

<<代码重构>>

图书基本信息

书名：<<代码重构>>

13位ISBN编号：9787302255550

10位ISBN编号：7302255555

出版时间：2011-6

出版时间：清华大学

作者：阿瑟诺维斯基

页数：498

版权说明：本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：<http://www.tushu007.com>

<<代码重构>>

内容概要

重构是迅速发现并修复有问题的代码的一种高效的方式。

在《代码重构（c# & asp.net版）》中首次提供了在c#和asp.net中进行重构的专业方法，您将学习如何运用重构技术管理和修改代码。

此外，还将学习如何从头开始构建原型应用程序，然后利用重构技术将原型转换为正确设计的、企业级的应用程序。

通过一步一步的指导，您将更好地理解不同的代码问题以及重构的转换。

很多这些转换都是从现实世界的情形中开发出来的，这些都是关键的业务决策的结果。

此外，《代码重构（c#

& asp.net版）》还将介绍重构技术的标准定义，这样您就可以在工作中引用到它。

《代码重构（c# & asp.net版）》

涵盖的重构技术将让您变得效率更高。

您将能使用这些信息对修改做出反应并改进既有代码的设计。

主要内容

- 组装重构工具箱的步骤

- 完成单元测试的技术

- 重构为模式的技巧

- 如何使用重构升级既有的c#和asp.net代码

- 利用方法提取消除重复代码的方式

- 如何让代码变得更简单、更易于修改以及更容易理解

- 所有关于面向对象的理论和设计模式

- 利用linq和其他c#3.0增强功能的技巧

读者对象

《代码重构（c# & asp.net版）》

适用于那些想要学习通过重构工具和功能来高效地管理和修改代码的c#和asp.net开发人员。

<<代码重构>>

作者简介

作者：（美国）阿瑟诺维斯基（Danijel Arsenovski）译者：潘立福 权乐阿瑟诺维斯基（Danijel Arsenovski），是一位作家、软件架构师，也是一位敏捷指导员。

他目前担任Excelsys S . A . 公司的产品和解决方案架构师，主要负责为当地的大量客户设计Web 2 . 0 银行解决方案。

在整改大型银行系统时，他开始尝试重构，并始终对重构保持着浓厚的兴趣。

他因提倡在 . NET平台上运用重构而声名远扬。

Arsenovski是Visual Studio Magazine、 . NET Developers Journal和Visual Systems Journal的撰稿人，拥有Microsoft Certified Solution Developer(MCSD)证书，并被评为2005年的Microsoft MVP。

<<代码重构>>

书籍目录

第1章 重构的全面介绍

1.1 重构的快速浏览

1.1.1 重构过程

1.1.2 软件开发现状概述

1.2 重构过程的详细介绍

1.2.1 代码味道的使用

1.2.2 代码转换

1.2.3 使重构的转换自动化

1.2.4 重构的优点

1.2.5 澄清一些常见的误解

1.3 没有孤军奋战的编程人员

1.4 c#和重构

1.5 小结

第2章 重构的初次体验

2.1 示例应用程序：calories calculator

2.1.1 具有计算推荐每日卡路里量功能的calories calculator应用程序

2.1.2 需求的增长：计算理想的体重

2.1.3 需求的增长：病人数据的持久化

2.2 重构实战

2.2.1 将btncalculate_click方法分解

2.2.2 计算并显示实际体重和理想体重之间差距的片段

2.2.3 按性别计算卡路里和理想的体重

2.2.4 经过方法提取之后的btncalculate_click方法

2.2.5 发现新的类

2.2.6 缩小patient类的接口

2.2.7 重新构建distancefromidealweight方法

2.2.8 创建patient类的层次结构

2.3 持久化功能的实现

2.3.1 保存数据

2.3.2 实现显示病人历史信息的功能

2.4 calories calculator的重构版本

2.5 小结

第3章 组建重构工具箱

3.1 使用自动化的重构工具

3.1.1 jetbrains提供的resharper

3.1.2 developer express提供的refactor !
pro3.1.3 developer express提供的refactor !
for asp

