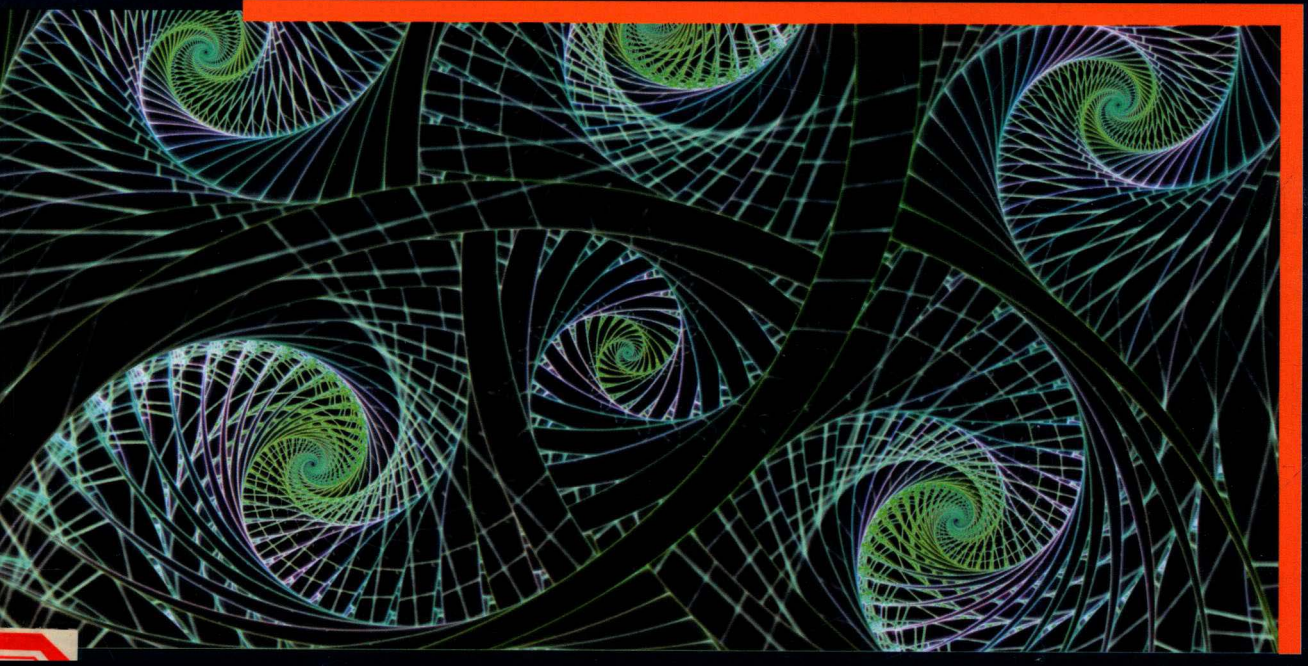
华章程序员书库

HZ BOOKS
华章IT

Julia语言程序设计

Introduction to Julia Programming



零基础学习Julia语言，基于版本v1.0

系统全面地介绍Julia编程语言，从基本概念到编程要点，从多维数组到并行计算，包含大量示例代码，以及机器学习综合案例

魏坤 编著

机械工业出版社
China Machine Press

华章程序员书库

Julia语言程序设计

魏坤 编著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

Julia 语言程序设计 / 魏坤编著 . —北京：机械工业出版社，2018.10
（华章程序员书库）

ISBN 978-7-111-60757-1

I. J… II. 魏… III. 程序语言 – 研究 IV. TP312

中国版本图书馆 CIP 数据核字（2018）第 195441 号

Julia 语言程序设计

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码：100037）
责任编辑：冯秀泳　　责任校对：李秋荣
印　刷：北京市兆成印刷有限责任公司　　版　次：2018 年 10 月第 1 版第 1 次印刷
开　本：186mm × 240mm 1/16　　印　张：27.5
书　号：ISBN 978-7-111-60757-1　　定　价：99.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：（010）88379426 88361066　　投稿热线：（010）88379604
购书热线：（010）68326294 88379649 68995259　　读者信箱：hzit@hzbook.com

Preface 前　言

几十年前，科学家为了避免处理重复、单调的事情，比如反复地按一套公式计算结果等，发明了计算机。计算机其实是工业自动化的一个产物，可以说是工业化时代的巅峰代表。当时人们怎么也不会想到，计算机的发展带来了互联网，而互联网导致了信息化时代的到来。如今，在数据蔓延、不断渗透的过程中，智能化代表了未来的发展方向。

