Continue



```
Ruby 3.4.4 已发布。 阅读全文... 由 k0kubun 发表于 2025-05-14 在 net-imap gem 中存在一个可能造成 DoS 的漏洞。此漏洞的 CVE 编号为 CVE-2025-43857。我们建议您更新 net-imap gem。 阅读全文... 由 nevans 发表于 2025-04-28 我们很高兴地宣布 Ruby 3.5.0-preview1 已发布。Ruby 3.5 除了将 Unicode 版本更新到 15.1.0外,还有一些其他变更。 阅读全文... 由 naruse 发表于
2025-04-18 Ruby 3.4.3 已发布。 阅读全文... 由 k0kubun 发表于 2025-04-14 Posted by k0kubun on 3 Sep 2024 Ruby 3.3.5 has been released. This is a routine update that includes minor bug fixes. We recommend upgrading your Ruby version at your earliest convenience. For more details, please refer to the GitHub release notes. Release Schedule As
 previously announced, we intend to release the latest stable Ruby version (currently Ruby 3.3) every 2 months following a .1 release. We expect to release Ruby 3.3.6 on November 5th and Ruby 3.3.7 on January 7th. If any significant changes arise that impact a large number of users, we may release a new version earlier than scheduled. Download
 SIZE: 22129139 SHA1: 59444476bbe9e789fc777d8fb4d4456bc057604f SHA256: 3781a3504222c2f26cb4b9eb9c1a12dbf4944d366ce24a9ff8cf99ecbce75196 SHA512: 5c482059628ef9de5d8a6ad4751f8043f2fc2b159b768265be7f3ee0574ad51d9500ee4fc9146c5978fbd51313039c3de39e7b7a4dedc9bcd5d09a41a713f1a7 SIZE: 16403660 SHA1:
692 bc 3188 bd b9e c 30 b8672543961 b0 11 d699590 a SHA 256: 51 aec 7 ea 89 b46125 a 2c 9 adc 6f 36766 b65023 d47952 b916b 1 aed 300 ddc c 042359 SHA 512: dd 5c 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 6e 05 a 57 b7 af 28 fd 921 6d d38d 714580 8188 bb 3695 a 7f 3a 4e da 3883 SIZE: 270 20 194 SHA 1: dd 5c 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 6e 05 a 57 b7 af 28 fd 921 6d d38d 714580 8188 bb 3695 a 7f 3a 4e da 3883 SIZE: 270 20 194 SHA 1: dd 5c 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 6e 05 a 57 b7 af 28 fd 921 6d d38d 714580 8188 bb 3695 a 7f 3a 4e da 3883 SIZE: 270 20 194 SHA 1: dd 5c 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 fc 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 fc 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 fc 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 fc 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 fc 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 fc 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 108345 dbf 4b 21f 61 ad 978 ad 680 fc 6a 7f 74854 e 143 e 0 ca 46 b9d 7c 0d 1983 fc 4886f 5f 733 cd 1983 fc 4886f 
ccb32746aef491ce05be42218301e2c47185e5fc SHA256: d3c13e124707494935d00ebc5c7983b0252bc13de49223fd31104ba5467a057a SHA512: bf83af835a74283aff21042538ee1f1eb70ff12dac1edd4672d787547cd29cb7b69a9299682f89c8499eb610737b10a7fc03eca038574cb4ba565205d96b0016 Release Comment Many committers, developers,
and users who provided bug reports helped us make this release. Thanks for their contributions. Guides, tutorials, and reference material to help you learn more about Ruby Installing Ruby. Official Ruby Documentation docs.ruby-
lang.org/en: List of documentation for Ruby 3.4. docs.ruby-lang.org/en/master: Documentation for Ruby 5.4. docs.ruby-lang.org/en/master: Documentation for Ruby 6.4. docs.ruby-lang.org/en/master: Documentation for Ruby 7.4. docs.ruby-lang.org/en/mas
Program A wonderful little tutorial by Chris Pine for programming newbies. If you don't know how to program, start here. Ruby in Twenty Minutes to complete. The Odin Project An open source full stack curriculum Exercises with automatic analysis and personal mentoring.
Codecademy Online code bootcamp with a variety of topics. Manuals / Books Beginner Programming Ruby 3.3. The Well-Grounded Rubyist A tutorial that begins with your first Ruby program and takes you all the way to sophisticated topics like reflection, threading, and recursion.
