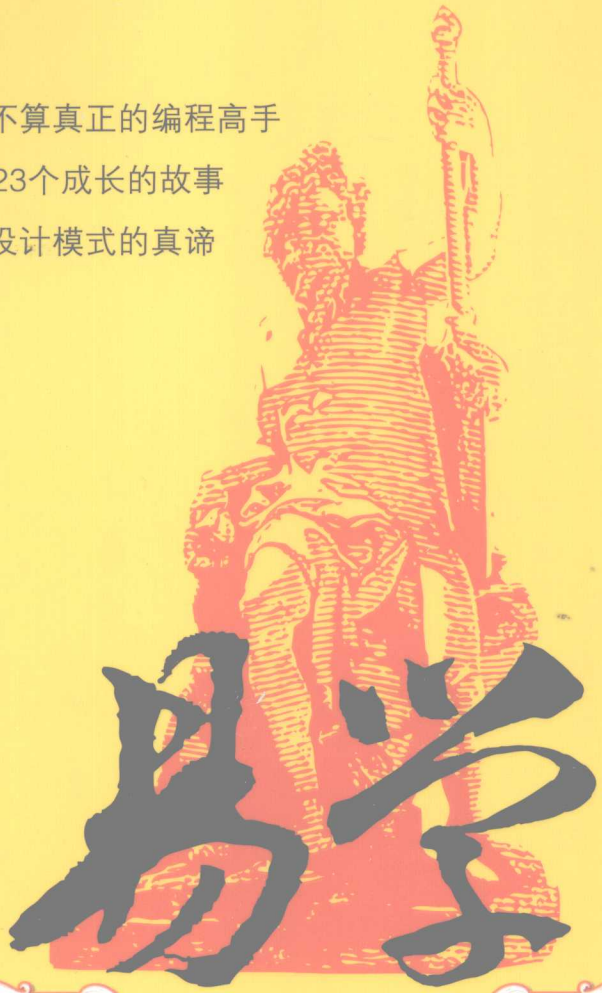
- 不会设计模式就不算真正的编程高手
- 23种设计模式，23个成长的故事
- 对话中轻松掌握设计模式的真谛

书中所有源代码

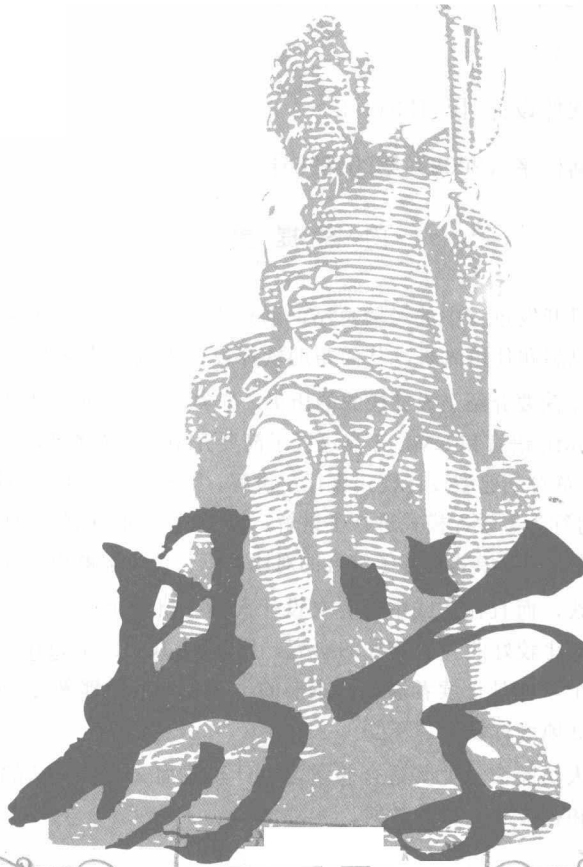


易学

设计模式

郭志学 编著

人民邮电出版社
POSTS & TELECOM PRESS



易学设计模式

郭志学 编著

人民邮电出版社
北京

图书在版编目（CIP）数据

易学：设计模式 / 郭志学编著. —北京：人民邮电出版社，2009.4
ISBN 978-7-115-19552-4

Ⅰ. 易… Ⅱ. 郭… Ⅲ. 软件设计 Ⅳ. TP311.5

中国版本图书馆CIP数据核字（2008）第204826号

内 容 提 要

本书是作者在多年项目开发过程中的经验总结，通过丰富的实例由浅入深、循序渐进地介绍了设计模式的基本原理、核心思想和使用时机，从而帮助软件设计人员快速掌握设计模式的使用方法。