与此同时，数据的不断累积、膨胀、延伸引发了计算领域的深刻变化，而且数据的密集性和分布性也提出了大量的计算密集性和分布式要求。很多工业级的生产场景中，在要求开发高效率、维护低成本、运行高可靠的同时，还需要具备高性能的特点。Julia 语言应运而生了。

为何撰写本书

Julia 借助于 JIT 动态解析器及其优秀的设计机制，在一些计算特性上能达到静态语言的性能，这是非常令人惊讶的，也是吸引笔者的地方。笔者在大数据挖掘与机器学习领域浸淫十几年，面对种类繁多、数量巨大、计算逻辑复杂的各种问题，深感性能与开发效率极为重要。本想浅尝，但却再无法舍弃，Julia 语言的各种特性令笔者兴奋不已。

几年前我开始接触 Julia，其自然快捷的编写感受，顺畅舒适的体验，与现在广为使用的 Python 颇为相近。但 Julia 更多的是以科学与数值计算为目的，原生的并发机制与分布式、云计算特性，简洁人性化的语法，以及媲美于静态语言的性能，所有这些表现都是笔者期待已久的。而今，在日常的数据分析和前期数据处理中，笔者都会首选 Julia 语言。

为了让喜爱的 Julia 语言能够更快普及，进入首选的工业级技术架构，笔者不揣浅陋，捉笔从文，写就此书，以求与各位爱好者共同进步。也期冀 Julia 能成为一个写着简单、读着愉悦、迁移方便、应用广泛、性能强劲的通用编程语言，让我们在开发工作中不再纠结语言的选择。

Julia 的官方文档还算详细，但组织结构并不清晰，概念散乱各处，对初学者并不友好。为此，笔者愿意以此书为契机，将这几年的经验分享给大家，希望能更条理清晰地展

现 Julia 的特色，帮助大家更快、更好地熟悉并掌握 Julia，并在实际开发中获益。

本书的结构

Julia 语言不仅提供了灵活、多样、简洁的语法，更有很多符合实际开发需求的强大特性，也充满了人性化的设计。它不仅支持各种类型的声明定义、贴近于数学概念的计算规则，还在常见的高维数组、字符串处理、国际化支持、元编程等方面提供了强大的支持。尤其是在并行计算、混合编程等方面更是独具特色，原生地提供了良好的机制，使得这方面的编程工作变得极为快捷便利。

为了能够让读者通过本书了解、认识、掌握 Julia 语言的基本概念并能付诸实践，笔者反复对掌握的资料进行了梳理、调整，并且基于真实的运行环境，尽可能地为每个功能点提供相应的示例代码，以求准确、明晰地阐明各个要点。本书主要内容如下：

第 1 章介绍 Julia 语言的基本情况，同时重点介绍 Julia 运行环境的使用方法。

第 2 章对编程语言的基础概念进行了简单的介绍，能够帮助读者在后续的学习中理解 Julia 语言的特点，对于有经验的读者可做选读内容。

第 3 章从包括有理数、复数在内的基本数值系统开始，详细地介绍 Julia 语言的基本语法。

第 4 章基于前一章介绍的各种数值类型介绍 Julia 的各种运算符使用规则。

第 5 章主要介绍经典的判断、循环逻辑，还有 Julia 中较为特别的复合表达式。

第 6 章介绍类型系统，这是 Julia 语言的精髓，包括抽象类型、元类型及复合类型等，都有着 Julia 自己的特点。另外，该章还会重点介绍类型参数化的内容，这也是 Julia 灵活适应各种应用场景的基础。该章介绍的元组、字典、集合等结构也是开发 Julia 程序时常会用到的数集。

第 7 章介绍函数与方法，这不但是 Julia 多态分发机制的基础，也是 Julia 博采众长的精华。

第 8 章介绍 Julia 被称为数值计算语言的核心优势特性——多维数组。数组是科学计算中最为常见的数据结构，但能够以统一的结构表达向量、矩阵、张量甚至高维空间的机制，却是 Julia 的特色。

第 9 章介绍开发中经常遇到的字符串处理方法，包括常见的正则表达式等。

第 10 章使我们能够更深刻地认识 Julia 中“一切皆对象”的理念，因为通过 Symbol 与 Expr 类型的封装，Julia 代码也是对象的一部分。这章介绍的宏，也是在 Julia 开发中极为强大的特性。

