模式。（57）模式可以将一个复杂的组件分成功能性抽象和内部实现两个独立的但又相关的继承层次结构，从而可以实现接口与实现分离。

2016年(54)

A.合成型

B.组合型

C.结构型

D.聚合型

2016年(55)

A.Adaptor

B.Facade

C.Command

D.Singleton

2016年(56)

A.Decorator

B.Composite

C.Memento

D.Builder

2016年(57)

A.Prototype

B.Flyweight

C.Adapter

D.Bridge

【答案】C D C D

【解析】 (54)设计模式包括：创建型、结构型、行为型三大类别。 2016年(55)

Singleton是单例模式，属于创建型设计模式。 (56) Memento是备忘录模式，属于行为型设计模式。 2016年(57)

Bridge是桥接模式，它的特点是实现接口与实现分离。

## 设计模式的概念

## 设计模式的组成

## GoF 设计模式

## 其他设计模式

## 设计模式与软件架构

## 设计模式分类

# 设计模式及实现

## Abstract Factory 模式

## Singleton 模式

## Decorator 模式

## Facade/Session Facade 模式

## Mediator 模式

## Observer 模式

## Intercepting Filter 模式

* 某软件公司基于面向对象技术开发了一套图形界面显示构件库VisualComponent。在使用该库构建某图形界面时，用户要求为界面定制一些特效显示效果，如带滚动条、能够显示艺术字体的透明窗体等。针对这种需求，公司采用（60）最为灵活。

2009年(60)

A.桥接模式

B.命令模式

C.组合模式

D.装饰模式

【答案】D 【解析】

根据题干描述，可以看出其基础是一个图形界面，并要求为图形界面提供一些定制的特效，例如带滚动条的图形界面，能够显示艺术字体且透明的图形界面等。这要求能够动态地对一个对象进行功能上的扩展，也可以对其子类进行功能上的扩展。对照选项中的4种设计模式，装饰模式最符合这一要求。

* 某软件公司承接了为某工作流语言开发解释器的工作。该工作流语言由多种活动节点构成，具有类XML的语法结构。用户要求解释器工作时，对每个活动节点进行一系列的处理，包括执行活动、日志记录、调用外部应用程序等，并且要求处理过程具有可扩展能力。针对这种需求，公司采用（61）最为恰当。

2009年(61)

A.适配器模式

B.迭代器模式

C.访问者模式

D.观察者模式.

【答案】C 【解析】

根据题干描述，可以看出本题的核心在于对某个具有固定结构的活动节点需要多种处理能力，且处理能力可扩展，也就是说要求在不改变原来类结构（活动节点）的基础上增加新功能。对照4个选项，发现访问者模式最符合要求。

* 某软件公司欲开发一个Windows平台上的公告板系统。在明确用户需求后，该公司的架构师决定采用Command模式实现该系统的界面显示部分，并设计UML类图如下图所示。图中与Command模式中的“Invoker”角色相对应的类是（30），与“Concrete Command”角色相对应的类是（31）。

2009年(30)

A.Command

B.Menultem

C.Open

D.BulktinBoardScreen

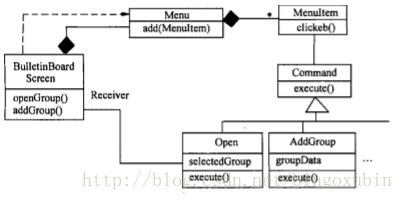
2009年(31)

A.Command

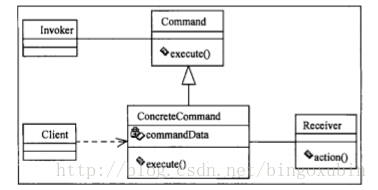
B.Menultem

C.Open

D.BulktinBoardScreen



【答案】B C 【解析】 Command (命令）模式是设计模式中行为模式的一种，它将“请求”封装成对象，以便使用不同的请求、队列或者日志来参数化其他对象。Command模式也支持可撤销的操作。Command模式的类图也如图所示：