Intermediate Practical OOD in Ruby (POODR) A programmer's tale about how to write object-oriented code. Expert Metaprogramming Explains metaprogramming in a down-to-earth style. Ruby Under a Microscope (RUM) An illustrated guide to Ruby internals. Community Documentation These documentation sites are maintained by the Ruby
community. RubyDoc.info The one-stop web site for reference documentation about Ruby gems and GitHub-hosted Ruby projects. RubyAPI.org Easily find and browse Ruby classes, modules, and methods. ruby-doc.org Online API documentation DevDocs.io Online API documentation Ruby QuickRef The Ruby quick reference rubyreferences A full
 language reference + detailed language changelog. Style Guides rubystyle.guide RuboCop's Ruby style guide RuboCop Automated enforcement of their style guide w3resource W3's Ruby style guide IRB The interactive Ruby Read-Eval-Print-Loop
(REPL) Pry An alternative Ruby REPL Rake A make-like build utility for Ruby. RI (Ruby Information) is the Ruby command-line utility that gives fast and easy on-line access to Ruby documentation. RBS Type Signature for Ruby type-Prof An experimental type-level Ruby interpreter for testing and understanding Ruby code. Steep Static type checker for
Ruby. Editors and IDEs For coding in Ruby, you can use the default editor of your operating system. By the way, to be more effective in coding, it is worth choosing a source code editor with basic Ruby support (e.g. syntax-highlighting, file browsing) or an integrated development environment with advanced features (e.g. code completion, refactoring,
testing support). Here is a list of popular editors used by Rubyists, broken up by learning curve: All of these editors support the Language Server Protocol (LSP), either by default or through their LSP plugins. Shopify's ruby-lsp is one of the most popular language servers for Ruby and supports all of the above editors. Older Reading / Resources These
links were more prominent but haven't been updated in ages. Ruby Koans The Koans walk you along the path to enlightenment in order to learn Ruby. The goal is to learn Ruby Essentials A free on-line book designed to provide a concise and
easy to follow guide to learning Ruby. Why's (Poignant) Guide to Ruby An unconventional but interesting book that will teach you Ruby through stories, wit, and comics. Originally created by why the lucky stiff, this guide remains a classic for Ruby learners. Learn Ruby the Hard Way A very good set of exercises with explanations that guide you from
the absolute basics of Ruby all the way to OOP and web development. Programming Ruby The seminal work on Ruby in English, this first edition of the Pragmatic Programming Wikibook A free online manual with beginner and intermediate content plus a thorough language reference. Ruby
3.4.4 has been released. Continue Reading... Posted by k0kubun on 14 May 2025 There is a possibility for DoS by in the net-imap gem. Continue Reading... Posted by nevans on 28 Apr 2025 We are pleased to announce the release of
Ruby 3.5.0-preview1. Ruby 3.5 updates its Unicode version to 15.1.0, and so on. Continue Reading... Posted by k0kubun on 14 Apr 2025 Welcome to the official Ruby programming language documentation. String - Text manipulation and string utilities. Symbol
 Named identifiers inside the Ruby interpreter. Array - Ordered collections of objects. Hash - Key-value pairs for efficient data retrieval. Integer - Integer number class. Float - Floating-point number class. Float - Floating-point number class. Enumerable - Collection traversal and searching. File - File operations and handling. IO - Input/output functionality. Time - Time representation
 Regexp - Regular expressions for pattern matching. Range - Representing a range of values. Exception - Base class for all exceptions. Thread - Multithreading and concurrency. There are some standard libraries included in Ruby that are also commonly used, such as: With package managers or third-party tools, you have plenty of options to install and
manage Ruby. You may already have Ruby installed on your computer. You can check inside a terminal emulator by typing: ruby -v This should output some information on the installed Ruby version. Choose Your Installation Method There are several ways to install Ruby: On a UNIX-like operating system, using your system's package manager is
easiest. However, the packaged Ruby version may not be the newest one. Installers can be used to install a specific or multiple Ruby versions. There is also an installer for Windows 10, you can also use the Windows
Subsystem for Linux to install one of the supported Linux distributions and use any of the installation methods: Package Management Systems If you cannot compile your own Ruby, and you do not want to use a third-party tool, you can use your system's package manager to install