第 11 章介绍时间和日期的处理方法。

第 12 章介绍与 IO 相关的内容，包括流、文件操作、网络通信及序列化等。通过该章的学习，我们会再一次为 Julia 的简洁、高效所折服。

第 13 章介绍 Julia 代码的组织方式，包括模块、文件以及包，尤其是对包的管理进行了较为详尽的阐述。

第 14 章介绍 Julia 原生提供的并行计算特性，这也是 Julia 最具魅力的内容之一。在该章中，我们会详尽地阐述协程任务、远程调用及引用，还有数据通道等方面的内容。

第 15 章可以作为选读内容，介绍 Julia 与 C/C++、Python 进行混合编程的基本方法。不过由于运行环境等方面的约束，在学习该章时，如果要通过实例进行实践，建议在 Linux 或 MacOS 下进行。

第 16 章给出了对 Julia 编程方面的经验总结或优化建议。该章也可作为选读内容，不过笔者仍建议所有的读者能认真学习该章内容，并通过实例进行体验，这样才能对 Julia 语言有更为深刻的认识。

第 17 章以机器学习领域中经典的决策树算法为例，展示如何用 Julia 实现该算法的主要过程。在这个实践中，我们对 Julia 各种语法技巧的使用会有更切实的认知。

在本书的结尾，以附录的方式列举了 Julia 中常见的异常类型、系统常量以及字符串操作函数，而且对可能有用的第三方包进行了简单的介绍，希望读者能够通过这方面的内容，了解 Julia 社区的强大力量，习惯性地从社区中获得各种支持。

另外，在本书撰写时，为了简明扼要地将概念阐述清楚，在确保不会影响读者了解语言的核心应用要点的情况下，在内容上进行了适当缩减。如果读者想了解更深入的内容，可以通过官网查阅更多的资料进行学习。

本书适合的读者

本书尽量从基础知识入手，逐步深入地介绍 Julia 语言。但因为 Julia 语言的设计与实现借鉴了众多先进的理念，所以本书难以进行大而全的阐述，省略了不少内容。所以本书不适合没有任何编程经验的读者，读者至少要了解面向对象、泛型编程与函数式等编程概念。

由于本书几乎涉及了 Julia 语言的方方面面，要点颇多，所以建议读者在通过本书学习 Julia 语言时，能够按照其中的实例，多多动手实践，并能在实际的编程工作中选用 Julia 语言，进行一些开发实践。无论哪一种语言，动手实践是掌握这门语言的唯一捷径。

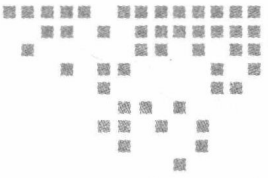
致谢

首先感谢设计与实现 Julia 语言的近 700 位贡献者，为计算机与科学领域提供了一门简洁易用的语言；也感谢近 2000 个第三方库的社区贡献者，让 Julia 语言能够快速普及，焕发出了蓬勃的生机。

此外，感谢上海交通大学副教授潘汉博士在本书校对期间给予的大力支持。

在本书数月的撰写过程中，妻子冯莹霞和家人的支持与照顾让笔者感动不已，有了她们本书才能够有机会顺利完成，与读者们相见。感谢她们给予我的一切！

目　录 Contents



第 1 章 Chapter 1

初识 Julia

Julia 语言[⊖]是一种为高性能数值计算设计的高层次动态编程语言，在分布式并行化、精确数值计算等方面提供了独具特色的支持，并包含大量可扩展的数学函数库。尤其是在线性代数、随机数生成、信号处理、字符串处理等方面，Julia 集成了众多成熟、优秀的基于 C 和 Fortran 开发的开源库，有着很高的性能与效率。另外，Julia 的开发者社区已经非常强大，贡献了大量的第三方库，我们可通过内置的包管理器方便地安装使用。

Julia 语言更多的特点还有：

- 多态分发（Multiple Dispatch）机制，通过不同类型的参数组合，可以定义同名函数不同的行为。
- 动态类型系统：用户自定义的类型可像内置类型一样快速、轻便。
- 简洁又可扩展的数值类型转换与提升机制。
- 高效能的多语言编码环境，支持包括 UTF-8 在内的各种 Unicode 编码[⊜]。
- 原生设计的并行与分布式计算机制。
- 轻量级的“绿色”线程——协程机制。
- 优秀的性能，可以与静态编译的 C 语言媲美。
- Lisp 语言式的宏及元编程（Meta-programming）范式的支持。
- 内置的第三方功能包管理器。
- 可与 Python、R、Matlab 及 Java 等语言进行混合编程。
- 类似于 Shell 的外部程序调用。

