Broadview®
www.broadview.com.cn

PEARSON

Jolt大奖精选丛书

基元设计模式 (英文版)

Elemental Design Patterns

[美] Jason McC. Smith 著

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
www.phei.com.cn

Jolt 大奖精选丛书

基元设计模式（英文版）

Elemental Design Patterns

【美】Jason McC. Smith 著

電子工業出版社
Publishing House of Electronics Industry
北京•BEIJING

内容简介

即使是经验丰富的软件专业人士，也会发现要为其企业找到能带来实质价值的模式应用方式殊非易事。本书首次以全面的方法论介绍基元设计模式，给出标准的命名和描述，阐述它们的重要性，帮助人们比较和选用，充分利用模式的真正力量，将它们转化成实际的、更加简洁直接的软件实现，并得到非常不错的效果。

对于开发工程师、设计师、架构师和分析师，本书都能提供有价值的指导，帮助他们在大多数语言、环境和问题领域使用模式。

图书在版编目（CIP）数据

基元设计模式：英文 /（美）史密斯（Smith,J.M.）著．—北京：电子工业出版社，2013.10
（Jolt 大奖精选丛书）
书名原文：Elemental design patterns
ISBN 978-7-121-21191-1

Ⅰ.①基… Ⅱ.①史… Ⅲ.①软件开发－英文 Ⅳ.①TP311.52

中国版本图书馆 CIP 数据核字 (2013) 第 180560 号

策划编辑：张春雨
责任编辑：许　艳
印　　刷：北京丰源印刷厂
装　　订：三河市鹏成印业有限公司
出版发行：电子工业出版社
　　　　　北京市海淀区万寿路173信箱　　邮编：100036
开　　本：787×980　1/16　印张：22.75　　字数：458.9千字
印　　次：2013年10月第1次印刷
定　　价：75.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

推荐序

读完本书映入我脑海的，是电影《2001太空漫游》（*2001: A Space Odyssey*）的一个精彩场景。

在已遭遗弃的飞船“发现号”上度过了几个月，并把失控的超级电脑Hal制服以后，David Bowman博士来到一块巨石前，然后被带到了一个崭新的世界。他留给地球的最后一句话是：“满目皆星辰！”

软件密集型系统就是我们通过脑力劳动创造出来的崭新世界。Bowman看到的世界由原子构成，所以满目皆星辰；而我们的这个新世界由比特构成，所以——满目皆模式。

无论是有意栽花还是无心插柳，所有结构良好的软件密集型系统里都用到了各种模式。识别系统中的模式，在分析系统时有助于提升抽象的水平；而在系统中运用模式，则有助于让系统变得更有秩序、优雅和简单。以我个人的经验来看，过去的二十多年来，模式是软件工程中最为重要的进步之一。

我有幸与Jason共事，从事改进SPQR的工作，容我向你们保证，他对模式实践背后的思想演进做出了巨大的贡献。本书有助于您打开有关模式的全新思路，以及将模式应用于自己所创造和维护的软件，并促进其改进。如果您对模式还比较陌生，这本书是极好的入门读物；如果您已经熟悉模式，我希望您能温故而知新。我本人就受益良多。

Grady Booch
IBM院士
2012年2月

前　言

本书介绍了一类全新的设计模式，称为基元设计模式（Elemental Design Pattern），其为软件工程设计模式的研究和应用打下了基础。基元设计模式深深植根于软件程序设计理论的研究，但其目的却在于实践性和实用性。从程序设计的新手，到资深的开发工程师，都是基元设计模式的目标受众。它能够带领学生加入软件工业的大军，也能够为研究者打开新的思路。