Ruby. Some members of the Ruby community feel that you should avoid package managers will install older Ruby versions instead of the latest release. To use the latest Ruby release, check that the package managers will install older Ruby version number. Or
use a dedicated installer. apt (Debian or Ubuntu) Debian GNU/Linux and Ubuntu use the apt package manager. You can use it like this: $ sudo apt-get install ruby-full yum (CentOS, Fedora, or RHEL) CentOS, Fedora, and RHEL use the yum package manager. You can use it like this: $ sudo yum install ruby The installed version is typically the latest
version of Ruby available at the release time of the specific distribution version. snap (Ubuntu or other Linux distributions) Snap is a package manager developed by Canonical. It is available out-of-the-box on Ubuntu, but snap also works on many other Linux distributions. You can use it like this: $ sudo snap install ruby --classic We have several
channels per Ruby minor series. For instance, the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the portage (Gentoo) Gentoo uses the portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following commands switch to Ruby 2.3: $ sudo snap refresh portage (Gentoo) Gentoo uses the following command
website for details. pacman (Arch Linux) Arch Linux uses a package manager named pacman. To get Ruby, just do this: $ sudo pacman -S ruby This should install the latest stable Ruby version. Homebrew (macOS) Ruby versions 2.0 and above are included by default in macOS releases since at least El Capitan (10.11). Homebrew is a commonly used
package manager on macOS. Installing Ruby using Homebrew is easy: $ brew install ruby This should install ruby Version. FreeBSD freeBS
 Ports Collection. This is useful if you want to customize the build configuration options. More information about Ruby and its surrounding ecosystem on FreeBSD can be found on the FreeBSD can be fou
to see the available versions and to install one: $ doas pkg_add ruby You can install multiple major versions side by side, because their binaries have different names (e.g. ruby27, ruby26). The HEAD branch of the OpenBSD ports collection might have the most recent version of Ruby for this platform some days after it is released, see directory
lang/ruby in the most recent ports collection. Ruby on OpenIndiana To install Ruby on OpenIndiana, please use the Image Packaging System (IPS) client. This will install runtime/ruby However, the third-party tools might be a good way to obtain the
latest version of Ruby. Windows Package Manager On Windows, you can use the Windows, you can use
ruby for projects, you may want to install RubyWithDevKit > winget install RubyWithDevKit > wi
Other Distributions On other systems, you can search the package repository of your Linux distribution's manager for Ruby, you can use a third-party installer. Some installers allow you to
install multiple versions on the same system; associated managers can help to switch between the different Rubies. If you are planning to use RVM as a version manager you don't need a separate installer, it comes with its own. ruby-build can
also be used as a standalone program without rbenv. It is available for macOS, Linux, and other UNIX-like operating systems. ruby-install allows you to compile and install different versions of Ruby into arbitrary directories. chruby is a compile and install different versions. It is available for macOS, Linux, and other
UNIX-like operating systems. RubyInstaller On Windows, RubyInstaller Gives you everything you need to set up a full Ruby development environment. Just download it, run it, and you are done! Ruby Stack provides a complete development
environment for Rails. It supports macOS, Linux, Windows, virtual machines, and cloud images. Managers to manage multiple Rubies. They allow easy or even automatic switching between Ruby versions depending on the project and other advantages but are not officially supported. You can however find support
 within their respective communities. asdf-vm is an extendable version manager that can manage multiple language runtime versions on a per-project basis. You will need the asdf-ruby plugin (which in turn uses ruby-build) to install Ruby. chruby allows you to switch between multiple Rubies. It can manage Rubies installed by ruby-
install or even built from source. mise-en-place allows you to switch between multiple Rubies without requiring additional tools. It manages installations automatically and includes a gem backend to manage versions of CLIs written in Ruby. It supports UNIX-like and Windows operating systems. rbenv rbenv allows you to manage versions of CLIs written in Ruby.