⊖ Julia 语言官网为 https://julialang.org，其中包括 Julia 基本介绍、源代码链接、英文说明文档、博客、社区、生态及教程等各种资源。

⊜ Unicode 是计算机领域的一个业界标准，统一为各种自然语言字符设定了唯一的编码，以满足跨语言、跨平台文本转换处理的需求。UTF-8、UTF-16、UTF-32 都是编码方案之一。

❑ 不需要额外的封装层或特别的 API，即可直接调用 C 语言的库函数。

可以说 Julia 在很多方面都独具特色。比如在并行化计算方面，Julia 并没有专门设计特殊的语法结构，而是提供了足够灵活的机制，并可自动进行分布式的部署，能够实现云端操作，使得并行化编程极为便捷。

值得称道的是，Julia 语言基于 MIT 许可证[㊀]是开源、免费的。而且其生态中的各种库与软件也主要采用 GPL、LGPL 或 BSD 等许可。核心代码及各种第三方大部分均托管在 GitHub 这个有名的开源代码管理平台中，用户可以获得源代码，了解语言的各种实现细节，不但能对语言进行更深入的学习，也能够在设计思路方面受益。

使用 Julia 开发有着非常好的体验。不但语法自然简洁，而且结构清晰，效率也非常高。更为可贵的是，其性能也不差。优秀的语言设计结合强大的即时（Just-In-Time，JIT）编译系统 LLVM[㊁]，使得 Julia 的运行性能在很多时候能够媲美 C 语言。在一份官方提供的 Benchmark 中，相比于 C、Fortran、Python、Matlab/Octave、R、JavaScript、Java、Lua 与 Mathematica 等其他语言，Julia 在性能方面有着非常卓越的表现。[㊂]

Julia 语言 v0.1 版发布于 2013 年 2 月 14 日，自 2017 年 12 月 14 日发布 v0.6.2 版本后，代码更新很快，2018 年 8 月 8 日发布了 v0.7 版后，当年 8 月 9 日便发布了 v1.0 的正式版本。经过数次的版本迭代，Julia 语言已经日趋成熟，并已经科学计算领域崭露头角。完全可以相信，在不久的将来，在众多的编程语言中，尤其是在科学计算领域方面，Julia 必能占有一席之地。

1.1 有用的资源

Julia 语言的设计者们显然是聪明的一群人，但笔者认为他们更具有符合时代的智慧。他们在设计实现这门包罗万象又简洁高效的语言时，便建立了开放的包管理机制，从而能够借助强大的开源社区，让 Julia 以前所未有的速度发展与普及。

截至本书成稿时，官方注册的包已近 2000 个，已经成为 Julia 生态系统的重要组成部分。笔者相信，随着 Julia 的快速发展，第三方包的规模与质量也会不断地提升。

Julia 的贡献者来自世界各地，提供了大量各自领域有针对性的包，在这些包中要找到我们想要的支持并不困难。为了更好地管理这些包，让这些包更好地服务于各种应用场景，Julia 将这些包分成多个频道，列举如下：

㊀ MIT 许可证（The MIT License）是开源软件授权中被广泛使用的一种，比 GPL、LGPL、BSD 等更为宽松：被授权人有权使用、复制、修改、合并、出版、传播、再授权及贩售软件或其副本，也可适当修改授权条款，但需在发布版本中包含该版权许可声明。

㊁ 底层虚拟机（Low Level Virtual Machine，LLVM），是编译器的基础建设之一，利用虚拟技术实现编译期、链接期、运行期及“空闲期”的优化，是一系列技术工具链的集合，通常作为某类语言的编译器的后台。LLVM 项目由伊利诺伊大学的维克拉姆·艾夫（Vikram Adve）和克里斯·拉特纳（Chris Lattner）于 2000 年发起。早期以 C/C++ 为主要对象，后来开始支持 Objective-C，并扩展到其他更多的编程语言。

㊂ 有兴趣的读者可以从 GitHub 中的官网地址下载代码运行看看，网址为 https://github.com/JuliaLang/Microbench marks.git。

通用类：

- JuliaDocs——Julia 文档系统相关的包。
- Julia-i18n——国际化（i18n）与本地化（L10n）支持。
- JuliaTime——日期与时间相关的库。
- JuliaPraxis——最佳实践案例与支持。
- JuliaEditorSupport——文本编辑器与 IDE 的扩展及插件。
- Juno——基于 Atom 编辑器的 Juno IDE。