对于题目所给出的图，与“Invoker”角色相对应的类是Menultem，与“Concrete Command”角色相对应的类是Open。

* 某公司欲开发一个软件系统的在线文档帮助系统，用户可以在任何一个查询上下文中输入查询关键字，如果当前查询环境下没有相关内容，则系统会将查询按照一定的顺序转发给其他查询环境。基于上述需求，采用(59)最为合适。

2010年(59)

A.责任链模式

B.桥接模式

C.装饰模式

D.适配器模式

【答案】A 【解析】本题主要考查设计模式的理解与应用。

根据题干描述，在线文档系统需要根据用户的查询需求逐步将查询请求依次传递，对比4个候选项，其中在责任链模式中，很多对象由每一个对象对其下家的引用而连接起来形成一条链。请求在这个链上传递，直到链上的某一个对象决定处理此请求。因此责任链模式是能够满足该要求的最好模式。

* 某公司欲开发一套窗体图形界面类库。该类库需要包含若干预定义的窗格(Pane) 对象，例如TextPane、ListPane等，窗格之间不允许直接引用。基于该类库的应用由一个包含一组窗格的窗口组成，并需要协调窗格之间的行为。基于该类库，在不引用窗格的前提下实现窗格之间的协作，应用开发者应采用(60)最为合适。

2010年(60)

A.备忘录模式

B.中介者模式

C.访问者模式

D.迭代器模式

【答案】B 【解析】本题主要考查设计模式的理解与应用。

根据题干描述，应用系统需要使用某公司开发的类库，该应用系统由一组窗格组成，应用需要协调窗格之间的行为，并且不能引用窗格自身，在这种要求下，对比4个候选项，其中中介者模式用一个中介对象封装一系列的对象交互。中介者使用的各对象不需要显式的相互调用，从而使其耦合松散。可以看出该模式最符合需求。

* 某公司开发一个文档编辑器，该编辑器允许在文档中直接嵌入图形对象，但开销很大。用户在系统设计之初提出编辑器在打开文档时必须十分迅速，可以暂时不显示当前页面以外的图形。针对这种需求，公司可以采用(61)避免同时创建这些图形对象。

(61)

A.代理模式

B.外观模式

C.桥接模式

D.组合模式

【答案】A 【解析】本题主要考查设计模式的理解与应用。

根据题干描述，该编辑器需要在文档中高系统效率，需要避免同时创建这些图像。针对这些要求，对比候选项，可以发现代理模式可以解决直接访问对象时带来的问题，例如，要访问的对象在远程的机器上；对象创建开销很大，或者某些操作需要安全控制，或者需要进程外的访问等。因此代理模式是最为合适的设计模式。

* 某银行系统采用Factory Method方法描述其不同账户之间的关系，设计出的类图如下所示。其中与Factory Method中的“Creator”角色相对应的类是(33)；与“Product” 角色相对应的类是(34)。

2010年(33)

A.Bank

B.Account

C.Checking

D.Savings

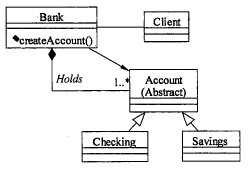
2010年(34)

A.Bank

B.Account

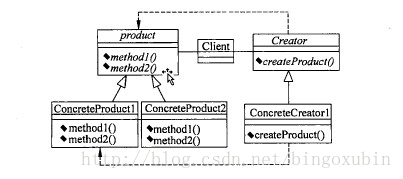
C.Checking

D.Savings



【答案】A B 【解析】 Factory Method模式的意图是，定义一个用于创建对象的接口，让子类决定实例化哪一个类。Factory

Method是一个类的实例化延迟到其子类。Factory Method模式的类图如下图所示。



其中： •类Product定义了Factory Method所创建的对象的接口。 •类 ConcreteProduct

用于实现Product 接口。

•类Creator声明了工厂方法，该方法返回一个Product类型的对象。Creator也可以定义一^工厂方法的缺省实现，它返回一个缺省的ConcreteProduct对象。