multiple installations of Ruby. While it can't install Ruby by default, its ruby-build plugin can. Both tools are available for macOS, Linux, or other UNIX-like operating systems. rbenv for Windows rbenv for Windows rbenv for Windows allows you to install and manage multiple installations of Ruby on Windows. It's written in PowerShell thus providing a native way to use
Ruby for Windows users. Besides, the command line interface is compatible with rbenv on UNIX-like systems. RVM ("Ruby Version Manager") RVM allows you to install and manage multiple installations of Ruby on your system. It can also manage different gemsets. It is available for macOS, Linux, or other UNIX-like operating systems. RVM 4
Windows RVM 4 Windows allows you to install and manage multiple installations of Ruby on Windows. It is a clone of the original RVM and supports the classic command line as well as Powershell by providing the same command line installations of Ruby on Windows. It is a clone of the original RVM and supports the classic command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as well as Powershell by providing the same command line as 
multiple Rubies on macOS, Linux, or Windows systems. Building from Source Of course, you can install Ruby from source. Download and unpack a tarball, then just do this: $./configure $ make $ sudo make install By default, this will install Ruby into /usr/local. To change, pass the --prefix=DIR option to the ./configure $ make $ sudo make install By default, this will install Ruby into /usr/local. To change, pass the --prefix=DIR option to the ./configure $ make $ sudo make install By default, this will install Ruby into /usr/local. To change, pass the --prefix=DIR option to the ./configure $ make $ sudo make install By default, this will install Ruby into /usr/local. To change, pass the --prefix=DIR option to the ./configure $ make $ sudo make install By default, this will be a subject to the ./configure $ make $ sudo make install By default, this will be a subject to the ./configure $ make $ sudo make install By default, this will be a subject to the ./configure $ make $ sudo make install By default, this will be a subject to the ./configure $ make $ sudo make install By default, this will be a subject to the ./configure $ make $ sudo make install By default By
information about building from source in the Building Ruby instructions. Using the third-party tools or package managers might be a better idea, though, because the installed Ruby won't be managed by any tools. Wondering why Ruby is so popular? Its fans call it a beautiful, artful language. And yet, they say it's handy and practical. What gives?
The Ideals of Ruby's Creator Ruby is a language of careful balance. Its creator, Yukihiro "Matz" Matsumoto, blended parts of his favorite language that balanced functional programming with imperative programming. He has often said that he is "trying to make Ruby natural, not simple,"
in a way that mirrors life. Building on this, he adds: Ruby is simple in appearance, but is very complex inside, just like our human body1. About Ruby's Growth Since its public release in 1995, Ruby has drawn devoted coders worldwide. In 2006, Ruby achieved mass acceptance. With active user groups formed in the world's major cities and Ruby-
related conferences filled to capacity. Ruby-Talk, the primary mailing list for discussion of the Ruby language, climbed to an average of 200 messages per day in 2006. It has dropped in recent years as the size of the community pushed discussion from one central list into many smaller groups. Ruby is ranked among the top 10 on most of the indices
that measure the growth and popularity of programming languages worldwide (such as the TIOBE index). Much of the growth is attributed to the popularity of software written in Ruby, particularly the Ruby on Rails web framework. Ruby is also completely free. Not only free of charge, but also free to use, copy, modify, and distribute. Seeing
Everything as an Object Initially, Matz looked at other languages to find an ideal syntax. Recalling his search, he said, "I wanted a scripting language that was more powerful than Perl, and more object-oriented than Python2." In Ruby, everything is an object. Every bit of information and code can be given their own properties and actions. Object-oriented than Python2."
oriented programming calls properties by the name instance variables and actions are known as methods. Ruby's pure object-oriented approach is most commonly demonstrated by a bit of code which applies an action to a number. 5.times { print "We *love* Ruby -- it's outrageous!" } In many languages, numbers and other primitive types are not
objects. Ruby follows the influence of the Smalltalk language by giving methods and instance variables to all of Ruby, since rules applying to objects apply to all of Ruby, since rules applying to objects apply to all of Ruby, since rules applying to objects apply to all of Ruby, since rules applying to objects.
redefined, at will. Existing parts can be added upon. Ruby tries not to restrict the coder. For example, addition is performed with the plus (+) operator. But, if you'd rather use the readable word plus, you could add such a method to Ruby's builtin Numeric class. Class Numeric def plus(x) self.+(x) end end y = 5.plus 6 # y is now equal to 11 Ruby's
operators are syntactic sugar for methods. You can redefine them as well. Blocks: a Truly Expressive Feature Ruby's block are also seen as a source of great flexibility. A programmer can attach a closure to any method, describing how that method should act. The closure is called a block and has become one of the most popular features for
newcomers to Ruby from other imperative languages like PHP or Visual Basic. Blocks are inspired by functional languages. Matz said, "in Ruby closures, I wanted to respect the Lisp culture3." search engines = %w[Google Yahoo MSN].map do |engine| "." + engine.downcase + ".com" end In the above code, the block is described inside the do ... end
construct. The map method applies the block to the provided list of words. Many other methods in Ruby leave a hole open for a coder to write their own block to fill in the details of what that method should do. Ruby and the Mixin Unlike many object-oriented languages, Ruby features single inheritance only, on purpose. But Ruby knows the concept of
modules (called Categories in Objective-C). Modules are collections of methods. Classes can mixin a module and receive all its methods for free. For example, any class WyArray include Enumerable end Generally,
Rubyists see this as a much clearer way than multiple inheritance, which is complex and can be too restrictive. Ruby's Visual Appearance While Ruby often uses very limited punctuation and usually prefers English keywords, some punctuation is used to decorate Ruby. Ruby needs no variable declarations. It uses simple naming conventions to denote
the scope of variables. var could be a local variable. @var is an instance variable. $\text{Properties of each variable}$ enhance readability by allowing the programmer to easily identify the roles of each variable. $\text{Properties of each variable}$ enhance readability by allowing the programmer to easily identify the roles of each variable.