基础计算：

- JuliaArrays——自定义的数组类型及相关工具。
- JuliaBerry——Raspberry Pi[⊖]相关的资源与支持组件。
- JuliaCI——用于 Julia 包的持续集成工具。
- JuliaGPU——GPU 计算支持。
- JuliaInterop——与其他语言进行混合编程的相关支持包。
- JuliaIO——包括序列化、通信协议及文件格式等 IO 相关的包。
- JuliaParallel——并行与分布式计算支持。
- JuliaWeb——Web 技术栈

数学：

- JuliaDiff——微分数值计算。
- JuliaDiffEq——微分方程求解与分析。
- JuliaGeometry——计算几何。
- JuliaGraphs——图理论与实现。
- JuliaIntervals——计算机精准算术支持。
- JuliaMath——包括积分、傅里叶变换、插值等在内的数学包。
- JuliaOpt——最优化。
- JuliaPolyhedra——多面体计算（polyhedral computation）。
- JuliaSparse——稀疏矩阵求解等支持。

科学：

- BioJulia——生物学。
- EcoJulia——生态学。
- JuliaAstro——天文学。
- JuliaDSP——数字信号处理。
- JuliaQuant——金融。
- JuliaQuantum——量子科学与技术。

⊖ 树莓派基金会（Raspberry Pi）是英国一个小型的慈善组织，旨在推广科技，且不以技术销售营利为目的。同名的 Raspberry Pi 则是该基金会提供的一款针对电脑业余爱好者、教师、小学生以及小型企业等用户的迷你电脑，预装 Linux 系统，体积仅信用卡大小，搭载 ARM 架构处理器，运算性能和智能手机相仿。

- JuliaPhysics——物理学。
- JuliaDynamics——线性及非线性动态系统、混沌等。

数据：

- JuliaML——机器学习。
- JuliaStats——数理与统计。
- JuliaImages——图像处理。
- JuliaText——自然语言处理、计算语言学及信息检索。
- JuliaDatabases——数据库及数据仓库驱动支持。
- JuliaData——数据操纵、存取及 IO 相关。

可视化：

- GiovineItalia——图表支持。
- JuliaPlots——数据可视化。
- JuliaGL——OpenGL API 及其生态。
- JuliaGraphics——绘图、色彩及 GUI 相关支持。

在官网中，我们可以通过这些频道快速地找到感兴趣的内容。另外，Julia 社区提供了论坛、年会等线下活动，也提供 Twitter、新闻组等线上方式，帮助开发者参与并了解 Julia 语言，并能够促进参与者的交流互动。而且在 YouTube 等平台提供了各种教程，帮助新学者熟悉 Julia 语言。

官方提供的开发文档、GitHub 中开源的代码，还有耗费心思撰写的本书，都可以成为我们学习这门语言的起点。

1.2 环境准备

Julia 语言是跨平台的，主要支持 Windows、Linux 及 MacOS 三类操作系统，不但能够在 i686 及 x86_64 架构上运行，而且还支持 ARM、PowerPC 等平台。所支持的环境详细如表 1-1 所示。

表 1-1 Julia 支持的环境

操作系统	架　构	持续集成 (CI)	官方二进制安装程序	支持层面
Linux 2.6.18+	x86-64（64 位）	✓	✓	官方
	i686（32 位）	✓	✓	官方
	ARM v7（32 位）		✓	官方
	ARM v8（64 位）		✓	官方
	PowerPC（64 位）			社区
	PTX（64 位）	✓		第三方
macOS 10.8+	x86-64（64 位）	✓	✓	官方

（续）

操作系统	架　构	持续集成 (CI)	官方二进制安装程序	支持层面
Windows 7+	x86-64（64 位）	✓	✓	官方
	i686（32 位）	✓	✓	官方
FreeBSD 11.0+	x86-64（64 位）	✓		社区

我们可以从其官方网站中直接下载编译好的进制安装包，也可以下载源代码在本机的环境中重新编译。不过建议使用提供的可执行程序安装 Julia 程序环境，除非非常熟悉 C++ 代码的编译过程。

1.2.1　二进制包安装

Julia 官方为 Windows、MacOS、Linux 及 FreeBSD 几个最常见的平台提供了已经编译好的可直接执行的安装程序，可通过官方页面 https://julialang.org/downloads 进行下载。安装文件并不大，30～80MB，不像微软各种东西，动辄就上 GB。二进制的安装方式能让我们不用操心各种依赖与复杂的编译过程，安装完毕即可轻松开始使用。