3.1.4 visual studio的重构功能

3.2 单元测试的基本要素：测试用具

3.2.1 单元测试架构出现的原因

3.2.2 nunit的初体验

3.2.3 nunit的安装

3.2.4 使用示例

<<代码重构>>

- 3.2.5 实现第一个测试
- 3.2.6 测试驱动的方法
- 3.2.7 可考虑的其他测试工具
- 3.3 关于版本控制的一些问题
 - 3.3.1 作为备份系统的版本控制
 - 3.3.2 版本控制和并发
- 3.4 小结
- 第4章 应用程序的原型：rent-a-wheels
 - 4.1 会见客户
 - 4.1.1 会见经理
 - 4.1.2 会见前台接待员
 - 4.1.3 会见停车场服务员
 - 4.1.4 会见维护人员
 - 4.2 实施rent-a-wheels项目中最初的步骤
 - 4.2.1 参与者和用例
 - 4.2.2 汽车的状态
 - 4.2.3 应用程序主窗口的第一次草图
 - 4.2.4 rent-a-wheels开发团队的会议
 - 4.3 让原型运转
 - 4.3.1 检查数据库模型
 - 4.3.2 检查c#代码
 - 4.4 快速的编程方法
 - 4.4.1 数据库驱动的设计
 - 4.4.2 基于gui的应用程序
 - 4.4.3 事件驱动的编程
 - 4.4.4 快速应用程序开发
 - 4.4.5 将复制/粘贴作为代码重用的机制
 - 4.4.6 通过重构过程从原型到最后交付
 - 4.5 小结
- 第5章 基本的代码清理
 - 5.1 消除无用代码
 - 5.1.1 无用代码的类型
 - 5.1.2 无用代码常见的来源
 - 5.2 降低过度暴露的元素的作用域和访问级别
 - 5.2.1 作用域和访问级别
 - 5.2.2 过度暴露常见的来源
 - 5.2.3 处理过度暴露的问题
 - 5.3 使用显式导入
 - 5.4 删除未使用的程序集引用
 - 5.5 rent-a-wheels应用程序中的基本清理工作
 - 5.6 小结
- 第6章 从问题域到代码：消除差距
 - 6.1 理解问题域
 - 6.1.1 第1步：收集信息
 - 6.1.2 第2步：就词汇表达成一致意见
 - 6.1.3 第3步：描述交互作用
 - 6.1.4 第4步：建立原型

