

Shay Friedman

IronRuby

UNLEASHED



SAMS

IronRuby Unleashed

Copyright © 2010 by Pearson Education, Inc.

All rights reserved. No part of this book shall be reproduced, stored in a retrieval system, or transmitted by any means, electronic, mechanical, photocopying, recording, or otherwise, without written permission from the publisher. No patent liability is assumed with respect to the use of the information contained herein. Although every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions. Nor is any liability assumed for damages resulting from the use of the information contained herein.

ISBN-13: 978-0-672-33078-0

ISBN-10: 0-672-33078-4

Library of Congress Cataloging-in-Publication Data

Friedman, Shay.

IronRuby unleashed / Shay Friedman.

p. cm.

ISBN 978-0-672-33078-0

1. IronRuby (Computer program language) 2. Microsoft .NET Framework. 3. Ruby (Computer program language) I. Title.

QA76.73.I586F74 2010

006.7'882—dc22

2009050114

Printed in the United States of America

First Printing: February 2010

Trademarks

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Pearson Education, Inc. cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

Warning and Disclaimer

Every effort has been made to make this book as complete and as accurate as possible, but no warranty or fitness is implied. The information provided is on an “as is” basis. The author and the publisher shall have neither liability nor responsibility to any person or entity with respect to any loss or damages arising from the information contained in this book.

Bulk Sales

Pearson offers excellent discounts on this book when ordered in quantity for bulk purchases or special sales. For more information, please contact:

U.S. Corporate and Government Sales

1-800-382-3419

corpsales@pearsontechgroup.com

For sales outside of the U.S., please contact:

International Sales

+1-317-581-3793

international@pearsontechgroup.com

Editor-in-Chief

Karen Gettman

Executive Editor

Neil Rowe

Acquisitions Editor

Brook Farling

Development Editor

Mark Renfrow

Managing Editor

Kristy Hart

Project Editor

Andy Beaster

Copy Editor

Keith Cline

Indexer

Word Wise Publishing
Services

Proofreader

San Dee Phillips

Technical Editor

Justin Etheredge

Publishing

Coordinator

Cindy Teeters

Interior Designer

Gary Adair

Cover Designer

Gary Adair

Composer

Nonie Ratcliff

Introduction

The Ruby language was developed by Yukihiro Matsumoto, who began developing Ruby on February 24, 1993. His main reason for doing so was his dissatisfaction with the scripting languages at the time, languages such as Perl and Python. He designed Ruby to be intuitive, to be natural, and to follow the “principle of least surprise”—making developers enjoy writing code and focus on the creative part of programming instead of fighting the language to fit their needs.

Ruby 1.0 was released on December 25, 1996, exactly 1 year after the first public release (version 0.9.5) of Ruby. For the first year afterward, Ruby was mainly used inside Japan. Its use expanded outside of Japan a few years later, but it was still used by a small number of eager early adapters.

In 2006, David Heinemeier Hansson from 37signals released a web development framework named Ruby on Rails. This innovative MVC web framework made the difference. More and more developers started using Ruby on Rails to develop their web applications and in the process became familiar with the Ruby language, too. Following these newcomers, a phrase was coined to explain how most current Ruby developers have come to use it: I came for the Rails, but I stayed for the Ruby.

Since then, Ruby has become one of the most popular programming languages in the world and is being used by thousands of developers every day.

Ruby is a dynamic language. It combines ideas from Perl, Smalltalk, Eiffel, Ada, and Lisp to provide an intuitive, flexible, and simple-to-use language. Its strengths are in its permissive syntax and powerful built-in capabilities, especially metaprogramming capabilities. However, it never appealed to .NET developers because it lacked integration with .NET code. This situation has changed with the birth of IronRuby.

IronRuby is Microsoft’s implementation of the Ruby language. It runs on top of the Dynamic Language Runtime (DLR), which is a special dynamic language service provider that is built on top of the Common Language Runtime (CLR).

IronRuby provides seamless integration with .NET code. It enables you to use .NET objects in Ruby code just as if they were pure Ruby objects. This opens vast opportunities to the .NET world and to the Ruby world. Both sides gain the power and strength of the other and provide together a new and exciting development environment.

In this book, I take you through all aspects of IronRuby so that you learn how to best leverage the language for the simplest of tasks to the most advanced ones.

Part I, “Introduction to IronRuby,” is an overview of the different pieces that make IronRuby possible. You learn about where it all began and the main concepts of the Ruby language, the .NET Framework, and the DLR. Part I ends with a chapter that introduces you to IronRuby for the first time. In that chapter, you learn the basics about using IronRuby and witness the powerful capabilities it brings to your development environment.

Part II, “The Ruby Language,” is devoted to an in-depth tutorial of the Ruby language. The part starts with the basic syntax, goes on with Ruby object-oriented programming capabilities, and ends with advanced concepts and techniques.

In Part III, “IronRuby Fundamentals,” I add the Iron to Ruby. This part contains all the information you need about IronRuby .NET integration. I explain how every .NET item can be used from IronRuby—from variables to implementing .NET interfaces.

Part IV, “IronRuby and the .Net World,” is the practical part. It contains guides for how to use IronRuby in several different scenarios. Most of the current .NET and Ruby frameworks are explained, including WPF, ASP.Net MVC, Ruby on Rails, and Silverlight. In addition, other possible usages are explained, such as testing .NET code using Ruby’s different unit testing frameworks and running IronRuby code from .NET code.

The last part of the book, Part V, “Advanced IronRuby,” covers IronRuby advanced topics. If you want to extend IronRuby objects or to create .NET code libraries that fit better to Ruby code, you will be interested in what this part has to offer.

I believe that IronRuby can enhance your work and enable you to do things you have not done before. I hope you find this book helpful and informative and that you can exploit its contents in your own projects and development tasks.

CHAPTER 4

Getting Started with IronRuby

IronRuby is Microsoft's implementation of the Ruby language on top of the DLR. Its main goal is to provide seamless interoperability between Ruby and the .NET Framework.

IronRuby combines the powers of both the .NET Framework and the Ruby language. On the one hand, it contains the built-in capabilities of Ruby, and on the other hand, it is capable of using the wide variety of frameworks and libraries of the .NET Framework. The combination opens a whole new set of opportunities to both Ruby and .NET developers.

In this chapter, you get your first taste of IronRuby. You install it, read an overview of the language and its tools, and start to discover the power it brings to the .NET family.

Overview

IronRuby is Microsoft's implementation of the Ruby programming language. It is built on top of the DLR and provides seamless integration between Ruby code and .NET Framework code. It is compatible with Ruby 1.8.6 and runs on .NET Framework 2.0 Service Pack 1 and above.

IronRuby was first announced on April 30, 2007, at the MIX conference. *Iron*, in its name, is actually an acronym and stands for "implementation running on .NET."

IronRuby is supported by the Common Language Runtime (CLR) and Mono, which means that it can be run on Windows, Linux, UNIX, BSD, Mac, and all other operating systems that are supported by Mono. Apart from operating

IN THIS CHAPTER

- ▶ Overview
- ▶ Installing IronRuby
- ▶ Executables and Tools
- ▶ Development Environments
- ▶ The Power of IronRuby

systems, IronRuby can also be run from the browser using Silverlight. (See Chapter 16, “Silverlight,” for more about IronRuby and Silverlight.)

IronRuby is an open source project and is released with full source code under the Microsoft Public License (MS-PL). The code is hosted on GitHub and can be downloaded from the CodePlex site, too. Because it is an open source project, the IronRuby team is looking for contributions both in bug fixing and library implementation. Look at the contribution page on IronRuby’s GitHub home page to see how you can help (<http://wiki.github.com/ironruby/ironruby/contributing>).

Installing IronRuby

IronRuby runs on .NET Framework 2.0 Service Pack 1 and above. Therefore, you need to have .NET Framework 2.0 SP1 or above installed on your machine before you start. Same goes for every machine on which you deploy your IronRuby applications.

You can download .NET Framework 2.0 SP1 from <http://www.microsoft.com/downloads/details.aspx?familyid=79BC3B77-E02C-4AD3-AACF-A7633F706BA5>.

When you have the correct framework, we can move on and install IronRuby. The recommended method of installing IronRuby is by using the IronRuby installer. Follow the next steps to do so:

1. Visit <http://www.ironruby.net/download> and click the Download IronRuby link.

2. The downloaded file is an MSI file. Double-click it to start the installation.

The first input you will be asked to enter is the installation folder for IronRuby. The default is your program files directory\IronRuby. The folder you choose will be referenced as the IronRuby installation folder throughout this book.