•类ConcreteCreator重定义了工厂方法，以返回一个ConcreteProduct实例。

对照两张类图可以看出，与“Creator”角色相对应的类是Bank;与“Product”角色相对应的类是Account。

* 某软件公司正在设计一个通用的嵌入式数据处理平台，需要支持多种数据处理芯片之间的数据传递与交换。该平台的核心功能之一要求能够屏蔽芯片之间的数据交互，使其耦合松散，并且可以独立改变芯片之间的交互过程。针对上述需求，采用（53)最为合适。

2011年(53)

A.抽象工厂模式

B.中介者模式

C.策略模式

D.状态模式

【答案】B 【解析】本题主要考查对设计模式的理解和掌握。

根据题干描述，该系统需要能够支持不同芯片之间的数据交互，并能够独立改变芯片之间的数据交互过程。这种情况下，可以引入一个中介层，通过中介层屏蔽不同芯片之间的两两交互。根据上述分析，选项中列举的设计模式中，中介者模式最符合要求。

* 某软件公司正在设计一个图像处理软件，该软件需要支持用户在图像处理过程中的撤销和重做等动作，为了实现该功能，采用（54)最为合适。

2011年(54)

A.单例模式

B.命令模式

C.访问者模式

D.适配器模式

【答案】B 【解析】本题主要考查对设计模式的理解和掌握。

根据题干描述，系统需要支持用户在图像处理过程中的撤销和重做的动作，因此可以将用户动作封装成对象，通过对象之间的传递和转换实现撤销和重做等动作。根据上述分析，选项中列举的设计模式中，命令模式最符合要求。

* 某互联网公司正在设计一套网络聊天系统，为了限制用户在使用该系统时发表不恰当言论，需要对聊天内容进行特定敏感词的过滤。针对上述功能需求，采用（55)能够灵活配置敏感词的过滤过程。

2011年(55)

A.责任链模式

B.工厂模式

C.组合模式

D.装饰模式

【答案】A 【解析】本题主要考查对设计模式的理解和掌握。

根据题干描述，系统需要对不同的敏感词进行过滤，针对每一个词需要对内容进行分析与过滤，而且需要支持敏感词处理的灵活添加。根据上述分析，选项中列举的设计模式中，责任链模式最符合要求。

* 某公司欲开发一门户网站，将公司的各个分公司及办事处信息进行整合。&决定采用Composite设计模式来实现公司的组织结构关系，并设计了如下图所示的UML类图。图中与Composite模式中的“(Component”角色相对应的类是（33),与“Composite” 角色相对应的类是（34)。

2011年(33)

A.Company

B.Finance

C.HRDepartment

D.Department

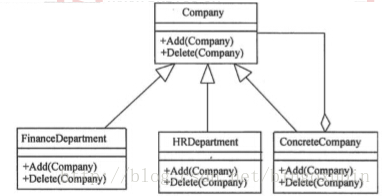
2011年(34)

A.Company

B.FinanceDepartmen

C.HRDepartment

D.ConcreteCompany

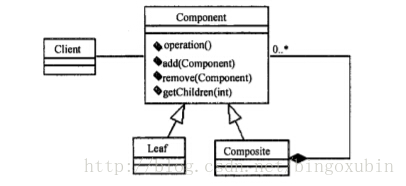


【答案】A D 【解析】

组合（Composite)模式又称为整体-部分（Part-whole)模式，属于对象的结构模式。在组合模式中，通过组合多个对象形成树形结构以表示整体-部分的结构层次。组合模式对单个对象（即叶子对象）和组合对象（即容器对象）的使用具有一致性。Composite模式的结构如下图所示。