全书共分 27 章。第 1 章简要介绍了设计模式的历史、分类，以及如何学习设计模式和本书学习设计模式的路线图。第 2 章对 UML 语言和 UML 工具进行了简要介绍，使读者能够在后面的章节中建立一个交流的平台。第 3 章～第 8 章对创建型设计模式进行了讲解。第 9 章～第 15 章对结构型设计模式进行了讲解。第 16 章～第 26 章对行为型设计模式进行了讲解。第 27 章着重讲解了面向对象的设计原则、Java 中接口和抽象类的区别，并对设计模式进行了综合的对比，以使读者能够更全面地了解设计模式。

本书既有理论又有实践，而且在实践中既有对设计人员在日常企业应用开发中遇到的实际问题的讲解，又有对一些公认设计比较好的开源软件的研究，比如对 JUnit、Log4j、Spring、Hibernate 等源码进行分析，目的只有一个，就是让读者通过对本书的学习，最终能够熟练地将设计模式应用到设计中，从而帮助设计人员更好地进行设计。

本书适用于软件设计人员阅读，尤其适合想学习设计模式而又不得其法的开发人员阅读，同时也可用作高校相关专业师生和社会培训班的教材。

易学——设计模式

◆ 编　　著　郭志学
责任编辑　屈艳莲
执行编辑　黄　焱

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京铭成印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：25.5
字数：655 千字　　2009 年 4 月第 1 版
印数：1－4 000 册　　2009 年 4 月北京第 1 次印刷

ISBN 978-7-115-19552-4/TP

定价：49.00 元（附光盘）

读者服务热线：(010)67132692　印装质量热线：(010)67129223
反盗版热线：(010)67171154

前　言

不会设计模式就不算真正的编程高手！

尽管精通设计模式很难，但对于本书中的小巩来说却是必须要学习的。

关于本书

小巩大学刚毕业，就进入了某软件公司当程序员。小巩以前已经学习了大半年的 Java 编程，进入公司后又经过了 3 个月的 Java 培训，感觉基础知识掌握得差不多了。但公司的资深人员说不会设计模式就不算真正的编程高手。年轻好胜的他心里就有点纳闷，“不就是设计模式嘛，有那么难？”，于是他就暗下决心，一定要学会设计模式。

刚开始学习设计模式时，小巩阅读了市面上很多讲解设计模式的书籍，但发现这些书要么只讲原理，要么列举的示例不切合实际，让人理解起来很费劲。后来小巩在潜心研究别人著作的同时，也慢慢总结自己的学习心得。如今，设计模式对小巩来说已经是可以拿出来炫耀的资本了。

本书便是小巩学习设计模式的经验和心得体会，书中记录了一个菜鸟成长为技术大牛的历程。

别样风味的讲解模式

- 通过对话展开讲解：本书通过小巩和公司的技术架构师大拿的对话来开展讲解。
- 一一解决小巩在实际开发中遇到的种种问题：通过对这些问题的分析以及大拿对这些问题的解决方案，展示设计模式的好处和学习方法。

独特的内容安排

本书记录了小巩对设计模式的理解过程。书中首先对 GoF 总结出来的 23 种设计模式进行整体介绍，让读者大体了解设计模式的分类。在对具体的设计模式进行讲解前，先介绍了 UML 语言和一些建模工具，使读者先了解工具的使用，以便在后面的讲解中对一些图形的表示达成共识。接着按照创建型模式、结构型模式和行为型模式的分类，对具体的设计模式进行讲解。

前言 Foreword

本书在讲解具体的设计模式时，首先给出不使用这种设计模式时的做法，再给出使用设计模式后的做法，从而让读者能够清楚地认识到使用设计模式的好处；然后对设计模式的定义、原理、使用时机进行总结，并给出一个具体的实例来加深读者对此设计模式的理解；最后给出使用这种设计模式的优缺点。