During the installation, you will be asked to select the features to install. The features that will be presented are as follows:

- ▶ **Runtime:** The main files of IronRuby. This feature is required.
 - ▶ **Standard Library:** Ruby standard libraries. Needed if the standard libraries are used in IronRuby code.
 - ▶ **Samples:** Sample IronRuby applications like WPF and PowerShell samples.
 - ▶ **Silverlight Binaries:** Binaries needed for using IronRuby in Silverlight applications.
 - ▶ **Add IronRuby to %PATH%:** Adds IronRuby binaries path to the PATH environment variable. This spares the need to provide full path to IronRuby executables when they are called from the command line.
3. After approving all steps, the installation of IronRuby takes place.

Another option to install IronRuby is manually. IronRuby can be downloaded as a zip package. This lets you fully control the installation process but also forces you to execute the automatic tasks manually (optionally).

Follow the next steps to install IronRuby manually:

1. Visit IronRuby's CodePlex homepage at <http://ironruby.codeplex.com> and click on the Downloads button on the page menu. In the downloads page, choose to download the IronRuby ZIP package.
2. After the download is complete, extract this Zip file to the folder in which you want to place IronRuby (for example, C:\IronRuby). This folder will be referenced as the IronRuby installation folder throughout this book.

You're actually done now and can start using IronRuby. However, if you want IronRuby to be available from every location on your Windows system, you want to add the <installation folder>\bin path to the Windows PATH environment variable. Be aware, however, that this can be done only if you have administrative privileges.

To do that, follow the next steps:

1. Navigate to Start > My Computer and right-click My Computer. On the menu, choose Properties as presented in Figure 4.1.

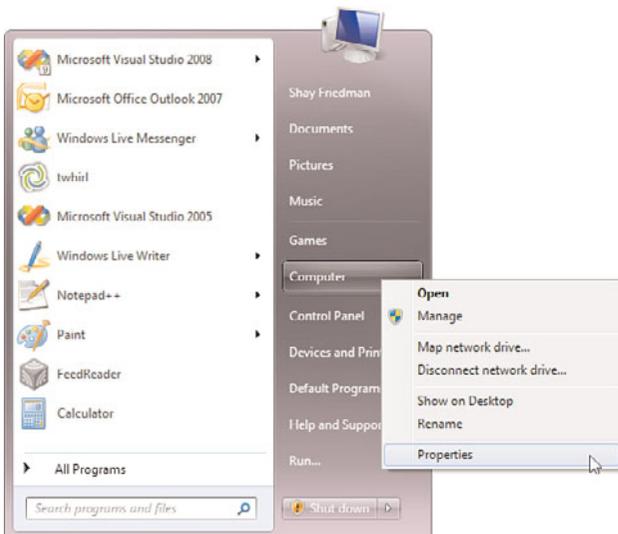


FIGURE 4.1 My Computer Properties Menu Item.

2. In the open dialog, click Advanced System Settings as presented in Figure 4.2.



FIGURE 4.2 The Advanced System Settings Link.

3. Click the Environment Variables button as presented in Figure 4.3.

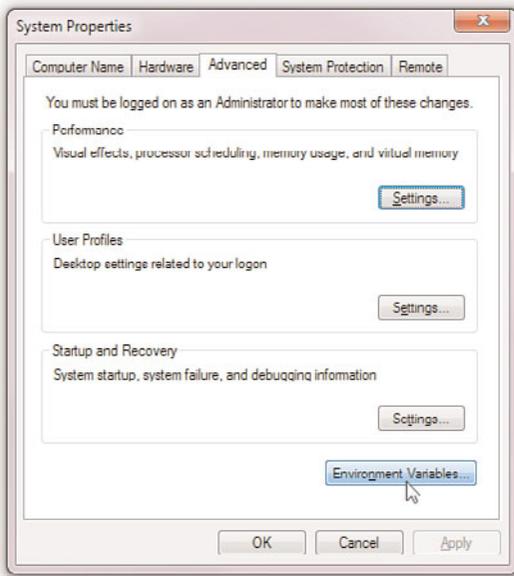


FIGURE 4.3 The Environment Variables button.

4. Find Path in the System Variables section (the lower part), select it, and click Edit as presented in Figure 4.4.
5. In the Edit System Variable dialog, place the cursor at the end of the Variable Value field and add a semicolon (;) and **<IronRuby installation folder>\Bin**. For example, if you've extracted IronRuby to C:\IronRuby, you add ;C:\IronRuby\Bin to the Variable Value field as presented in Figure 4.5.

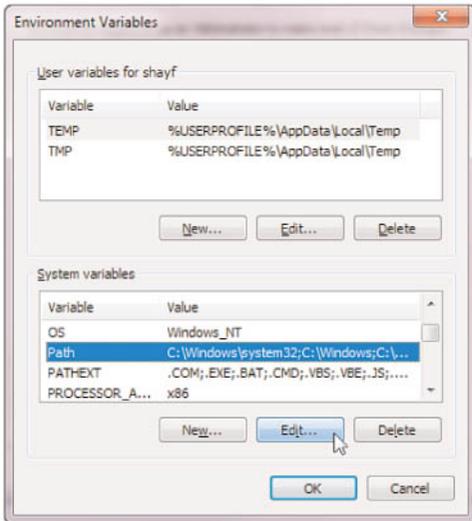


FIGURE 4.4 The PATH Environment Variable.

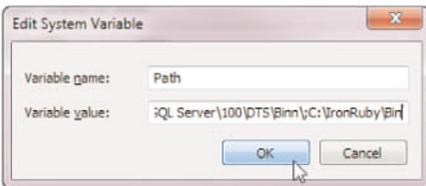


FIGURE 4.5 Setting the PATH Environment Variable.

6. Click OK on all the dialogs you've opened during the process to save the new setting.

Congratulations, IronRuby is now installed on your machine.

IronRuby Folders

After you extract IronRuby to the desired installation folder, you notice several folders there. Table 4.1 lists and describes the folders.

Getting the Sources

If you'd like to go deep into IronRuby and go through its code, you need to download the source code of IronRuby. The source code is hosted on GitHub and uses Git as its source control application. You can download the source code in a Zip format from <http://github.com/ironruby/ironruby/zipball/master>.

TABLE 4.1 IronRuby Folders and Their Roles

Folder	Description
Root	Contains license files, release notes, and the package readme file
Bin	Contains the IronRuby binaries, executables, and tools
Libs	Contains the standard libraries, including special IronRuby libraries and the RubyGems repository
Samples	Contains a few IronRuby samples and the IronRuby tutorial application
Silverlight	Contains binaries, samples, and tools for embedding IronRuby in Silverlight

To download the sources, you do not have to install Git on your machine. You need to do so only to contribute to the IronRuby code. For more information about how to contribute and how to use Git, look at the IronRuby wiki at <http://wiki.github.com/ironruby/ironruby>.

Executables and Tools

IronRuby comes with several different executables and tools. All of them are located under the Bin folder in the IronRuby installation directory.

Table 4.2 lists and describes the executables and tools in the IronRuby Bin folder, and then the following subsections take a closer look at them.

TABLE 4.2 IronRuby Executables and Tools

File	Description
ir.exe and ir64.exe	The main IronRuby executable. It is the IronRuby interpreter file. ir64.exe is for 64-bit systems.
iirb.bat	The IronRuby REPL (read-evaluate-print loop) console.
igem.bat	Used to work install and manage RubyGems. See Chapter 8, “Advanced Ruby.” for more information about RubyGems.
irackup.bat	Runs Rack, which is a Ruby framework that simplifies the way of interacting with different Ruby web servers.
irake.bat	Executes the Rake. See Chapter 8 for more about Rake.
irails.bat	Used to create a Ruby on Rails application. See Chapter 14, “Ruby On Rails.” for more about Ruby on Rails.
irdoc.bat	Runs RDoc, which is a Ruby tool to create formatted documentation out of Ruby code. The output can be plain text or a formatted HTML.
iir.bat	Ruby tool to read the textual documentation of Ruby objects. (The documentation is created by RDoc.)

The IronRuby Interpreter (ir.exe)

The IronRuby interpreter is the heart of IronRuby. Everything goes through it. For example, all the tools mentioned in Table 4.2 eventually run `ir.exe`.

The IronRuby interpreter can run Ruby files as well as a REPL console. These are two different modes, and so I discuss each of them separately.

REPL Console Mode

The REPL console mode opens a console where you can write code and execute it immediately.

To run IronRuby in the REPL console mode, follow these steps:

1. Click Start > Run.
2. Type `cmd` and click OK. The command prompt opens.
3. If you haven't added the IronRuby installation folder to the Path system variable, navigate to `<installation folder>\Bin`. If you have updated the Path system variable, skip this step.
4. Type `ir` and press Enter.