•类Component为组合中的对象声明接口，在适当的情况下，实现所有类共有接口的缺省行为，声明一个接口用于访问和管理Component的子部件；

•类Leaf在组合中表示叶结点对象，叶结点没有子结点；并在组合中定义图元对象的行为：



•类Composite定义有子部件的那些部件的行为，存储子部件，并在Component接口中实现与子部件有关的操作；

•类Client通过Component接口操纵组合部件的对象。

根据上述描述可知，与Composite模式中的“Component”角色相对应的类是Company,与 “Composite”角色相对应的类是ConcreteCompany。

* 某公司欲开发一个用于分布式登录的服务端程序，使用面向连接的TCP协议并发地处理多客户端登录请求。用户要求该服务端程序运行在Linux、Solaris和WindowsNT等多种操作系统平台之上，而不同的操作系统的相关API函数和数据都有所不同。针对这种情况，公司的架构师决定采用“包装器外观(Wrapper Facade)”架构模式解决操作系统的差异问题。具体来说，服务端程序应该在包装器外观的实例上调用需要的方法，然后将请求和请求的参数发送给（49），调用成功后将结果返回。使用该模式（50）。

2014年(49)

A.客户端程序

B.操作系统API函数

C.TCP协议API函数

D.登录连接程序

2014年(50)

A.提高了底层代码访问的一致性，但降低了服务端程序的调用性能

B.降低了服务端程序功能调用的灵活性，但提高了服务端程序的调用性能

C.降低了服务端程序的可移植性，但提高了服务端程序的可维护性

D.提高了系统的可复用性，但降低了系统的可配置性

【答案】B A 【解析】本题主要考查考生对设计模式的理解与应用。 针对题目给出的情况，公司的架构师决定采用“包装器外观（Wrapper

Facade)”架构模式解决操作系统的差异问题。具体来说，服务端程序应该在包装器外观的实例上调用需要的方法，然后将请求和请求的参数发送给操作系统API函数，调用成功后将结果返冋。使用该模式提高了底层代码访问的一致性，但降低了服务端程序的调用性能。

* 一组对象以定义良好但是复杂的方式进行通信，产生的相互依赖关系结构混乱且难以理解。采用（32）模式，用一个特定对象来封装一系列的对象交互，从而使各对象不需要显式地相互引用，使其耦合松散，而且可以独立地改变它们之间的交互。

2014年(32)

A.解释器(Interpreter)

B.策略(Strategy)

C.中介者(Mediator)

D.迭代器(Iterator)

【答案】C 【解析】

解释器模式属于类的行为模式，描述了如何为语言定义一个文法，如何在该语言中表示一个句子，以及如何解释这些句子，这里的“语言”是使用规定格式和语法的代码。

策略模式是一种对象的行为型模式，定义一系列算法，并将每个算法封装起来，并让它们可以相互替换。策略模式让算法独立于使用它的客户而变化，其目的是将行为和环境分隔，当出现新的行为时，只需要实现新的策略类。

中介者模式是一种对象的行为行模式，通过一个中介对象来封装一系列的对象交互。中介者使得各对象不需要现实地相互引用，从而使其耦合松散，而且可以独立地改变它们之间的交互。中介者对象的存在保证了对象结构上的稳定，也就说说系统的结构不会因为新对象的引入带来人量的修改工作。

迭代器模式是一种对象的行为型模式，提供了一种方法来访问聚合对象，而不用暴露这个对象的内部表示。迭代器模式支持以不同的方式遍历一个聚合对象。

由上述可知，与题目所描述场景符合的是中介者模式。

* 某广告公司的宣传产品有宣传册、文章、传单等多种形式，宣传产品的出版方式包括纸质方式、CD、DVD、在线发布等。现要求为该广告公司设计一个管理这些宣传产品的应用，采用（33）设计模式较为合适，该模式（34）。

2014年(33)

A.Decorator

B.Adapter

C.Bridge

D.Façade

2014年(34)