一言以蔽之，本书的写作目的在于为人所用。

读毕掩卷之时，您将在工具箱中收获一整套全新的工具，您将对日常的程序设计基本概念有更深入的理解，并知道它们是以怎样的方式联系和交互才产生了伟大的创意。基元设计模式，简称EDP，是一组基础的程序设计思想，我们本能地使用它们，而几乎不会去反思。但本书给予了它们正式的描述以及标准的名字，以便在讨论时使用；本书还提供了一个框架来保证概念的一致性，并针对它们各自的特点加以比较。如果您是一个刚入学的学生，您将会了解到，如雨后春笋般涌现的设计模式文献并非要么全盘接受、要么全盘弃用的铁板一块，而是可以将其分解为若干的片段逐个取用，采用科学的方法论来一点点地理解这些鸿篇巨著。如果您早已是软件设计和模式方面的行家里手，也仍然可以找到研究老办法的新视角，并发现我们这个行业里的新机会。

本书假定您对于设计模式领域曾多少有过一些了解和熟悉，但并未深度应用或研究过。仅仅知道存在这么一个领域，并对其有一些道听途说的认知，已经足够开展我们的讨论。本书并不期望您有程序设计理论或语言设计的背景，甚至不要求您精通一门面向对象的程序设计语言，只要心怀渴望，想要学习软件设计的辩证思维即可。在本书中这些主题会有所涉及，但只是以针对想要借由参考文献深入了解相关主题者的出发点论述。本书将采用统一建

模语言（Unified Modeling Language，UML）来描述一些小例子，如果您还不了解 UML，请参考 [20] 或 [33]。您需要有一点点程序设计基础，面向过程或面向对象皆可，最好是后者，但并不是必需的——本书内容已经把面向对象程序设计必知必会的信息打包成易于理解的模块了。经验丰富的面向对象程序开发工程师亦会有惊喜，发现自认为早已掌握的概念仍有新意，从而提升自己对于面向对象程序设计的整体认识。

许多程序员都把“设计模式社区”看作一个专家怪人的群体，认为自己并非其中一员。通过带给您新的视角，让您认识到设计模式是怎么来的，本书应能说服您接受如下观点，即每一位程序员都是设计模式社区的一员，无论其自觉与否。每一位程序员在写下每一行代码的时候，都在运用设计模式，无论其是否这样去想。正因如此，有些选项近在眼前却被人弃如草芥。设计模式是一种共享的概念空间，我们在其中编织自己的电子梦想，并由此改变世界。是时候给出一张标识地界的地图了，这样我们才能更好地工作和享受。

遵循 Gang of Four[21] 树立的典范，本书亦分为两部分。前半部分讲述了本书为何而写，为谁而写，解释了什么是 EDP，它源自何处，为何重要。该部分阐述的是基本理论，是 EDP 背后的“为什么”。该部分还介绍了模式实例记号（Pattern Instance Notation），这是一种能和各种粒度级别上的模式协同工作的图表系统。前半部分的收尾章节讨论了如何使用 EDP 来构建出更大尺度的设计模式，以及如何与其协同工作。后半部分则是设计模式的大阅兵，以 EDP 为起点，通过案例来讲述它们如何以组合方式形成过渡模式，最后，从 EDP 组合的视角重新审视了若干 GoF 设计模式。这里讲述的 EDP 只是全部 EDP 目录的一部分，只是首轮被定义和描述的基础模式集合。软件工程社区将不断地定义和精化更多的 EDP，只要它们背后的概念能够生根发芽。我们期待您能参与其中。

热烈欢迎您的加入！

致　谢

本书得以问世，有太多人要感谢，就让我不按时间先后地致谢吧。

我在北卡罗莱纳州立大学教堂山分校的博士生导师 David Stotts，见证了 SPQR 和 EDP 的诞生过程。我所在的委员会里的同事们，尽管他们一度感觉 EDP 没有什么可行性，仍然认为这会是桩有意义的工作，并给予我自由，让我追梦。Jan Prins、David Plaisted、Al Segars 和 Sid Chatterjee，你们每个人都在关键的时候施以我无价的帮助。