The IronRuby REPL console opens. You can now write Ruby code there (for example, `puts 'hello world'`) or even write whole classes and use them.

The format is simple. Each line where you can write a Ruby statement starts with a triple right-angle sign (`>>>`). If the line is not assumed to be continued (like method or class definitions), when you press Enter its output (if any) is printed to the console and the next line contains the return value of the statement preceded by an equal or greater than sign (`=>`). If the line is assumed to be continued, the next statement line starts with an ellipsis (`...`).

THE RETURN VALUE OF METHODS THAT DO NOT RETURN ONE

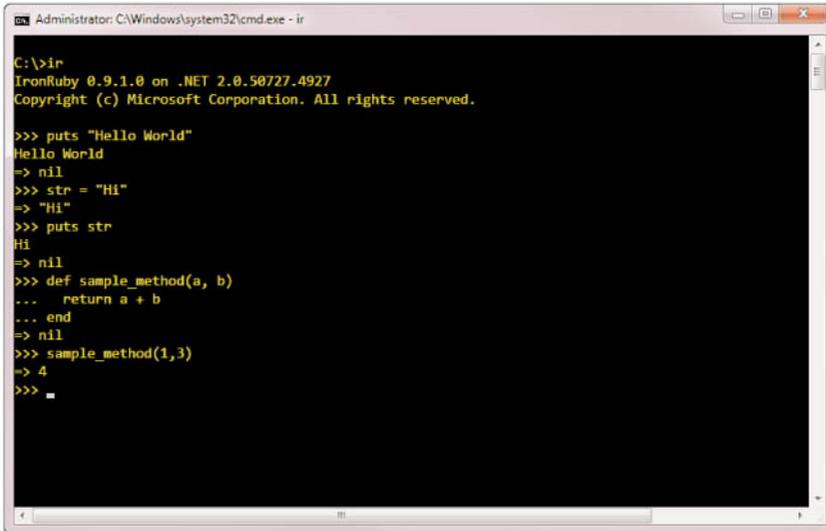
You notice that after executing methods with no return value, such as `puts`, the console still shows a return value: `nil`. This is really the return value.

Every method in Ruby returns a value. If the method code doesn't return a value, the method returns `nil` to the caller.

Figure 4.6 shows an REPL console session.

The REPL console mode has some specific command-line switches that can be used when running `ir.exe`. Table 4.3 lists and describes the switches. Be aware that the switches are case sensitive.

These are the REPL mode-only switches. All other switches appear on Table 4.4, and some can be used in this mode, too.



```
Administrator: C:\Windows\system32\cmd.exe - ir
C:\>ir
IronRuby 0.9.1.0 on .NET 2.0.50727.4927
Copyright (c) Microsoft Corporation. All rights reserved.

>>> puts "Hello World"
Hello World
-> nil
>>> str = "Hi"
-> "Hi"
>>> puts str
Hi
-> nil
>>> def sample_method(a, b)
...   return a + b
... end
-> nil
>>> sample_method(1,3)
-> 4
>>> .
```

FIGURE 4.6 A screenshot of an IronRuby REPL console session.

TABLE 4.3 ir.exe REPL Mode Command-Line Switches

Switch	Description
-	Makes the REPL console colorful.
X:ColorfulConsole	Prompt signs such as >>> and ... appear gray, errors red, and warnings yellow; messages (like the banner on top) appear cyan, and all other output uses the default console output color.

File Execution Mode

This mode executes a given file. Its format is as follows:

```
ir [options] [file path] [file arguments]
```

All switches and ir.exe command-line arguments should be placed before the file path. Otherwise, they will be considered arguments of the file and will be passed to it rather than to the interpreter.

The simplest way to execute an IronRuby file is by passing only its path to the interpreter. For example, the next command executes the file test.rb, which is located within the current directory:

```
ir test.rb
```

Along with this way, IronRuby provides several command-line arguments that affect the way the code executes. Table 4.4 lists and describes available arguments for the `ir.exe` file execution mode.

TABLE 4.4 `ir.exe` Command-Line Arguments for File Execution Mode

Argument	Description
-d or -D	Debug mode. Allows using breakpoints in Ruby code with the Visual Studio debugger.
-e "command"	Executes the command and exits. Several -e arguments are allowed. When used, the file path should not be passed and will be ignored if it exists. For example: <code>ir -e "str = 'Hello World'" -e "puts str"</code>
-I "directory"	Includes the given directory in the <code>\$LOAD_PATH</code> variable. This means that it will be included in the search paths for required libraries.
-r "library"	Requires the library before executing the file. For example: <code>ir -r "csv" test.rb</code>
-v	Prints the IronRuby version on the first line.
-w	Turns on warnings in the default level (verbose).
-W[level]	Sets the warning level. 0 = silence, 1 = medium, 2 = verbose (default). For example: <code>ir -W1 test.rb</code>
-K[kcode]	Specifies KANJI (Japanese) code set. E or e = EUC, S or s = SJIS, U or u = UTF8. For example: <code>ir -KU test.rb</code>
-trace	Enables Ruby tracing capabilities.
-profile	Enables profiling. When this switch exists, a <code>profile.log</code> file will be added to the directory of the executed file with profiling information about the latest execution.
-18 or -19 Or -20	Run IronRuby in Ruby 1.8 compatibility mode (default), Ruby 1.9, or Ruby 2.0 compatibility mode accordingly. Ruby 2.0 doesn't currently exist, so this switch is for future release only.

TABLE 4.4 ir.exe Command-Line Arguments for File Execution Mode

Argument	Description
-X:ExceptionDetail	In this mode, every exception is presented with a full call stack.
-X:NoAdaptiveCompilation	Disables adaptive compilation feature. This affects performance (for the worse).
-X:PassExceptions	In this mode, exceptions are not caught by the interpreter. This means that in case of an exception with no handling in the script, the application crashes.
-X:PrivateBinding	Enables binding to private members of CLR objects. Chapter 9, “.Net Interoperability Fundamentals,” discusses the uses of this switch.
-X:ShowClrExceptions	When this mode is on, a CLR exception part is added to every exception with full exception details (the <code>exception.ToString</code> output).
-X:CompilationThreshold	Specifies the number of iterations before the interpreter starts compiling the code. Should be followed by a number. Note that this switch can affect performance dramatically. Hence it is not recommended to use it when not needed.
-h	Shows all available command-line arguments with a short description. When this exists, the file or REPL console does not run.

Development Environments

Support for IronRuby in Visual Studio is not available in IronRuby 1.0. Such support is not in Microsoft’s current plans, and no one can really promise it will be in the near future.

However, the Ruby language already has several IDEs available. This section discusses some of them so that you can choose the one that best fits your needs.

Ruby in Steel

This commercial add-on to Visual Studio by SapphireSteel makes developing Ruby applications inside Visual Studio much more natural. It adds new Ruby project types, intellisense, code snippets, and syntax highlighting.

Figure 4.7 is a screenshot from Visual Studio that shows the syntax highlighting and intellisense capabilities of Ruby in Steel.

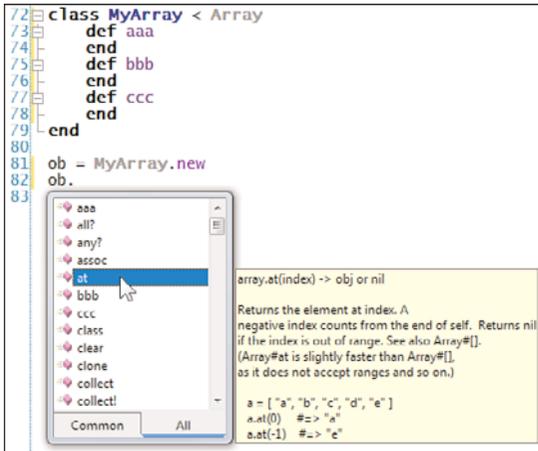


FIGURE 4.7 Ruby in Steel syntax highlighting and intellisense.

Although you cannot run IronRuby with Ruby in Steel out of the box, it is possible to alter the solution settings to execute `ir.exe`. Follow these steps to do so:

1. Inside Visual Studio, click Project > Project Settings.
2. In the Settings window, you see a Ruby region with a “Ruby Interpreter” line. Change the value on this line to the path of `ir.exe`.
For example, if you installed IronRuby in `C:\IronRuby`, you must set the value to “`C:\IronRuby\Bin\ir.exe`”.
3. Save the project and press `Ctrl + F5` to execute the Ruby files.