<<代码重构>>

6.2 命名的指导原则

6.2.1 大写风格

6.2.2 简单的命名指导原则

6.2.3 良好的沟通：选择恰当的单词

6.2.4 "重命名"重构

6.2.5 visual studio中的"重命名"重构

6.3 已发布接口和公有接口

6.3.1 自包含的应用程序与可重用的模块

6.3.2 修改已发布接口

6.4 rent-a-wheels应用程序中的"重命名"和"安全重命名"重构

6.5 小结

第7章 对重复代码进行方法提取

7.1 封装代码和隐藏细节

7.2 分解方法

7.2.1 周长计算--长方法的一个示例

7.2.2 提取周长计算的代码

7.2.3 提取计算半径的代码

7.2.4 提取"等待用户关闭"代码

7.2.5 提取读取坐标的代码

7.2.6 visual studio中的extract method重构

7.3 方法内联化

7.4 重复代码的味道

7.4.1 重复代码的来源

7.4.2 复制/粘贴式编程

7.4.3 幻数

7.5 rent-a-wheels应用程序中的"提取方法"和"用常量取代幻数"重构

7.6 小结

第8章 方法合并与方法提取的技术

8.1 临时变量的处理

8.1.1 "将声明靠近引用处"重构

8.1.2 "将初始化移至声明处"重构

8.1.3 "拆分临时变量"重构

8.1.4 "临时变量内联化"重构

8.1.5 "用查询取代临时变量"重构

8.1.6 引入解释性的临时变量

8.2 处理长条件和嵌套条件

8.3 方法重组与rent-a-wheels

8.3.1 删除rent-a-wheels中的重复代码

8.3.2 rent-a-wheels中的"幻数"、"注释"以及"事件处理盲目性"味道

8.4 小结

第9章 发现对象

9.1 面向对象编程的简单回顾

9.1.1 oop中的对象

9.1.2 封装与对象

9.1.3 visual studio中的"封装字段"重构

9.1.4 对象状态的保持

9.1.5 类

<<代码重构>>

- 9.1.6 对象标识
- 9.1.7 作为基本构建块的对象
- 9.1.8 根对象
- 9.1.9 对象的生存期和垃圾回收
- 9.2 类的设计
 - 9.2.1 使用分析产物
 - 9.2.2 类是名词, 操作是动词
 - 9.2.3 类、责任和协作者
 - 9.2.4 在头脑风暴会议中运用卡片
 - 9.2.5 实体和关系
- 9.3 发现隐藏的类
 - 9.3.1 处理数据库驱动的设计
 - 9.3.2 从过程式设计到面向对象设计的转移
 - 9.3.3 领域层、表示层和持久化层的分离
 - 9.3.4 发现对象与rent-a-wheels应用程序
- 9.4 小结
- 第10章 面向对象的高级概念和相关的重构
 - 10.1 继承、多态性和泛型
 - 10.1.1 继承
 - 10.1.2 类继承与接口继承
 - 10.1.3 多态性
 - 10.1.4 泛型
 - 10.2 继承的滥用和重构解决方案
 - 10.2.1 误用为继承的组合和其他误用情形
 - 10.2.2 继承的重构--打印系统的示例
 - 10.2.3 用委托替代打印系统中的继承
 - 10.3 泛型的使用
 - 10.4 rent-a-wheels应用程序中的继承和泛型
 - 10.4.1 提取超类
 - 10.4.2 运用泛型
 - 10.4.3 提取dataobjectsprovider类
 - 10.5 小结
- 第11章 大规模的代码组织
 - 11.1 命名空间
 - 11.1.1 命名指导原则与命名空间的组织
 - 11.1.2 嵌套的命名空间
 - 11.1.3 修改默认命名空间的名称
 - 11.1.4 使用using指令
 - 11.2 程序集
 - 11.2.1 二进制重用
 - 11.2.2 命名空间组织的指导原则
 - 11.2.3 依赖性方面的考虑
 - 11.3 c#项目文件的结构组织
 - 11.4 rent-a-wheels中命名空间的组织与windows窗体继承
 - 11.4.1 通过抽象窗体辅助类模式提取父管理窗体
 - 11.4.2 命名空间和程序集的重组
 - 11.5 小结

<<代码重构>>

第12章 重构为模式

12.1 什么是设计模式

12.1.1 设计模式的定义

12.1.2 模式的分类

12.1.3 模式的元素

12.1.4 权衡设计模式的利弊

12.1.5 模式的使用

12.2 设计模式的示例：抽象工厂模式

12.2.1 抽象工厂模式的使用

12.2.2 解决方案

12.2.3 结果

12.3 依赖注入模式

12.3.1 使用依赖注入的问题

12.3.2 解决方案

12.3.3 基于构造函数的注入与基于属性的注入

12.3.4 应该注入什么服务实现

12.3.5 di模式的优点

12.3.6 重构成di

12.4 重构成模式与rent-a-wheels应用程序

12.4.1 消除重复.net架构功能的代码

12.4.2 通过依赖注入向gui类中注入data类

12.4.3 crud持久化模式

12.5 小结

第13章 linq和c# 3.0的其他增强功能

13.1 局部变量的类型推断

13.1.1 自动实现的属性

13.1.2 扩展方法

13.1.3 对象、数组和集合的初始化器

13.1.4 通过linq查询对象

13.1.5 旧示例换新颜

13.1.6 通过linq to sql进行对象-关系映射

13.1.7 linq与rent-a-wheels应用程序

13.2 小结

第14章 web技术简史与asp.net重构工具

14.1 refactor !

for asp.net

14.1.1 调用refactor !

for asp.net

14.1.2 refactor !

for asp.net的用户界面

14.2 html的历史及其遗留问题

14.3 紧跟web

14.3.1 visual studio和xhtml

14.3.2 xml和编码

14.3.3 visual studio中html的dtd验证

14.3.4 提供严格的xhtml

14.4 小结

<<代码重构>>

第15章 asp.net应用程序的重构

15.1 html的重构

15.1.1 格式完整的xhtml文档

15.1.2 xhtml的有效性

15.1.3 用于升级遗留的、非遵从xhtml的标记的工具支持

15.1.4 以优美的格式打印html文档

15.1.5 将结构与表示分离

15.1.6 通过rest来使用http

15.2 asp.net代码的重构

15.2.1 asp.net代码模型：单文件和代码隐藏

15.2.2 母版页面

15.2.3 web用户控件与自定义的服务器控件

15.3 rent-a-wheels与asp.net重构

15.4 小结

附录a rent-a-wheels原型的内部机理

附录b refactor !

for asp.net揭密

<<代码重构>>

章节摘录

版权页：插图：

<<代码重构>>

编辑推荐

《代码重构(C#&ASP.NET版)》由清华大学出版社出版。

<<代码重构>>

版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问:<http://www.tushu007.com>