A.将一系列复杂的类包装成一个简单的封闭接口

B.将抽象部分与它的实现部分分离，使它们都可以独立地变化

C.可在不影响其他对象的情况下，以动态、透明的方式给单个对象添加职责

D.将一个接口转换为客户希望的另一个接口

【答案】C B 【解析】本题考查设计模式的概念及应用。

题目所给出的应用中，不希望在不同的宣传产品与具体所采用的出版方式之间建立一个固定的绑定关系，以避免这两者之间的紧耦合关系。这种情形适合于采用Bridge(桥接）模式。桥接模式属于结构型设计模式的一种。结构型模式描述如何将类或对象合在一起形成更大的结构。桥接模式将抽象部分与它的实现部分分离，使它们都可以独立地变化。

在以下情况可以使用Bridge模式：

①不希望在抽象以及抽象的实现部分之间有一个固定的绑定关系。例如这种情况可能是因为，在程序运行时刻可以选择或切换实现部分；

②类的抽象以及它的实现都应该可以通过生成子类的方法加以扩充，使用Bridge模式可以对不同的抽象接口和实现部分进行组合，并分别对它们进行扩充。

③对一个抽象的实现部分的修改应该对用户不产生影响，即客户的代码不必重新编译。

* 某软件公司欲开发一个绘图软件，要求使用不同的绘图程序绘制不同的图形。在明确用户需求后，该公司的架构师决定采用Bridge模式实现该软件，并设计UML类图如下图所示。图中与Bridge模式中的“Abstraction”角色相对应的类是（32），与“Implementor”角色相对应的类是（33）。

2015年(32)

A.Shape

B.Drawing

C.Rectangle

D.V2Drawing

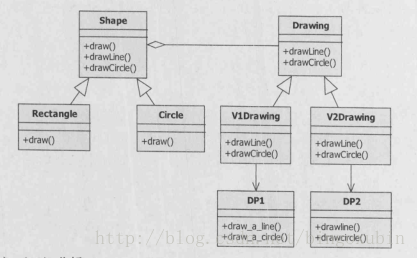
2015年(33)

A.Shape

B.Drawing

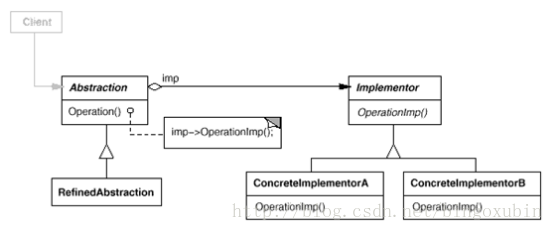
C.Rectangle

D.V2Drawing



【答案】A B 【解析】 桥接模式将抽象部分与它的实现部分分离，使它们都可以独立地变化。它是一种对象结构型模式，又称为柄体(HandleandBody)模式或接口(Interface)模式。桥接模式类似于多重继承方案，但是多重继承方案往往违背了类的单一职责原则，其复用性比较差，桥接模式是比多重继承方案更好的解决方法。

桥接模式的结构如下图所示，其中：



• Abstraction定义抽象类的接口；维护一个指向Implementor类型对象的指针。 •

RefinedAbstraction扩充由Abstraction定义的接口。 •

Implementor定义实现类的接口，该接口不一定要与Abstraction的接口完全一致;

事实上这两个接口可以完全不同。一般来说，Implementor接口仅提供基本操作，而Abstraction则定义了基于这些基本操作的较高层次的操作。

•Concretelmplementor实现Implementor接口并定义它的具体实现。

图中与Bridge模式中的“Abstraction”角色相对应的类是Shape，与“Implementor”角色相对应的类是Drawing。

* 某系统中的文本显示类（TextView)和图片显示类（PictureView)都继承了组件类(Component),分别显示文本和图片内容，现需要构造带有滚动条或者带有黑色边框，或者既有滚动条又有黑色边框的文本显示控件和图片显示控件，但希望最多只增加3个类。那么采用设计模式（35)可实现该需求，其优点是（36)。

2013年(35)

A.外观

B.单体

C.装饰

D.模板

2013年(36)