Ruby in Steel is a commercial product that costs money. It supports Visual Studio 2005, 2008, and also machines without Visual Studio at all (uses the Visual Studio Shell).

To read more about it, try it, or buy it, visit <http://www.sapphiresteel.com/Ruby-In-Steel-Developer-Overview>.

NetBeans

NetBeans is a free, open source IDE that supports several programming languages, along with the Ruby language. It is Java-based and can run on all operating systems that run Java applications.

For Ruby, you get code completion, naming convention warnings, a convenient project tree, and Ruby on Rails support.

Figure 4.8 shows the NetBeans window with Ruby code inside.

RubyMine doesn't come with IronRuby as its Ruby interpreter, and you have to add it as one to execute files with the IronRuby interpreter directly from the interface.

Follow these steps to add IronRuby as a Ruby interpreter in RubyMine:

1. Go to File > Settings.
2. On the left, choose Ruby SDKs and Gems.
3. On the settings on the right, click the Add SDK button, which is located in the upper-right corner of the dialog.
4. On the file selector dialog that opens, navigate to <IronRuby installation folder>\Bin\ir.exe. Click OK after you select the file.
5. When you click OK on the settings form, RubyMine makes IronRuby its default interpreter.

Before you run a file, you need to modify something else. RubyMine uses configuration settings for each execution. In this configuration, the command-line arguments are sent to the interpreter. The default ones do not work with IronRuby, and you need to remove them. Follow these steps to do that:

1. Go to Run > Edit Configurations.
2. Click the Edit Defaults button, which is located in the lower-left corner of the dialog.
3. Choose Ruby on the left panel.
4. On the right, clear the text from the Ruby Arguments field.
5. Click OK on all open dialogs to save the changes.

Now you can work on and run files by using the IronRuby interpreter.

RubyMine is a commercial product that costs money. It works on Windows, Mac OS X, and Linux. You can read more about it, try it, and buy it at <http://www.jetbrains.com/ruby>.

Others

Along with these IDEs, a lot of other great IDEs are available. Some are appropriate for bigger applications, and some for smaller applications and scripts. Most of them offer simple code completion and syntax highlighting.

Like the others, they still cannot run IronRuby directly from their interface, so you have to switch to the command prompt and use `ir.exe` to run the Ruby file you've been working on.

Some of these IDEs are RadRails (<http://www.apptana.com/radrails>), SciTE (<http://www.scintilla.org/SciTE.html>), and Notepad++ (<http://notepad-plus.sourceforge.net>). Search the Internet for "Ruby IDEs" to find more Ruby IDEs.

The Power of IronRuby

As a finale to the IronRuby introduction, I want to leave you wanting more. Instead of just writing a simple Hello World application, I want you to see how great IronRuby is and how it can enhance your development work.

Ruby comes with very powerful built-in metaprogramming capabilities. One of its features is a built-in method called `method_missing` (read more about it in Chapter 6, “Ruby’s Code-Containing Structures”). The `method_missing` method catches all calls to undefined methods and lets you handle them the way you want.

In Listing 4.1, by using `method_missing`, I implement a `Recorder` class that records calls to methods and plays them back on request. Notice how easy it is to accomplish this with Ruby. Even though you might not be familiar with all the syntax elements, this code is pretty simple and straightforward.

LISTING 4.1 A Recorder Class Implementation

```
class Recorder
  def initialize
    @calls = []
  end
  def method_missing(method, *args, &block)
    @calls << [method, args, block]
  end
  def playback(obj)
    @calls.each do |method, args, block|
      obj.send method, *args, &block
    end
  end
end
```

Now that the `Recorder` class is ready, we can take advantage of it. Unlike regular Ruby code, we can take advantage of IronRuby capabilities and run the `Recorder` class with .NET code. Listing 4.2 takes advantage of the `Stack` class of the .NET Framework and runs recorded operations on it using our `Recorder` class.

LISTING 4.2 Using the Recorder Class on a CLR Object

```
recorder = Recorder.new
recorder.push 5.6
recorder.pop
recorder.push 1
recorder.push "IronRuby"

# Run the recorded calls on the CLR Stack instance
```

```
stack = System::Collections::Stack.new
recorder.playback(stack)
recorder.playback(stack)

stack.each { |x| puts x }
# Prints "IronRuby
#       1
#       IronRuby
#       1"
```

Summary

After reading this chapter, you are ready to start developing with IronRuby on your computer. You have installed IronRuby and have been introduced to the tools it comes with. This chapter also covered a few options for IronRuby development environments. At the end of the chapter, you saw an IronRuby sample that gave you a taste of the great power IronRuby holds inside.

We now delve further into the Ruby language and examine the fundamentals of IronRuby, and then you learn how to use IronRuby with the different frameworks of Ruby and .NET, all on the road to becoming an IronRuby master.

Index

SYMBOLS

>, >= (greater than/greater than or equal to) operator, 65
<, <= (less than/less than or equal to) operator, 65 []
(array access) operator, 112, 114
|| (Boolean OR) operator, 65
== (equality) operator, 112
<=> (general comparison) operator, 65
< <= > > (order comparison) operator, 112
() (parentheses), 45
<< (shift-left) operator, 112
>> (shift-right) operator, 112
-@ (unary minus) operator, 112
+@ (unary plus) operator, 112
| (vertical bars), 96
! (no pattern match) operator, 65
!= (not equal to) operator, 65
\$LOAD_PATH variable, 210
&& (Boolean AND) operator, 65
; (semicolon), 44
=== (case equality) operator, 65, 112, 113
== (equal to) operator, 65
@ (at sign), 453
[] = (array access setter) operator, 112, 114

A

abbrev library, 132
 references, 135
About Your Application's Environment link, 337
abstract classes, 123, 242
 implementation, 247
abstract methods, 246-247
accept_verbs method, 374
accessing. *See also* security
 arrays, 55-56
 CAS, 19
data. *See* data access
 file properties, 173-174
 hashes, 58
 HTML, Silverlight, 414-415
 strings, 53-54
 variables from outside, 106-107
 XAML elements, 412-414
accessors
 classes, 107-109
 implementation, 180
 properties, overriding, 252
ActionController, 341
ActionExecutingContext object, 388

actions

- canceling, 388
- controllers, return values, 371
- customizing, 395-396
- filters, 387-390

ActionView, 341**ActiveRecord model, 340-341****Ada, 1, 5****add_EventName method, 253****adding**

- build configurations, 502-504
- code, Silverlight, 411-415
- connection strings to classes, 262
- functionality, 353-354
- IronRubyMvs Dll files, Visual Studio, 365
- layouts, 351-352
- log entries, 143
- references to assemblies, 367
- Refresh button, 360-361
- stylesheets, 350-351
- video, 418
- web pages to Silverlight, 406-408
- WinForms
 - controls, 289-293
 - functionality, 293-295

add_log method, 456**add_one method, 88****Advanced System Settings, 27****AdventureWorksLT, 260****after_action filter, 389****after method, 443****after_result filter, 391****alias_action method, 375****aliasing**

- methods, 499
- namespaces, 214
- syntax, 215

alias keyword, methods, 91-92**and, 30****And directive, 448****AND operator, 65****animation**

- Silverlight, 417-418
- WPF, 324-325

app/controllers folder, 333**App folder, 333****app/helpers folder, 334****Application class, 304****Application.run method, 305****applications**

- ASP.NET MVC, building, 365-367
- code-containing structures, 86
- controllers, creating, 375-377
- C#/VB.Net, 459-461

data access. See data access

- execution code, writing, 300
- Hello World!, 48
- Java, 35
- layouts, applying, 360
- models, creating, 368-371
- NetBeans, 35-36

reflection. See reflection

- RoR, creating, 332-337
- Ruby in Steel, 34-35
- Silverlight, 402-406, 405
- structures, WinForms, 282
- threads, 161-169
- views, creating, 382-385

applying

- built-in mixins, 488
- CachedDataAccess class, 278-279
- designer, Visual Studio, 295-296
- extensions
 - IronRuby, 501
 - .NET, 509-510
- gems, 183-184
- layouts, 360
- libraries, MVC, 375
- methods as block arguments, 94
- mixins, 254-255
- objects
 - IronRuby, 470-471
 - .NET, 214-231
- previous layout lists, 358-360
- regular expressions, 61
- SqlServerAccessor class, 265
- standard libraries, 131
- symbols, 58

app/models folder, 334**app/view folder, 334****app/view/layouts folder, 334****architecture**

- DLR, 20-21, 22-23. *See also* DLR
- .NET Framework, 15-16
- REST, 339-340

arithmetic operators, 49, 112**around_action filter, 382****around_result filter, 392****Array class methods, 56-57****arrays**