 features, among which are the following: Ruby has exception handling features, like Java or Python, to make it easy to handle errors. Ruby features a true mark-and-sweep garbage collector for all Ruby objects. No need to maintain reference counts in extension libraries. As Matz says, "This is better for your health." Writing C extensions in Ruby is
 easier than in Perl or Python, with a very elegant API for calling Ruby from C. This includes calls for embedding Ruby in software, for use as a scripting language. A SWIG interface is also available. Ruby can load extension libraries dynamically if an OS allows. Ruby features OS independent threading. Thus, for all platforms on which Ruby runs, you
also have multithreading, regardless of if the OS supports it or not, even on MS-DOS! Ruby is highly portable: it is developed mostly on GNU/Linux, but works on many types of UNIX, macOS, Windows, DOS, BeOS, OS/2, etc. Other Implementations of Ruby Ruby, as a language, has a few different implementations. This page has been discussing the
reference implementation, in the community often referred to as MRI ("Matz's Ruby Interpreter") or CRuby (since it is written in C), but there are also others. They are often useful in certain situations, provide extra integration to other languages or environments, or have special features that MRI doesn't. Here's a list: JRuby is Ruby atop the JVM
(Java Virtual Machine), utilizing the JVM's optimizing JIT compilers, garbage collectors, concurrent threads, tool ecosystem, and vast collection of libraries. Rubinius is 'Ruby written in Ruby'. Built on top of, too. TruffleRuby is a high performance Ruby
implementation on top of GraalVM. mruby is a lightweight implementation of the Ruby language that can be linked and embedded within an application. Its development is led by Ruby's creator Yukihiro "Matz" Matsumoto. IronRuby is an implementation "tightly integrated with the .NET Framework". MagLev is "a fast, stable. Ruby implementation
with integrated object persistence and distributed shared cache". Cardinal is a "Ruby compiler for Parrot Virtual Machine" (Perl 6). For a more complete list, see Awesome Rubies. References 1 Matz, in Blocks and
Closures in Ruby, December 22nd, 2003. This is a small Ruby tutorial that should take no more than 20 minutes to complete. It makes the assumption that you already have Ruby on your computer install it before you get started.) Interactive Ruby tutorial that should take no more than 20 minutes to complete. It makes the assumption that you already have Ruby on your computer install it before you get started.)
statements you feed it. Playing with Ruby code in interactive sessions like this is a terrific way to learn the language. Open up Terminal and type irb, then hit enter. If you're using Mindows, open Interactive Ruby. If you're using macOS open up Terminal and type irb, then hit enter. If you're using Mindows, open Interactive Ruby.