在 IBM 沃森研究中心度过的岁月里，是 Sid Chatterjee 说服了我参与 Big Blue Playpen 项目；Clay Williams 放手让我沉浸在这些天马行空的思想中，我十分怀念与他共度的咖啡时光。Peter Santhanam 支持我的这些思想，而我也从他身上学到了对于遗留系统的敬畏。Brent Hailpern 教给了我很多管理方面的无价经验，我也学到了职场生活的黑色幽默，以及人性的简单。Edith Sconberg 忍受了我的多次恶作剧，本来作为一个老板他大可不必如此。还有好多同事都听我一遍遍地唠叨关于本书的长篇痴语。朋友们，我真想念你们！

还是在 IBM，但我要特别鸣谢的是 Grady Booch，将我爱护于他的羽翼之下，我却无以为报。Grady，您的指导、帮教和建议对我的意义无法度量，唯有期望未来的合作和友谊长青。

位于华盛顿州柯克兰市的 The Software Revolution 公司，是我如今任职资深计算机科学家的单位。我要感谢这里的每一位同事的理解，感谢他们在我发表论文时提供我所需要的信息。我为能与你们共事倍感自豪，我急切地盼望看到我们能一起把公司做到多大。

对本书的多位审阅者，你们的建议和评论具有深刻的洞见，颇有教益。你们使本书增色不少，对下列同仁则致以最高的谢意：Lee Ackerman、Lars Bishop、Robert Bogetti、Robert Couch、Bernard Farrell、Mary Lou Hines

Fritts、Gail Murphy、Jeffrey Overbey、Ethan Roberts、Carlota Sage、Davie Sweis、Peri Tarr 和 Rebecca Wirfs-Brock。Addison-Wesley 出版社的 Elizabeth Ryan、Raina Chrobak、Chris Zahn 和 Chris Guzikowski 在整个成书过程中堪称模范支持团队——在此对你们，以及其余同事致谢，特别要鸣谢的是 Carol Lallier，她对全书的打磨真是无以伦比。

我谨以个人的名义向我的朋友和家人致谢，他们在我投入那似乎遥遥无期的写书工作时表现出了特别的耐心，现在我搬回了西雅图地区，他们总算能和我多见上几面了。

最后要感谢的是我的妻子 Leah。我们在一起的时候，你给了我大大小小的帮助何以计数。你付出的是时间、耐心和爱，我对你的爱和谢意也是无尽的。谢谢你！和这些相比，任何语言都是苍白的。

谢谢所有以你们的方式改进了本书的思想和文字的人们。本书虽然是我的“孩子”，但无疑离不开众多“助产士”的一臂之力。

Jason McC. Smith
西雅图
2011 年 9 月 4 日

关于作者

Jason McC. Smith，2005年毕业于北卡罗莱纳州立大学教堂山分校，获计算机科学博士学位。该校也是基元设计模式的诞生地，基元设计模式当时是模式查询和识别系统（System for Pattern Query and Recognition，SPQR）项目的组成部分。Smith博士因其在校的研究项目而荣获两项美国国家专利，一项与SPQR所采用的技术相关，另一项则来自FaceTop分布式文档协作系统。

此前，Smith博士在物理仿真工程和咨询界工作过多年，取得了华盛顿州立大学的物理学和数学学士学位。值得一提的项目包括声纳和海洋环境仿真、电子工程仿真、商用和军用飞机飞行仿真，以及实时图形训练系统等。

在IBM沃森研究中心工作的四年，使Smith博士有机会将从SPQR和EDP目录中获得的经验加以组织，并应用到大量的软件实体中去，包括遗留系统和现代系统。

Smith博士现在供职于华盛顿州柯克兰市的The Software Revolution公司，任资深研究科学家。在那里，他持续地优化EDP目录，并寻找各种方法来推进公司在自动控制现代化以及遗留系统改造方面的业务目标。

For B.
You were there at the beginning of this journey,
I wish you'd been able to see the end.

Foreword

There's a wonderful scene in the movie *2001: A Space Odyssey* that comes to mind.

Having spent several months alone on the derelict ship *Discovery*—and that after having earlier lobotomized the errant Hal—Dr. David Bowman approaches a monolith that draws him in to a new world. His final message back to earth ends "It's full of stars!"

Software-intensive systems are new worlds that we create with our own mental labor. Whereas the world that Bowman saw was formed from atoms and thus full of stars, our worlds are formed from bits…and are full of patterns.