- accessing, 55-56
- defining, 54-55
- ranges, converting, 59
- Ruby, 54-57

ASP.NET MVC, 363

- applications, building, 365-367
- classic ASP.NET, 398
- environments, preparing, 363-365
- features, 398
- filters, 387-396
- installing, 364
- routes, 385-387
- validations, 396-398

assemblies, 18

- deleting, 366
- GAC, 18
- loading, 260, 267
- .NET
 - loading, 207-210
 - WinForms, 296
- references, adding, 367
- requirements, Chat class, 282

- running, 16
- WinForms, loading, 285
- assertions, unit testing, 428-431
- associating methods with objects, 94-95
- at sign (@), 453
- attributes
 - Window, 309-310
 - WindowStyle, 310-311
- audio, 418
- AuthorizationContext object, 392-393
- authorization filters, 392-393
- automatic log rotation, 144
- availability
 - libraries, 11, 133-135
 - socket services, 149-152

B

- Background directive, Cucumber, 452
- Base Class Library. *See* BCL
- base64 library, 132
 - references, 135
- BasicSocket class, 155
- BCL (Base Class Library), 19
- BDD (behavior-driven development), 435
- be_an_instance_of RSpec expression matcher, 440
- be_an RSpec expression matcher, 440
- be_a RSpec expression matcher, 440
- be_close RSpec expression matcher, 440
- be_false RSpec expression matcher, 440
- before_action filter, 388
- before method, 443
- before_result filter, 390-391
- BEGIN class, 77
- begin clause, 88
- behavior
 - RSpec, creating, 438
 - rules, Cucumber, 443-457
- behavior-driven development (BDD), 435
- Behavior object, 438
- be_instance_of RSpec expression matcher, 440
- be_kind_of RSpec expression matcher, 440
- benchmark library, 132, 136
- be_nil RSpec expression matcher, 440
- be_[predicate] RSpec expression matcher, 440
- best practices, extension development, 481
- be_true RSpec expression matcher, 440
- BigDecimal library, 132
 - references, 136
- binary marshaling, 181
- binders, 24
- binding
 - data
 - Silverlight, 419-422
 - WPF, 325-329
 - dynamic data, 327-328
 - private binding mode, 213-214
 - static data, 325-326

- Bin folder, 30
- bitwise operators, 112
- block arguments, applying methods as, 94
- BlockParam parameter, 492
- blocks, 96-97, 496
 - flow, 100-101
- body parts, web pages, 352
- Booleans, 60
- break keyword, 74
- brushes
 - Silverlight, 415
 - WPF, 322-324
- BuildConfig property, 483, 485
- build configurations, adding, 502-504
- builder pattern, 196-199
- building
 - applications, ASP.NET MVC, 365-367
 - chat, WinForms, 285-299
 - Chat class, 282-285
 - ChatForm class, 285-286
 - class structures, 260, 267
 - connection strings, 261, 267
 - extensions, 501-510
- built-in mixins, 488
- buttons
 - Environment Variables, 28
 - Refresh, adding, 360-361

C

- CachedDataAccess class, 276-279
 - applying, 278-279
- cached_data_access.rb file, 277-278
- caches, 22
 - GAC, 18
- calc_numerological_value.feature, 448
- calling
 - accessors, 108
 - class methods, 111
 - methods, 45-46, 94
 - procs, 97
- call sites, 23
- CallSiteStorage parameter, 492
- canceled actions, 388
- canvas, formatting, 320-321
- Canvas control, 411
- CAS (Code Access Security), 19
- case statement, 67-69
- catching exceptions within methods, 88
- CGI (Common Gateway Interface), 132
- change RSpec expression matcher, 440
- characters, numeric values of, 51
- chat, building WinForms, 285-299
- Chat class, 282
 - building, 282-285
- Chatform class, 282, 297-299
 - building, 285-286

chat.rb file, 284-285

ChatRunner class, 282

Chiron, initParams parameter, 407

CIL (Common Intermediate Language), 17-18

Class class methods, 232-233

classes

abstract, 123, 247

accessors, 107-109

Application, 304

Array, methods, 56-57

BasicSocket, 155

BCL, 19

BEGIN, 77

CachedDataAccess, 276-279

Chat, 282-285

ChatForm, 282, 285-286

Chatform, 297-299

ChatRunner, 282

Class, methods, 232-233

CLR, 216, 217

inheritance from, 239-243

opening, 254-256

code standards, 47

connection strings, adding, 262

constants, 105-106

defining, 93

duck typing, 124-126

END, 77

errors, customizing, 85-86

Exception, methods, 78

ExceptionContext, 393-394

exceptions, 490-491

extensions, 488-491

File, 170

Form, initializing, 286

generic, 241-242

Hash, methods, 58-59

helper, 349-350

inheritance, 120-124

instances, creating, 102

IPSocket, 156

IronRuby, 235-236

IronRuby::Clr, 236

methods, 109-111

overriding, 122

undefining, 491

module-contained objects, 126

modules, 126

Mutex, 167-168

MySQLAccessor, 272

Numerology.Calculator, 426

Object

methods, 231-232

opening, 255-256

Recorder, implementation, 38-39

regular, 239-242

Ruby, 101-126

RubyClassAttribute properties, 488-489

RubyMethodAttributes, 492, 494-495

ScriptEngine, 23, 463-465

ScriptRuntime, 23, 462-463

ScriptRuntimeSetup, 462

ScriptScope, 23, 465-466

ScriptSource, 23, 466-467

sealed, 243

singleton, 490

SqlServerAccessor, applying, 265

Stack, 38

static, 243

String, 53, 54, 234-235

structures, building, 260, 267

System.String, 254

System.Windows.Forms.Application, 300

TCPServer, 156

TCPsocket, 156, 283

ToDoListModel, 370

UDPSocket, 156

variables, 102-107, 103

visibility control, 118-120

classic ASP.NET, 398

class keyword, 101

clauses

begin, 88

block, 96

ensure, 82-83

rescue statement, 79

CLI (Common Language Infrastructure), 14-18

ClientSize property, 287

closures, blocks, 97. See also blocks

CLR (Common Language Runtime), 1, 15, 17

classes, 216, 217

inheritance from, 239-243

opening, 254-256

constants, 222

delegates, 217-218

fields, 228

interfaces, 216, 243-244

members, hiding, 487

namespaces, converting, 214

naming conventions, 212-213

objects

applying Recorder class on, 38

reflection, 237

properties, 228-229

Ruby, type differences, 211

structs, inheritance from, 243

clr_constructor method, 232

clr_ctor method, 232

clr_member method, 231

clr_members method, 233

clr_new method, 233

CoC (Convention over Configuration), 340

code

compiling, 467

creating, 504-506

dynamically, executing, 180-181

execution, 300-301, 467

file structure, 46-47

Gherkin, 451

hosts, 26

naming, 212

- .NET
 - Framework, 16
 - mapping, 210-214
 - standards, 211-213
- reading, 467
- reflection. See reflection**
- RSpec, injecting, 442-444
- Ruby, 2-5. *See also* Ruby
- ScriptEngine class, executing, 464-465
- ScriptRuntime class, executing, 463
- Silverlight, adding, 411-415
- source. See source code**
- standards, 47
- unit testing, 426-427
- XAML, 305-307
- Code Access Security (CAS), 19**
- code-containing structures, Ruby, 86**
- CodePlex, 22, 26**
- Collatz conjecture, 77**
- collisions**
- inheritance, 124
- .NET, mapping code, 210-214
- command-line**
- arguments for file execution mode, 33-34
- chr tool, 404-406
- databases, creating from, 337
- sl tool, 402-403
- switches, 31
- commands**
- db:migrate, 345-346
- icucumber, 457
- patterns, 190-192
- scaffold, 344, 345
- script/destroy, 345
- script/generate, 343-345
- script/generate controller, 346
- script/server, 342
- comma-separated value. See CSV**
- comments**
- code standards, 47
- syntax, 43-44
- Common Gateway Interface. See CGI**
- Common Intermediate Language. See CIL**
- Common Language Infrastructure. See CLI**
- Common Language Runtime. See CLR**
- Common Type System (CTS), 16**
- comparison operators, 64-65, 112, 114**
- Compatibility property, 483, 485**
- compilers, DLR, 24**
- compiling**
- code, 467
- JIT, 16
- complex library, 132**
- references, 137
- components**
- RoR, 340-342
- runtime, 22, 23-24
- conditions, 64-74**
- Config folder, 334**
- configuration method, 235**
- configuring**
- applications, RoR, 332-337
- behavior, RSpec, 438
- build, adding, 502-504
- class instances, 102
- controllers, 346-349
- databases, RoR, 334-337
- extensions, .NET, 478-501
- form properties, 287-289
- keys for SQL Server connections, 261
- PATH Environment Variable, 29
- RubyMine, 37
- ScriptRuntime class, 462-463
- values to variables, 44-45
- views, 346-349
- visibility control, 118-120
- web pages, 346-354
- connections**
- MySQL, opening, 268
- SQL Server, opening, 262
- strings
- adding, 262
- building, 261, 267
- examples of, 261
- connectors, MySQL, 260**
- consoles, 11**
- keys, 408
- modes, REPL, 31
- constants**
- classes, 105-106
- CLR, 222
- code standards, 47
- extensions, 501
- mapping, 222
- module-contained objects, 127
- Ruby, 63-64
- const-missing method, 118**
- constructors**
- defining, 102
- inheritance, 241
- methods, 497
- contacting**
- MySQL, 265-272
- SQL Server, 260-265
- content, WPF, 315-317**
- contribution pages, 26**
- ControlBox property, 287**
- controllers**
- ActionController, 341
- actions, return values, 371
- applications, creating, 375-377
- ASP.NET MVC validations, 396-397
- creating, 343, 346-349
- features, 371-372
- filters, 394
- MVC, 371-372
- views, creating, 378