所有的设计模式都离不开面向对象的思想，本书最后一章对面向对象的设计原则进行了讲解，这是程序设计中最本质的内容。

本书有哪些特色

- 按照创建型、结构型、行为型设计模式的分类进行讲解，并在讲解每个分类的设计模式时按照从易到难的梯度讲解，以降低学习难度。
- 对一些公认较好的开源软件的源码进行分析，例如对 JUnit、Log4j、Spring 和 Hibernate 等源码进行分析，并与实际软件开发相结合。
- 用别开生面的情景模式讲解，增强读者阅读的兴趣。

应该如何阅读本书

- 如果您没有设计模式的基础，最好从本书第 1 章开始学习，首先了解设计模式的历史和分类，然后了解 UML 语言。
- 如果您已经初步了解了设计模式，或者对 UML 语言比较熟悉，则可以跳过第 1 章和第 2 章而直接从第 3 章开始学习，按照创建型模式、结构型模式和行为型模式的分类，对具体的设计模式进行学习。
- 如果您只对某种设计模式感兴趣，也可以单独对该章进行学习。
- 如果您对面向对象的设计原则不太了解，建议先了解一下本书第 27 章中的相关内容，再进入设计模式的学习。
- 如果您已经学习完了本书所讲解的设计模式，建议再次阅读本书第 27 章中所介绍的面向对象的设计原则，这时您可能会有新的感受和发现。

一些心得体会

- 设计模式并不只是一种方法和技术，而更是一种思想、一个方法论。
- 设计模式和具体的语言没有关系，学习设计模式最重要的目的就是要建立面向对象的思想，尽可能地面向接口编程、低耦合、高内聚，使设计的程序可复用。
- 在程序设计时，有些软件开发人员总想着往某个设计模式上套，其实这样是不

对的，这是没有真正掌握设计模式思想的表现。

- 很多时候我们在程序设计时已经用了某种设计模式，自己却不知道这个模式叫什么。因此在程序设计时，关键要考虑如何设计才合理，如何才能复用，如何才能低耦合、高内聚。
- 一些软件开发人员在程序设计时动不动就给类起个类似于模式的名字，比如叫某某 Façade、某某 Factory 等，其实这和设计模式根本没有一点关系。
- 学习设计模式，首先要对设计模式分类，大体上清楚都有哪些设计模式可以使用，然后再分别研究每个设计模式的原理、方法和使用时机。
- 在了解了某个设计模式的使用时机时，还要了解如果不使用这个设计模式会有什么样的后果。
- 当对每个模式的原理和使用方法都理解了以后，更重要的是学习面向对象的思想方式。
- 在掌握面向对象的思想方式后，再回过头来看设计模式，就会有更深刻的理解。
- 学习设计模式，一定要勤学多练。

本书为谁而写

本书并没有创造新的设计模式，也没有超出 GoF 总结的 23 种设计模式的范围，本书记录了小巩学习设计模式的历程，与读者一起分享学习过程的点点滴滴。

如果你是一个想学习设计模式而没有入门的初学者，或者正在为重构一个系统而烦恼，想实现软件的复用而不得其法，那么本书非常适合你，你一定可以在本书中找到解决问题的方法。当然，在阅读本书之前，还需要有一定的 Java 和 UML 基础，最好有一定的软件设计开发经验。

致谢

编写这样的一本书是非常不容易的。特别感谢 51Team 的技术总监方飞先生，当他听说我要编写这样的一本图书时，毫不犹豫地把他多年来在设计模式领域的学习心得贡献了出来，并在我的写作过程中提出了许多宝贵的建议。

本书用到了一些设计模式的定义，有些内容摘自笔者的学习笔记，这些内容可能是当时我在学习时引用的一些著作或资料的描述，请原谅我已经不记得到底是哪些大师讲过或写过的内容了，在此对他们说声感谢！

前言 Foreword

再次感谢所有对本书做出贡献的人们，谢谢你们的支持！

由于作者水平有限，书中错误和疏漏之处在所难免，恳请广大读者批评指正。本书责任编辑的联系方式是 huangyan@ptpress.com.cn，欢迎大家来信交流。

编　者

2009.1

目　录

第 1 篇　设计模式基础

第 2 篇　创建型模式详解

目录 Contents

第 3 篇 结构型模式详解

第 4 篇 行为型模式详解

目录 Contents

第 5 篇 设计模式的综合应用

第1篇 设计模式基础

第1章 欲速则不达：了解设计模式

在建筑工地上，经常看到的是建筑工人在盖房子，却看不到制砖工人在切制砖瓦；在汽车厂房里，经常看到汽车工人在组装汽车，却看不到有工人在制造轮胎。社会化的分工越来越细，这就是当今社会的运转模式，同样，在软件开发过程中，分工也越来越细，有人专门做 UI 设计，有人专门做系统设计，有人专门做代码编写，有人专门做测试，这就是软件开发的模式。分工越细，从事某项分工的人员就越专业，比如程序设计，由于它是把人类思维具体化过程中的一种方法，所以特别不好掌握，因此就有一些专家把他们在解决这一问题过程中积累的一些经验进行总结，以帮助后来人更好地掌握从抽象到具体的方法，就是设计模式。