Whether intentional or not, all well-structured, software-intensive systems are full of patterns. Identifying the patterns in a system serves to raise the level of abstraction in reasoning about that system; imposing patterns on a system serves to bring even further order, elegance, and simplicity to that system. In my experience, patterns are one of the most important developments in software engineering in the past two decades.

I've had the pleasure of working with Jason as he evolved his work on SPQR, and let me assure you that he has contributed greatly to the advance of the understanding and practice of patterns. *Elemental Design Patterns* will help you think about patterns in a new way, a way that will help you apply patterns to improve the software worlds that you create and evolve. If you are new to patterns, this is a great book to start your journey; if you are an old hand with patterns, then I expect you'll learn some new things. I certainly did.

Grady Booch
IBM Fellow
February, 2012

Preface

This book is an introduction to a new class of design pattern, the Elemental Design Patterns, which form a foundation for the study and application of software engineering design patterns. Its foundations are in research into the very fabric of software programming theory, but it is intended to be practical and pragmatic. It is intended for both the beginning programmer and the seasoned developer. It should help students engage with the software industry and give researchers new points to ponder.

In short, this book is meant to be *used.*

By the end of it, you should have a new set of tools in your toolbelt, a richer understanding of some of the basic concepts of programming that we all use every day, and knowledge of how they relate and interact with one another to do amazing things. The Elemental Design Patterns, or EDPs, are a collection of fundamental programming ideas that we use reflexively and probably don't think twice about when doing so. This body of work gives them explicit descriptions, regularized names to use in discussions, and a framework for using them in concert and for comparing them on their own merits. If you're a new student, you'll learn that instead of facing the ever-growing design patterns literature as a collection of daunting all-or-nothing blocks, you have a chance to take them on piece by piece and gradually understand the literature in a methodical way. If you're an old hand at software design and patterns, you'll find new ways to look at old approaches and see new opportunities for our discipline.

This book assumes you have a passing familiarity with design patterns as a field but have not used or studied them in detail. Knowing that they exist and having a brief colloquial knowledge of what they are is enough to start the discussion. The book does not assume you have a background in programming theory, language design, or even a strong one in object-oriented programming, just a desire to learn how to think critically about software design. These subjects will be touched on but only as a starting point for those interested in diving deeper into them through the provided references. The Unified Modeling Language is used to describe small examples, and I suggest either [20] or [33] as references if you do not already know UML. You should have a basic foundation in programming, either procedural or object oriented. The latter will help, but it's not absolutely required—this text provides much of the necessary information to explain object-oriented programming in easily digestible chunks. Developers experienced with object-oriented systems may still be surprised at finding new perspectives on concepts that they thought they had mastered long ago and a greater appreciation for object-oriented programming as a whole.

Many programmers see the "design patterns community" as an esoteric body of experts and one that they themselves are not a part of. By giving you a new perspective on what can constitute a design pattern, this book should convince you that *every* programmer is a member of the design patterns community, whether they know it or not. Every single programmer uses design patterns every time they write a line of code, even if don't think of it that way. Nor are they likely to realize the options they have at their disposal. Design patterns are the shared conceptual space in which we write the electronic dreams that shape our world. It's time we had a map of the landscape in which we work and play.

Following the example of the seminal Gang of Four text [21], this book is divided into two sections. First is a discussion of why this book was written and who it is written for and an explanation of what EDPs are, where they came from, and why they're important. This section explains the rationale, the *why*, behind the EDPs. Next is an introduction to the Pattern Instance Notation, a diagramming system for working with patterns at many levels of granularity and in a multitude of environments. Wrapping up this first section is a discussion of how EDPs can be used to build up to, and in conjunction with, the greater design patterns literature. The second section of the book is a collection of design patterns, starting with the EDPs and working through examples of how they combine to form Intermediate patterns, and finally, a selection of the Gang of Four patterns recast as EDP compositions. The EDPs presented here are only a portion of the EDP Catalog, a collection of the first round of defined and described fundamental patterns. The software engineering community will continue to define and refine additional EDPs as the underlying concepts take root. I hope you decide to help in the endeavor.