controls

layouts

- Silverlight, 410-411
- WPF, 317-321

- Silverlight, 411

- StackPanel, 317-319

- structures, Ruby, 64-77

- WinForms, adding, 289-293

- WrapPanel, 318-319

Convention over Configuration. See CoC

conventions, naming, 212

- Ruby, 484

- unit testing, 427-428

converting

- namespaces, CLR, 214

- ranges to arrays, 59

CRUD (Create Read Update Delete), 259, 341

CSV (comma-separated value), 132

- references, 137

CTS (Common Type System), 16

Cucumber, 443-457

- Background directive, 452

- executing, 457

- features, 446-447

- hooks, 454-455

- installing, 445

- multilanguage, 456-457

- project structures, 445-446

- scenarios, 447-452

- tags, 453-454

- worlds, 456

culture of assemblies, 18

customizing

- error classes, 85-86

- filters, 395-396

- index pages, 356

- installation, 26

- routes, 386-387

C#/VB.Net, 458-459

- applications, 459-461

- external libraries, 472-473

- IronRuby, executing from, 468-473

- ScriptEngine class, 463-465

- ScriptRuntime class, 462-463

- ScriptScope class, 465-466

- ScriptSource class, 466-467

D

-D, 33

-d, 33

data access, 256

- environments, preparing, 260-259

- overview, 259

- SQL Server, contacting, 260-265

databases. See also MySQL; SQL Server

- MySQL, preparing, 266

- queries, 263-264, 268

- RoR configuration, 334-337

- web pages, formatting, 354-361

data binding

- Silverlight, 419-422

- WPF, 325-329

data templates, Silverlight, 422

dates

- and times, Ruby, 62-63

- libraries, 132

DayError, 86

Db folder, 334

db:migrate command, 345-346

debug key, 408

Debug library, 132

DebugMode property, 462

declaring

- floats, 49

- integers, 49

- types, 48

default parameter values, 92

Defineln property, 483, 486

defining

- arrays, 54-55

- blocks, 96

- classes, 93

- class methods, 110

- constants, 105

- constructors, 102

- exception types, 139

- hashes, 57

- Lambdas, 99

- methods, 88-89, 95-96

- operator behavior, 111

- proc objects, 97

- regular expressions, 60-61

- strings, 50

- types, 44

def keyword, 88

Delegate library, 132

delegates, CLR, 217-218

deleting

- assemblies, 366

- method definitions, 95-96

- records, MySQL, 269

- resources, 345

delimiters, strings, 50

describe method, 438

design

- MySQL, 272-276

- patterns, 186-202

- builder, 196-199

- command, 190-192

- iterator, 188-190

- observer, 194-196

- singleton, 192-194

- strategy, 186-188

- SQL Server, 272-276

designer, Visual Studio, 295-296

development

- environments, 34-37

integrated development environments. See IDEs

dialog boxes

- Edit System Variable, 28

- New Project, 366

differences between Lambdas and Procs, 99

digest library, 132

references, 138

directives

And, 448

Background, Cucumber, 452

directories

listing, 174-175

structures, 333-334

templates, sl tool, 403

Distributed Ruby. See Drb

dividing integers, 50

DLR (Dynamic Language Runtime), 1

architecture, 22-23

features, 23-24

overview of, 20-21

Doc folder, 334

documents. See also text

here, 51

words, modifying, 357

XAML, 305-307

XML

generating, 153

reading, 154

domain-specific languages. See DSLs

Don't Repeat Yourself. See DRY

DoubleAnimation element, 324

double-quoted strings, 44-50

downloading

code, 26

IronRubyMvs Dll files, 364-365

.NET Framework, 26

standard libraries, 159

Drb (Distributed Ruby), 132

DRY (Don't Repeat Yourself), 340

DSLs (domain-specific languages), 199-202

duck typing, 8-9

classes, 124-126

dynamic data

binding to, 327-328

Silverlight, 420-421

Dynamic Language Runtime. See DLR

dynamic languages, 6-7, 20-21

implementation, 24

dynamic messages, 139

E

each method, 59, 116

-e "command," 33

Edit System Variable dialog box, 28

Eiffel, 1, 5

elements

DoubleAnimation, 324

MediaElement, 418

Silverlight, retrieving, 412-414

Storyboard, 418

TextBlock, 304

WPF, retrieving, 308

else statements, 81-82

e2mmap library, 132, 139

empty namespaces, 214

END class, 77

end keyword, 88

English library, 132

references, 140-141

ensure clause, 82-83

entries, adding log, 143

enumerable objects, 72-73

enums, 221-222

environments

ASP.NET MVC, preparing, 363-365

data access, preparing, 260-259

development, 34-37

integrated development environments. See IDEs

reflection. See reflection

RoR

navigating, 342-346

preparing, 331-332

Silverlight, preparing, 402

target, .NET extensions, 482

Environment Variables button, 28

env.rb file, 446

Eql RSpec expression matcher, 440

Equal RSpec expression matcher, 441

Erb library, 132

references, 141-143

Eregexp library, 132

error.js file, 408

errors

classes, customizing, 85-86

DayError, 86

IOError, 80

StandardError, 85

SyntaxError, 80

events

handling, Silverlight, 414

layout, suppressing, 286

.NET, 218-221

overriding, 253-254

startup, 305

subscribing, 219-220

TextBlock.Loaded, 418

unsubscribe, 220

WPF, handling, 308-309

examples, tables, 451-452

Exception class methods, 78

ExecutionContext class, 393-394

exceptionDetail key, 408

exceptions

catching within methods, 88

classes, 490-491

filters, 393-394

handling, 78-86

raising, 83-85

- threads, 163
- types, defining, 139
- executables, 30-34**
- executing**
 - code, 467
 - dynamically, 180-181
 - RSpec, 443
 - ScriptEngine class, 464-465
 - ScriptRuntime class, 463
 - Cucumber, 457
 - IronRuby from C#/VB.Net, 468-473
 - virtual methods, 246
- execution code, writing, 300-301**
- expectation methods**
 - RSpec, 439-442
- expressions**
 - interpolation, 52
 - matchers, RSpec, 441-442
 - regular, 60-62
 - trees, 23, 24
- extending IronRuby, 473-478**
- Extends property, 484, 486**
- eXtensible Application Markup Language. See XAML**
- extensions**
 - classes, 488-491
 - constants, 501
 - infrastructure, 478-481
 - IronRuby
 - applying, 501
 - building, 501-510
 - methods, 491-500
 - .NET
 - applying, 509-510
 - creating, 478-501
 - modules, 482-488
 - projects, 481
 - Visual Studio, creating projects, 502
- external libraries, C#/VB.Net, 472-473**