1.1 小巩的疑惑

小巩大学刚毕业，就进入了某软件公司做程序员，经过 3 个月的 Java 培训后，被分配到了大拿这个项目组，大拿是公司的技术架构师，水平很高。大拿的项目组目前正在给一家企业做人事软件，小巩被安排来开发薪资程序的一小部分。

小巩早就想一显身手了，还没有接到大拿的设计文档，就粗略看了一下需求文档，开始设计开发了。因为这家企业有很多子公司，所以小巩进行了如下设计，首先抽象一个薪资类，然后所有的子公司薪资类都继承这个抽象的薪资类。抽象薪资类的示意代码如下：

```
//******* Salary.java**************
package com.gongdan;
public interface Salary {
    void computerSalary ();
}
```

河北子公司的薪资类示意代码如下：

```
//******* HeBeiSalary.java**************
package com.gongdan;
public class HeBeiSalary implements Salary {
    public void computerSalary() {
        System.out.println("开始计算河北子公司的薪资");
    }
}
```

吉林子公司的薪资类示意代码如下：

```
//******* JiLinSalary.java**************
package com.gongdan;
public class JiLinSalary implements Salary {
```

```
    public void computerSalary() {
        System.out.println("开始计算吉林子公司的薪资");
    }
}
```

这样当客户端调用的时候，首先判断是哪个子公司，然后再新建一个薪资类的对象就可以了，客户端调用的示意代码如下：

```
//******* Client.java**************
package com.gongdan;
public class Client {
    /*根据参数创建不同的对象*/
    public Salary createSalary (String name) {
          Salary salary = null;
          if ("HeBei".equals(name))
              salary = new HeBeiSalary();
          else if ("JiLin".equals(name))
                 salary = new JiLinSalary();
          return salary;
    }
    /*具体的薪资计算*/
    public void  computerSalary (String name) {
          Salary salary = createSalary(name);
          salary. computerSalary();
    }
}
```

小巩很快就完成了设计，看到自己设计的薪资类，小巩心里很高兴，但同时又有了疑问："听说这家公司很快要兼并很多公司，这样一来，它就有了很多的子公司，那客户端的条件判断岂不是也要写很多？代码也肯定越写越长了。"

刚想到这里，大拿的设计方案就送过来了，小巩一看，茅塞顿开："原来可以这样设计啊，高手就是高手。"

大拿是这样来设计的：首先也是一个抽象的薪资类，然后子公司的薪资类都继承这个抽象的薪资类，这点和小巩设计的一样，但是大拿这里又多设计了一个工厂类，把具体产生薪资类的判断语言抽取到工厂类里，并且使用了 Java 的反射机制，而客户端则只负责具体的薪资计算。这样一来类的职责也清晰了，而且由于使用了 Java 的反射机制，代码也变少了。抽象薪资类的示意代码如下：

```
//******* Salary.java**************
package com.gongdan;
public interface Salary {
    void computerSalary ();
}
```

河北子公司的薪资类示意代码如下：

```
//******* HeBeiSalary.java**************
package com.gongdan;
public class HeBeiSalary implements Salary {
    public void computerSalary() {
        System.out.println("开始计算河北子公司的薪资");
    }
}
```

吉林子公司的薪资类示意代码如下：

```
//******* JiLinSalary.java**************
package com.gongdan;
public class JiLinSalary implements Salary {
    public void computerSalary() {
        System.out.println("开始计算吉林子公司的薪资");
    }
}
```

产生薪资类对象的工厂类示意代码如下：

```
//******* SampleFactory.java**************
package com.gongdan;
public class SampleFactory {
    /*根据参数创建不同的对象*/
    public static Salary createSalary (String name) {
          Class c = Class.forName(name);
          Salary salary = (Salary)c.newInstance();
          return salary;
    }
}
```