1. Windows

Julia 支持 Windows 7/ Windows Server 2012 及更新的操作系统，所以最好不要在 Windows 2000 或 Windows XP 这些太老的系统中尝试，不确定会发生什么问题。32 位版本的 Julia 可运行在 32 位（x86）或 64 位（x86_64）的操作系统中，但 64 位版本只能运行在 64 位的 Windows 系统中。

在 Windows 上安装 Julia 极为简单，将 exe 安装包下载后，双击文件然后按照引导一步步执行既可。需要说明的是，对于 Windows 7 或 Windows Server 2012 系统，需要额外进行以下两个工作：

1）更新操作系统组件，以支持 1.1 及 1.2 版的传输层安全协议（Transport Layer Security，TLS）。这是因为 GitHub 已经停止了 TLS 1.0 的使用，所以需要操作系统支持更高的 TLS 版本，Julia 的包管理器才能正常工作。

2）安装 Windows Management Framework（WMF）3.0 版或者更高版本。

若要卸载 Julia，只需直接将 Julia 的安装目录以及工作目录（一般为 `%HOME%/.julia` 路径）直接删除，并将 `%HOME%/.juliarc.jl` 和 `%HOME%/.julia_history` 两个文件同时删除即可。

2. MacOS

Julia 支持 MacOS 10.8 及更新的发行版本，提供的二进制安装包是一个扩展名为 dmg 的可执行程序，包含了 `Julia.app` 目录，可以直接打开执行。

除了二进制安装包，MacOS 还可以使用 HomeBrew 安装 Julia 环境，执行语句如下：

```
$ brew update
```

```
$ brew tap staticfloat/julia
$ brew install julia
```

或者采用 cask 的方式，即：

```
$ brew cask install julia
```

在安装完成后，在系统终端中执行

```
$ julia
```

即可启动 Julia 环境。

如果要卸载 Julia，只需移除 Julia.app 目录及 ~/.julia 工作目录，如果不再需要配置文件，可将 ~/.juliarc.jl 文件同时删除即可。

3. Linux 及 FreeBSD

除了 FreeBSD 外，Julia 还支持多种 Linux 发行版本，包括 Fedora、RHEL、CentOS、Scientific Linux、Oracle Enterprise Linux、Ubuntu、Debian、openSUSE 及 Arch Linux 等。

下载官方通用的 tar.gz 格式二进制包后，安装并没有特别的执行步骤，解压后便可执行。为了后续的方便，最好是将执行目录添加到系统环境变量中，简单的办法是在 /usr/local/bin 或 /usr/bin 目录中建立安装目录中 bin/julia 可执行文件的软链接（Symbolic Link），语句一般为：

```
$ sudo ln -s <where you extracted the julia archive>/bin/julia /usr/local/bin/julia
```

此后便可无须切换到安装目录或输入完整的路径来执行 Julia 主程序了。

实际上，对于 Linux 及 FreeBSD 来说，通过其中的包管理命令安装 Julia 可能更为方便，而且也能够更便捷地对 Julia 进行更新。

对于使用 RHEL、CentOS、Scientific Linux 及 Oracle Enterprise Linux（版本 5 或更高）的用户来说，需要开启操作系统的 EPEL（Extra Packages for Enterprise Linux）支持，然后便可与 Fedora（版本 19 或更高）一样，执行下述安装命令：

```
$ sudo dnf copr enable nalimilan/julia
$ sudo yum install julia
```

若使用的是 CentOS 7 或更高版本，也可直接执行：

```
$ sudo yum-config-manager --add-repo https://copr.fedorainfracloud.org/coprs/
   nalimilan/julia/repo/epel-7 /nalimilan-julia-epel-7.repo
$ sudo yum install julia
```

若发行版中没有 dnf 与 yum-config-manager，需在 Copr 网站中下载相关的 .repo 文件，复制到系统的 /etc/yum.repos 源管理目录，再次尝试执行安装过程。

每日构建（Nightly Building）是一些系统或程序通常采用的更新方式，如果需要及时跟踪 Julia 的最新功能，可以通过 yum 对已经安装的 Julia 程序进行更新。首先，执行：

```
$ sudo dnf copr enable nalimilan/julia-nightlies
```

以开启每日更新版本库，再执行以下命令时：

```
$ sudo yum install julia
```