F

- features**
 - ASPNet MVC, 398
 - controllers, 371-372
 - Cucumber, 446-447
 - DLR, 23-24
 - installation, 26
 - .NET Framework, 16-20
 - Ruby, 6-11
 - tagging, 453
- fields, CLR, 228**
- File class, 170**
- filenames, loading .NET assemblies, 208**
- File.open method, 170**
- files, 169-175**
 - cached_data_access.rb, 277-278
 - chat.rb, 284-285
 - code structure, 46-47
 - env.rb, 446
 - error.js, 408
 - initial project, creating, 333
 - IronRuby, executing, 468
 - IronRubyMvs Dll, adding to Visual Studio, 365
 - mysql.rb, 270-272
 - operations, 175
 - properties, accessing, 173-174
 - reading, 158, 170-172
 - sql.rb, 264-265
 - writing, 158, 172-173
 - ZIP, 27
- FileUtils library, 132**
 - references, 143
- filters**
 - actions, 387-390
 - ASPNet MVC, 387-396
 - authorization, 392-393
 - controllers, 394
 - customizing, 395-396
 - exceptions, 393-394
 - results, 390-392
- finding**
 - gems, 185
 - living objects, 176-177
 - standard libraries, 159
- Find library, 133**
- find_name method, 308**
- flags, regular expressions, 61**
- floats, declaring, 49**
- flow**
 - filters, 387
 - keywords, 100-101
 - loops, modifying, 74
- folders, 29-30**
 - Lib, 209
 - RoR, 333-334
 - structures, sl tool, 403
- for loops, 71**
- formatting**
 - applications
 - controllers, 375-377
 - RoR, 332-337
 - views, 382-385
 - behavior, RSpec, 438
 - canvas, 320-321
 - class instances, 102
 - code, 504-506
 - controllers, 346-349
 - databases, web pages, 354-361
 - extensions, .NET, 478-501
 - form properties, 287-289
 - MVC views, 378-385
 - views, 346-349
 - visibility control, 118-120
 - web pages, 346-354
- FormBorderStyle property, 287**
- Form class, initializing, 286**
- forms**
 - data, posting, 145
 - properties, formatting, 287-289
 - WinForms, 280. *See also* WinForms
- Forwardable library, 133**

frameworks

- Cucumber, 443-457
- .NET
 - Framework, 20
 - overview of, 11-13

RoR. See RoR

- RSpec, 435-444

WPF. See WPF**full paths, loading .NET assemblies, 208****functionality**

- adding, 353-354
- WinForms, adding, 293-295

G**GAC (Global Assembly Cache), 18****Garbage Collector, 176****gems**

- applying, 183-184
- finding, 185
- installing, 183
- RoR, installing, 332

generating

- ASP.NET MVC initial projects, 365-367
- helper classes, 344
- resources, web pages, 354-355
- XML documents, 153

Generator library, 133**generic classes, 229-231, 241-242****GetOptLong library, 133****get_products method, 276****Gherkin code, 451****GitHub, 26****Given step, 447****Global Assembly Cache. See GAC****global hooks, 454****globals method, 235****graphics**

- Silverlight, 415-417
- WPF, 321-325

Grid control, 410**grids, 319-320****GServer library, 133****guidelines, RoR, 339-311****GUI (graphical user interface) tools, installing, 260****H****-h, 34****Hailstone Sequence, 77****handled property, 309****handling**

- events
 - Silverlight, 414
 - WPF, 308-309
- exceptions, 78-86
- files, 169-175

Hansson, David Heinemeier, 1**Hash class methods, 58-59****hashes, 57-59**

- accessing, 58
- defining, 57

have_at_least RSpec expression matcher, 441**have_at_most RSpec expression matcher, 441****Have RSpec expression matcher, 441****head parts, web pages, 352****Hello World!, 48****helper classes, 349-350**

- generating, 344

helper methods

- HTML, 378-379
- views, 380

here documents, 51**HideClrMembers property, 484****hiding**

- CLR members, 487
- methods, 500

highlighting syntax, 35**history**

- of .NET Framework, 13-14
- of Ruby, 5-6

hooks, Cucumber, 454-455**HostArguments property, 462****hosts**

- code, 26
- models, 22, 23

HostType property, 463**HTML (Hypertext Markup Language)**

- helper methods, 378-379
- Silverlight, accessing, 414-415

I**-I "command," 33****icucumber command, 457****identifiers, tags, 453****IDEs (integrated development environments), 35****if, 65-67****if-else statements, 69****igem.bat, 30****iir.bat, 30****iirb.bat, 30****implementation, 24-25**

- abstract classes, 247
- accessors, 180
- dynamic languages, 24
- invoking superclass method, 123
- Recorder class, 38-39
- regular methods, 247-248
- Ruby, 6
- sealed methods, 250-251
- static methods, 248-249
- steps, 449-451
- threads, 162
- virtual methods, 245-246

include method, 215
include RSpec expression matcher, 441
indexer methods, 224-225
index pages, customizing, 356
infrastructure extensions, 478-481
inheritance
 classes, 120-124
 CLR classes, 239-243
 CLR interfaces, 243-244
 CLR structs, 243
 collisions, 124
 constructors, 241
 numeric types, 49
Inherits parameter, 489-490
Initialize method, 102, 498-499
initializers, libraries, 508-509
initializing
 Form class, 286
 ScriptRuntime class, 462-463
 visibility, 102
initial project files, creating, 333
initiating Chat class, 282-283
initParams parameter, 407
injecting code, RSpec, 442-444
inserting records, MySQL, 269
installing, 26-30
 ASP.NET MVC, 364
 Cucumber, 445
 features, 26
 gems, 183, 332
 GUI tools, 260
 RSpec, 436
 SQL Server, 332
instances
 classes, creating, 102
 methods, 222
 variables, 103-104, 453
integers
 declaring, 49
 dividing, 50
integrated development environment. See IDEs
integration, 1
intellisense, 35
interfaces
 CGI, 132
 CLR, 216, 243-244
 programming, 506-508
WinForms. See WinForms
interoperability
 .NET, 203
interpolation
 expressions, 52
interpreters, 31. See also ir.exe
 RubyMine, 37
inverted until loops, 71
inverted while loops, 70
investigating objects, 177-178
invoking
 methods, 178-180
 superclass method implementation, 123
IOError, 80

IPAddr library, 133
IPSocket class, 156
irackup.bat, 30
irails.bat, 30
irake.bat, 30
irake tool, 337
irdoc.bat, 30
ir.exe, 30
 command-line arguments for file execution mode,
 33-34
ir64.exe, 30
IronRuby, 6. See also Ruby
 classes, 235-236
 C#/VB.Net, 468-473
 extending, 473-478
 extensions
 applying, 501
 building, 501-510
 objects, applying, 470-471
IronRuby::Clr class, 236
IronRubyMvs Dll files
 adding, 365
 downloading, 364-365
iterator pattern, 188-190

J

Java applications, 35
Java Virtual Machine. See JVM
JCode library, 133
JIT (just-in-time) compiling, 16
join method, 162
JRuby, 6
just-in-time. See JIT
JVM (Java Virtual Machine), 14

K

Kconv library, 133
keys, initParams Chiron-related, 408
keywords, flow, 100-101
-K[kcode], 33

L

lambdas, 99-100, 496
 flow, 100-101
Language-Integrated Query. See LINQ
languages, 1
 context, 24
 DSLs, 199-202
 dynamic, 6-7, 20-21
 object-oriented, 7
 programming, 13
Ruby. See Ruby
LanguageSetups property, 463
layout
 controls, WPF, 317-321
 events, suppressing, 286

layouts. See also formatting

- adding, 351-352
- applications, applying, 360
- previous layout lists, applying, 358-360
- Silverlight, 410-411

Lib folder, 209, 334**libraries**

- available, 11
- BCL, 19
- code without IronRuby extension methods, 505
- DLR, 22. *See also* DLR
- external, C#/VB.Net, 472-473
- initializers, 508-509
- MVC, 375
- RSpec requirements, 436-437
- standard, 130-131. *See also* standard libraries

Libs folder, 30**licenses, MS-PL, 22****limitations of RubyGems, 185****LINQ (Language-Integrated Query), 279-280****Linux operating system, 14****Lisp, 5****listen method, 283****listing directories, 174-175****lists, applying previous, 358-360****living objects, finding, 176-177****load_assembly method, 209, 501****load_component method, 412****loaded_assemblies method, 235****loaded_scripts method, 236****loading assemblies, 260, 267**

- .NET, 207-210

- WinForms, 285

load method, 210, 236**load_root_visual method, 412****local variables, 105****log entries, adding, 143****Log folder, 334****logger library, 133**

- references, 143

loops, 70-72

- flow, modifying, 74

- for, 71

- inverted until, 71

- inverted while, 70

- loop loops, 72

- numbers, 73-74

- ranges, 74

- until, 70

- while, 70

M**MacRuby, 6****MailRead library, 133****managing memory, .NET Framework, 19****manual installation, 27. See also installation****mapping**

- constants, 222
- namespaces, 256
- .NET code, 210-214
- objects, 211
- ORM, 341

marshaling, 181-182

- binary, 181
- textual, 182

matchers, expressions, 441-442**Match RSpec expression matcher, 441****MathN library, 133****Matrix library, 133****Matsumoto, Yukihiro, 1****Matz's Ruby Interpreter. See MRI****MaximizeBox property, 287****media. See also animation; graphics**