这样客户端只负责具体的薪资计算而不需要判断要创建哪个子公司的薪资类了，客户端调用的示意代码如下：

```
//******* Client.java**************
package com.gongdan;
public class Client {
    /*具体的薪资计算*/
    public void  computerSalary (String name) {
        Salary salary = SampleFactory.createSalary(name);
        salary. computerSalary();
    }
}
```

说明

上述只是示意代码，实际的薪资计算程序要比这复杂。

小巩看完大拿的设计，心里自愧不如，马上问大拿："大拿，你是如何思考才这样设计的啊？"

大拿说："其实没什么，这是参考设计模式里面的简单工厂模式，随着设计经验的积累，你也会慢慢设计出更好的程序的，你先找些设计模式的书籍看看吧。"

1.2 从历史开始

小巩听完大拿的话，马上上网找了一些设计模式的资料，逐渐对设计模式的历史有了了解。对于大多数经历过软件开发的人来说，都明白这样一个道理：软件的质量在代码编写出来以前，很难进行验证，从而导致软件开发很少有按照项目进度准时完成的。同样的道理在建筑行业也存在：在房子盖起来供人们使用之前，人们很难感受到这个房子设计的好与坏，住着是否舒服，是否满足房子主人的审美观。

为了探索建筑师的建筑设计是否满足人们的需要，在 1979 年，一个叫 Alexander（亚历山大）的建筑师，编写了一本叫《建筑的永恒之道》的书籍，这本书全面地阐述了建筑与规划的新观点，作者通过对当代建筑的研究发现：优秀的建筑中，总是存在着一些相似之处，而优秀建筑的这些

特点是没有办法用名称来命名的。作者指出如果能够找到这些优秀建筑的特征，就能够找到这些建筑是采用了哪些方法，如何设计才变得优秀的，从而也就能够客观地评价一个建筑设计的好与坏。通过找出并掌握这些优秀建筑的设计方法，建筑师就可以把这些方法复制到其他的建筑设计中，从而设计出同样优秀的建筑来。

从上面的介绍可以看出，建筑行业中存在的情况和软件行业存在的情况类似。在软件行业，是否也能够从那些优秀的软件中，找到一些相同的特征、优秀的设计方法，从而使软件开发人员能够掌握这些方法，并将其应用到其他的软件开发中，从而也开发出同样优秀的软件呢？

为了探讨这个问题，从 20 世纪 80 年代开始，就有很多的软件开发人员开始了这方面的研究，其中影响最大的文献是《设计模式》一书，该书由 Gamma、Johnson、Helm、Vlissides 合作编写，发表于 20 世纪 90 年代中期，这 4 个作者也因此被称为 GoF 或 Gang of Four。这本书总结了人类历史上软件开发的经验，给出了描述模式的一些特征，并提炼出用于指导软件设计的 23 种模式和一些面向对象的设计方法。

1.3 设计模式的分类

在了解完设计模式的历史后，小巩又了解了设计模式的分类。这些模式大体上分为 3 类，分别从对象的创建、对象的结构及对象的行为这 3 个方面来总结软件开发人员在设计方面的经验。

1．创建型模式

前面讲过，社会化的分工越来越细，自然在软件设计方面也是如此，因此对象的创建和对象的使用分开也就成为了必然趋势。因为对象的创建会消耗掉系统的很多资源，所以单独对对象的创建进行研究，从而能够高效地创建对象就是创建型模式要探讨的问题。这里有 6 个具体的创建型模式可供研究，它们分别是：

- 简单工厂模式（Simple Factory）；
- 工厂方法模式（Factory Method）；
- 抽象工厂模式（Abstract Factory）；
- 创建者模式（Builder）；
- 原型模式（Prototype）；
- 单例模式（Singleton）。

说明

严格来说，简单工厂模式不是 GoF 总结出来的 23 种设计模式之一。

2．结构型模式

在解决了对象的创建问题之后，对象的组成以及对象之间的依赖关系就成了开发人员关注的焦点，因为如何设计对象的结构、继承和依赖关系会影响到后续程序的维护性、代码的健壮性、耦合性等。对象结构的设计很容易体现出设计人员水平的高低，这里有 7 个具体的结构型模式可供研究，它们分别是：