from the Ruby section of your Start Menu. irb(main):001:0> Ok, so it's open. Now what? Type this: "Hello World" irb(main):001:0> "Hello World" program? Not exactly. The second line is just IRB's way of telling us the result of the last
expression it evaluated. If we want to print out "Hello World" we need a bit more: irb(main):002:0> puts "Hello World" Hello World" we need a bit more: irb(main):002:0> puts "Hello World" Hello World" Hello World" Hello World => nil puts is the expression. puts always returns nil, which is Ruby's absolutely-positively-nothing value. Your Free
Calculator is Here Already, we have enough to use IRB as a basic calculator: irb(main):003:0> 3+2 => 5 Three plus two. Easy enough, but you may also be able to go up and change what you just entered. Try hitting the up-arrow on your keyboard and see if it brings up the line with
3+2 on it. If it does, you can use the left arrow key to move just after the + sign and then use backspace to change it to a * sign. irb(main):004:0> 3*2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3**2 = > 6 Next, let's try three squared: irb(main):005:0> 3*
irb(main):006:0> Math.sqrt(9) => 3.0 Ok, wait, what was that last one? If you guessed, "it was figuring out the square root of nine," you're right. But let's take a closer look at things. First of all, what's Math? Modules Group Code by Topic Math is a built-in module for mathematics. Modules serve two roles in Ruby. This shows one role: grouping
similar methods together under a familiar name. Math also contains methods like sin() and tan(). Next is a dot. What does the dot do? The dot is how you identify the receiver of a message? In this case it's sqrt(9), which means call the method sqrt, shorthand for "square root" with the parameter of 9. The result of this method call
is the value 3.0. You might notice it's not just 3. That's because most of the time the square root of a number won't be an integer, so the method always returns a floating-point number. What if we want to remember the result to a variable. irb(main):007:0> a = 3 ** 2 => 9 irb(main):008:0> b = 4 ** 2 => 16
irb(main):009:0> Math.sqrt(a+b) => 5.0 As great as this is for a calculator, we're getting away from the traditional Hello World message that beginning tutorials are supposed to focus on... so let's go back to that. Page 2 What if we want to say "Hello" a lot without getting our fingers all tired? We need to define a method! irb(main):010:0> def hi
irb(main):011:1> puts "Hello World!" irb(main):012:1> end => :hi The code def hi starts the definition of the method, that its name is hi. The next line end tells Ruby we're done defining the method. Ruby's
response => :hi tells us that it knows we're done defining the method. This response could be => nil for Ruby 2.0 and earlier versions. But, it's not important here, so let's go on. The Brief, Repetitive Lives of a Method Now let's try running that method a few times: irb(main):013:0> hi Hello World! => nil irb(main):014:0> hi() Hello World! => nil
Well, that was easy. Calling a method in Ruby is as easy as just mentioning its name to Ruby. If the method doesn't take parameters that's all you need. You can add empty parentheses if you'd like, but they're not needed. What if we want to say hello to one person, and not the whole world? Just redefine hi to take a name as a parameter.
irb(main):015:0> def hi(name) irb(main):016:1> puts "Hello #{name}!" irb(main):016:1> end => :hi irb(main):016:1> puts "Hello Matz! => nil So it works... but let's take a second to see what's going on here. Holding Spots in a String What's the #{name} bit? That's Ruby's way of inserting something into a string. The bit between the braces is
turned into a string (if it isn't one already) and then substituted into the outer string at that point. You can also use this to make sure that someone's name is properly capitalized: irb(main):021:1> end => :hi irb(main):021:1> end => :hi irb(main):022:0> hi "chris" Hello Chris! => nil
irb(main):023:0> hi Hello World! => nil A couple of other tricks to spot here. One is that we're calling the method without parentheses again. If it's obvious what you're doing, the parentheses are optional. The other trick is the default parameter World. What this is saying is "If the name isn't supplied, use the default name of "World"". Evolving Into a
Greeter What if we want a real greeter around, one that remembers your name and welcomes you and treats you always with respect. You might want to use an object for that. Let's create a "Greeter" class. irb(main):025:1> def initialize(name = "World") irb(main):026:2> @name = name irb(main):027:2> end
irb(main):028:1> def say hi irb(main):032:2> puts "Hi #{@name}!" irb(main):033:2> end irb(mai
notice @name. This is an instance variable, and is available to all the methods of the class. As you can see it's used by say hi and say bye. So how do we get this Greeter class set in motion? Create an object. net-imap gem に DoS の脆弱性は CVE-2025-43857 として登録されています。net-imap gem のアップグレードを推奨します。
- もっと読む... Posted by nevans on 28 Apr 2025 Ruby 3.5.0-preview1 が公開されました。Ruby 3.5では、Unicodeバージョンの15.1.0へのアップデートなど様々な改善が行われています。 もっと読む... Posted by naruse on 18 Apr 2025 Ruby 3.4.3がリリースされました。 もっと読む... Posted by k0kubun on 14 Apr 2025 Ruby 3.3.8 がリリースされました。 もっと読む...
Posted by nagachika on 9 Apr 2025 Ruby is a programming language from Japan which is revolutionizing software development. The beauty of Ruby is found in its balance between simplicity and power. You can type some Ruby code in the editor copy - Copies the example code
to the editor Next → Allows you to go to the next lesson Back → Allows you to return to the previous lesson Click on Next to start learning.
```