A.比静态继承具有更大的灵活性

B.提高已有功能的重飪使用性

C.可以将接口与实现相分离

D.为复杂系统提供了简单接口

【答案】C A 【解析】

装饰（Decorator)模式可以再不修改对象外观和功能的情况下添加或考删除对象功能。它可以使用一种对客户端来说是透明的方法来修改对象的功能，也就是使用初始类的子类实例对初始对象进行授权。装饰模式还为对象动态地添加了额外的:重任，这样就在不使用静态继承的情况下，为修改对象功能提供了灵活的选择。

在以下情况中，应该使用装饰模式： •想要在单个对象中动态并且透明地添加责任，而这样并不会影响其他对象；

•想要在以后可能要修改的对象中添加责任； •当无法通过静态子类化实现扩展时。

* 若系统中的某子模块需要为其他模块提供访问不同数据库系统的功能，这些数据库系统提供的访问接口有一定的差异，但访问过程却都是相同的，例如，先连接数据库，再打开数据库，最后对数据进行查询。针对上述需求，可以采用（49)设计模式抽象出相同的数据库访问过程，该设计模式（50)。

2012年(49)

A.外观

B.装饰

C.桥接

D.享元

2012年(50)

A.可以动态、透明地给单个对象添加职责

B.为子系统定义了一个高层接口，这个接口使得这一子系统更加容易使用

C.通过运用共享技术，有效支持大量细粒度的对象

D.将抽象部分与它的实现部分分离，使它们都可以独立地变化

【答案】A B 【解析】

外观（fa9ade)模式是对象的结构模式，要求外部与一个子系统的通信必须通过一个统一的外观对象进行，为子系统中的一组接口提供一个一致的界面，外观模式定义了一个高层接口，这个接口使得这一子系统更加容易使用。

* 某软件公司欲设计一款图像处理软件，帮助用户对拍摄的照片进行后期处理。在软件需求分析阶段，公司的系统分析师识别出了如下3个关键需求：图像处理软件需要记录用户在处理照片时所有动作，并能够支持用户动作的撤销与重做等行为。图像处理软件需要根据当前正在处理的照片的不同特征选择合适的处理操作，处理操作与照片特征之间具有较为复杂的逻辑关系。图像处理软件需要封装各种图像处理算法，用户能够根据需要灵活选择合适的处理算法；软件还要支持高级用户根据一定的规则添加自定义处理算法。在系统设计阶段，公司的架构师决定采用设计模式满足上述关键需求中对系统灵活性与扩展性的要求。具体来说，为了支持灵活的撤销与重做等行为，采用（51)最为合适；为了封装图像操作与照片特征之间的复杂逻辑关系，采用（52)最为合适;为了实现图像处理算法的灵活选择与替换，采用（53)最为合适。

2012年(51)

A.工厂模式

B.责任链模式

C.中介者模式

D.命令模式

2012年(52)

A.状态模式

B.适配器模式

C.组合模式

D.单例模式

2012年(53)

A.模板方法模式

B.访问者模式

C.策略模式

D.观察者模式

【答案】D A C 【解析】本题主要考查设计模式知识。

题干描述了某软件公司一款图像处理软件的需求分析与设计过程，并明确指出采用设计模式实现关键需求对系统灵活性与扩展性的要求。针对需求1，为了支持灵活的撤销与重做等行为，采用命令模式最为合适，因为命令模式可以将一个请求封装为一个对象，从而使你可用不同的请求对客户进行参数化，还可以对请求排队，或记录请求日志，以及支持可撤销的操作。针对需求2,为了封装图像操

作与照片特征之间的复杂逻辑关系，采用状态模式最为合适，因为状态模式将每一个条件分支放入一个独立的类中，这样就可以根据对象自身的情况将对象的状态作为一个对象，这一对象可以不依赖于其他对象而独立变化；针对需求3,为了实现图像处理算法

的灵活选择与替换，采用策略模式最为合适，因为策略模式定义一系列的算法，把它们封装起来，并且使它们可相互替换，使得算法可独立于使用它的客户而变化。

# 设计模式总结