Welcome, it's good to have you join us.

Acknowledgments

I have many people to thank for this book coming to life. In not quite chronological order...

From the University of North Carolina at Chapel Hill, David Stotts, my Ph.D. advisor who oversaw the birth of SPQR and the EDPs over many years; also my committee, who, even though they were convinced it was probably infeasible, thought it would be an interesting journey and let me go for the brass ring anyway: Jan Prins, David Plaisted, Al Segars, and Sid Chatterjee. You each added invaluable help at critical times.

From my years at IBM Watson Research in New York, Sid Chatterjee again, who convinced me to come play in the Big Blue Playpen; Clay Williams, who gave me free rein to pursue these crazy ideas further and with whom I still miss having coffee; Peter Santhanam, who championed those ideas and from whom I learned a greater appreciation for legacy systems; Brent Hailpern, from whom I learned many valuable lessons in management, the dark humor of corporate life and simple humanity; Edith Schonberg, who put up with my shenanigans more than any manager should have to; and many others who listened to me maniacally talk about this body of work at every turn. My friends, I miss you all.

Also from IBM but deserving a special mention, Grady Booch, who took me under his wing for a wild ride that I wouldn't have traded for anything. Grady, your guidance, mentoring, and advocacy have been immeasurable, and I look forward to future collaborations and continued friendship.

From The Software Revolution, Inc., in Kirkland, Washington, where I am now Senior Computer Scientist, I have to thank everyone for being understanding and supportive of my need to commit this information to paper. It has been a true pleasure working with all of you, and I am eager to see where we can take our company.

To my many reviewers, your advice and comments were highly insightful and helpful. You made this book a much better product, and you have my deepest thanks: Lee Ackerman, Lars Bishop, Robert Bogetti, Robert Couch, Bernard Farrell, Mary Lou Hines Fritts, Gail Murphy, Jeffrey Overbey, Ethan Roberts, Carlota Sage, Davie Sweis, Peri Tarr, and Rebecca Wirfs-Brock. Elizabeth Ryan, Raina Chrobak, Chris Zahn, and Chris Guzikowski at Addison-Wesley were the model of compassionate support during the trials of this process—my thanks to you and the rest of the crew there, with a special thanks to Carol Lallier, whose expert polish on this book was invaluable.

On a personal note, I thank my friends and family, who have been incredibly patient while I have put in seemingly endless hours on this, even though they were hoping to see more of me now that I've moved back to the Seattle area.

Finally, my wife Leah. You have supported me in so many large and small ways throughout our time together. You have given your time, your patience, and your love, and you have my immense love and gratitude. Thank you. Words are simply inadequate.

Thank you all. Every one of you contributed in some way to the refinement of these ideas and this text. This may have been my baby, but it had many midwives.

—Jason McC. Smith
Seattle
September 4, 2011

About the Author

Jason McC. Smith received his Ph.D. in computer science in 2005 from the University of North Carolina at Chapel Hill, where the Elemental Design Patterns were born as part of the System for Pattern Query and Recognition project. Dr. Smith has been awarded two U.S. patents for research performed at UNC-CH, one for technologies related to SPQR and one for the FaceTop distributed document collaboration system.

Prior to that, Dr. Smith spent many years in industry as a physics simulation engineer and consultant building off of dual B.Sc. degrees in physics and mathematics from the University of Washington. Projects of note included sonar and oceanic environment simulation, electronic engineering simulation, commercial and military aircraft flight simulation, and real-time graphical training systems.

Four years at IBM Watson Research provided Dr. Smith with an opportunity to apply the lessons of SPQR and the EDP catalog and compositional approach to immense bodies of software, both legacy and modern.

Dr. Smith is currently Senior Research Scientist at The Software Revolution, Inc., in Kirkland, Washington, where he continues to refine the EDP catalog and look for ways to enhance the company's goal of automated modernization and transformation of legacy systems.

Contents