- 外观模式（Facade）；

- 适配器模式（Adapter）；
- 代理模式（Proxy）；
- 装饰模式（Decorator）；
- 桥模式（Bridge）；
- 组合模式（Composite）；
- 享元模式（Flyweight）。

3．行为型模式

在对象的结构和对象的创建问题都解决了之后，就剩下对象的行为问题了，如果对象的行为设计的好，那么对象的行为就会更清晰，它们之间的协作效率就会提高，这里有 11 个具体的行为型模式可供研究，它们分别是：

- 模板方法模式（Template Method）；
- 观察者模式（Observer）；
- 状态模式（State）；
- 策略模式（Strategy）；
- 职责链模式（Chain of Responsibility）；
- 命令模式（Command）；
- 访问者模式（Visitor）；
- 调停者模式（Mediator）；
- 备忘录模式（Memento）；
- 迭代器模式（Iterator）；
- 解释器模式（Interpreter）。

本书主要讲解的就是 GoF 总结出来的 23 种设计模式，对这些设计模式的原理、使用方式、使用时机等进行讲解，使读者能够熟练掌握它们。

1.4 如何学习设计模式

在了解了设计模式的历史和分类后，应该如何学习设计模式呢？在学习设计模式之前，读者一定要树立一种意识，那就是：设计模式并不只是一种方法和技术，它更是一种思想、一个方法论。它和具体的语言没有关系，学习设计模式最主要的目的就是要建立面向对象的思想，尽可能地面向接口编程、低耦合、高内聚，使你设计的程序尽可能地复用。

有些软件开发人员，在程序设计时，总想着往某个设计模式上套，其实这样是不对的，并没有真正掌握设计模式的思想。其实很多时候读者用了某种设计模式，只是自己不知道这个模式叫什么名字而已。因此，在程序设计时，要根据自己的理解，使用合适的设计模式。

而有另外一些软件开发人员，在程序设计时，动不动就给类起个类似模式的名字，比如叫某某 Façade、某某 Factory 等，其实类里面的内容和设计模式根本没有一点关系，只是用来标榜自己懂设计模式而已。

因此，学习设计模式，首先要了解有哪些方面的设计模式可以供开发人员使用，然后再分别研究每个设计模式的原理，使用时机和方法，也就是说要在什么情况下才使用某个设计模式，在了解某个设计模式的使用时机时，还要了解此时如果不使用这个设计模式，会造成什么样的后果。当对每个模式的原理和使用方法都了解了以后，更重要的是，学习面向对象的思想方式，在掌握面向对象的思想方式后，再回过头来看设计模式，就会有更深刻的理解，最后，学习设计模式，一定要勤学多练。

1.5　本书的学习线路图

本书首先对 GoF 总结出来的 23 种设计模式进行整体介绍，让读者大体了解设计模式的分类。在对具体的设计模式进行讲解前，先讲解了 UML 语言和一些建模工具的使用方法，目的是使读者先了解工具的使用，在后面的讲解中能够对一些图形的表示达成共识。

接着按照创建型模式、结构型模式和行为型模式的分类，对具体的设计模式进行讲解，目的是使读者在学习某一类设计模式时，能够对这个分类下的所有设计模式都有所了解，并能够进行对比分析，认识同一类的设计模式有哪些不同，分别应该使用在哪些不同的场合。

在讲解具体的设计模式时，每章都会用一个初学者和技术大拿对话的方式进行，以提高读者的阅读兴趣。先给出不使用这种设计模式时的做法，再给出使用设计模式后的做法，目的是让读者能够清楚地认识到此时使用设计模式的好处。接着每章会对设计模式的定义、原理、使用时机进行总结，并给出一个具体的实例来加深读者对此设计模式的理解，最后会给出使用这种设计模式的优缺点。

其实，所有的设计模式都离不开面向对象的思想，因此，在本书最后一章，会对面向对象的设计原则进行讲解，这是程序设计中最本质的东西。

1.6　学习设计模式的资源

这里把小巩收集到的一些比较好的资料列出来，供读者进行学习。

（1）学习设计模式最重要的著作：GoF 的《设计模式》。

（2）板里桥人的系列文章：http://www.jdon.com/designpatterns/index.htm。

（3）林信良的系列文章：

http://caterpillar.onlyfun.net/Gossip/DesignPattern/DesignPattern.htm。

（4）Alexander（亚历山大）的《建筑的永恒之道》。

（5）埃克尔的《Java 编程思想》。