- Silverlight, 417-418

MediaElement element, 418**members, hiding CLR, 487****memory management, .NET Framework, 19****messages**

- dynamic, 139
- receiving, 283
- sending, 283

metadata, 16**metaprogramming, 9-10****method_missing method, 38, 117****methods, 222-227**

- abstract, 246-247
- accept_verbs, 374
- add_EventName, 253
- add_log, 456
- add_one, 88
- after, 443
- alias_action, 375
- aliasing, 499
- alias keyword, 91-92
- Application.run, 305
- Array class, 56-57
- assert, unit testing, 428-431
- before, 443
- calling, 45-46
- Class class, 232-233
- classes, 109-111
 - overriding, 122
 - undefining, 491
- clr_constructor, 232
- clr_ctor, 232
- clr_member, 231
- clr_members, 233
- clr_new, 233
- code standards, 47
- configuration, 235
- const-missing, 118
- constructors, 497
- defining, 88-89, 95-96
- describe, 438

- each, 59, 116
- Exception class, 78
- expectation, RSpec, 439-442
- extensions, 491-500
- File.open, 170
- find_name, 308
- get_products, 276
- globals, 235
- Hash class, 58-59
- helper. See helper methods
- hiding, 500
- include, 215
- indexers, 224-225
- Initialize, 498-499
- initialize, 102
- instances, 222
- invoking, 178-180
- IronRuby class, 235-236
- join, 162
- listen, 283
- load, 236
- load_assembly, 501
- load_component, 412
- loaded_assemblies, 235
- loaded_scripts, 236
- load_root_visual, 412
- method-missing, 117
- method_missing, 38
- Module, 108
- module-contained objects, 127
- multiple overloads, 249-250
- multiply, 254
- naming, 90
- NavigationWindow, 314-315
- new, 161
- non_action, 374
- Object class, 231-232
- objects, associating, 94-95
- open, 171
- overload, 234
- overloaded, 223
- overriding, 121-122, 245-251
- overview of, 87-88
- parameters, 495
- parse, 307
- print, 48
- printf, 52-53
- private, 223
 - overriding, 122
 - testing, 428
- public, 222
- regular, 247-248
- remove_EventName, 253
- require, 131, 207, 236
- run, 169, 300
- sealed, 250-251
- selectors, 374
- setter, 117
- setup, unit testing, 432-433
- singleton, 96
 - special, 115-118
 - special IronRuby, 231-236
 - special .NET, 225
 - special parameters, 493
 - start, 161
 - static, 223, 248-249
 - String class, 54, 234-235
 - String.new, 51
 - succ, 116
 - teardown, unit testing, 432-433
 - the_next_big_thing, 92
 - to_clr_type, 231
 - undef, 96
 - values, returning from, 90-91
 - virtual, 245-246
 - visibility control, 118-120
- Microsoft Intermediate Language (MSIL). See CIL**
- Microsoft Public License (MS-PL), 22**
- Microsoft SQL Server. See SQL Server**
- MinimizeBox property, 287**
- mixins**
 - applying, 254-255
 - built-in, 488
 - definitions, 487-488
 - modules, 128-129
- models**
 - ActiveRecord, 340-341
 - ASPNet MVC validations, 396
 - creating, 343
 - hosts, 22, 23
 - MVC, 368-371
- model-view-controller. See MVC**
- modes**
 - consoles, REPL, 31
 - private binding, 213-214
- modifier statements, 80**
- modifying**
 - databases, 335
 - loop flow, 74
 - print separators, 172
 - visibility control, 118-120
 - words, 357
- Module method, 108**
- ModuleRestrictions enum values, 486-487**
- modules, 126-129**
 - classes, 126
 - code standards, 47
 - mixins, 128-129
 - module-contained objects, 126-127
 - namespaces, 127-128, 256
 - .NET extensions, 482-488
- monitor library, 133**
 - references, 144
- monitors, 168-169**
- Mono, 17**
- MRI (Matz's Ruby Interpreter), 6**
- MSIL (Microsoft Intermediate Language). See CIL**
- MS-PL (Microsoft Public License), 22**
- multilanguage, Cucumber, 456-457**
- multiline comments, 43**

multiple overloads, methods with, 249-250
 multiple-parameter replacement, 89
 multiple statements, writing, 44
 multiply method, 254
Mutex class, 167-168
Mutex_m library, 133
MVC (model-view-controller), 339, 368-385
ASP.NET MVC. See ASP.NET MVC
 controllers, 371-372
 models, 368-371
 views, 378-385
My Computer, 27
MySQL, 260
 connections, opening, 268
 connectors, 260
 contacting, 265-272
 databases, preparing, 266
 design, 272-276
 records
 deleting, 269
 inserting, 269
MySQLAccessor class, 272
mysql.rb files, 270-272

N

Name property, 483, 484
namespaces
 aliasing, 214
 mapping, 256
 modules, 127-128
 .NET objects, 214-215
 opening, 256
 XAML, 306
naming
 assemblies, 18
 code, 212
 conventions
 Ruby, 484
 unit testing, 427-428
 methods, 90
navigating environments, RoR, 342-346
NavigationWindow, 314-315
nested test suites, 434
.NET, 1
 assemblies
 loading, 207-210
 WinForms, 296
 code
 mapping, 210-214
 standards, 211-213
 events, 218-221
 extensions
 applying, 509-510
 creating, 478-501
 modules, 482-488

Framework
 downloading, 26
 features, 16-20
 frameworks, 20
 history of, 13-14
 memory management, 19
 overview of, 11-13, 15-16
 security, 19
 versions, 14
 interoperability, 203
 objects, applying, 214-231
 special methods, 225

NetBeans, 35-36
Net/ftp library, 133
Net/ftptls library, 133
Net/http library, 133
 references, 144
Net/imap library, 133
Net/pop library, 133
Net/smtp library, 133
Net/telnet library, 133
Net/telnets library, 133
 new method, 161
New Project dialog box, 366
Next Generation Windows Services. See NGWS
 next keyword, 75
NGWS (Next Generation Windows Services), 14
non_action method, 374
Notepad++, 37
Novell, 14
numbers
 loops, 73-74
 Ruby, 48-50
Numerology.Calculator class, 426

O

Object class
 methods, 231-232
 opening, 255-256
object-oriented languages, 7
Object Rational Mapping. See ORM
objects
 ActionExecutingContext, 388
 AuthorizationContext, 392-393
 Behavior, 438
 CLR
 applying Recorder class on, 38
 reflection, 237
 enumerable, 72-73
 investigating, 177-178
 IronRuby, applying, 470-471
 living, finding, 176-177
 mapping, 211
 methods, associating, 94-95
 module-contained, 126-127

- .NET, applying, 214-231
- procs, defining, 97
- receiving exception, 79
- Ruby, 237
- ObjectSpace module, 176**
- Observer library, 133**
 - references, 145
- observer patterns, 194-196**
- onerror parameter, 407**
- opening**
 - CLR classes, 254-256
 - connections
 - MySQL, 268
 - SQL Server, 262
 - namespaces, 256
 - Object class, 255-256
- Open3 library, 134**
- open method, 171**
- open source projects, 6**
- OpenSSL library, 134**
- open-uri library references, 145**
- operating systems, 14**
- operations, files, 175**
- operators, 16**
 - AND, 65
 - arithmetic, 49, 112
 - array access ([]), 112, 114
 - array access setter ([] =), 112, 114
 - bitwise, 112
 - Boolean AND (&&), 65
 - Boolean OR (||), 65
 - case equality (===), 65, 112, 113
 - comparison, 64-65, 112, 114
 - equality (==), 112
 - equal to (==), 65
 - general comparison (<=>), 65
 - greater than/greater than or equal to (>, >=), 65
 - less than/less than or equal to (<, <=), 65
 - no pattern match (!), 65
 - not equal to (!=), 65
 - OR, 65
 - order comparison (< <= => >), 112
 - overloading, 111-115
 - pattern match, 65, 112
 - precedence, 64
 - shifting, 113
 - shift-left (<<), 112
 - shift-right (>>), 112
 - ternary, 69
 - unary, 113
 - unary minus (-@), 112
 - unary plus (+@), 112
- options, installation, 26**
- Options property, 463**
- Optparse library, 134**
- ORM (Object Rational Mapping), 341**
- OR operator, 65**
- or property, 287**
- OStruct library, 134**
- outline scenarios, 451-452**
- out parameter, 226**

- overloading**
 - methods, 223
 - operators, 111-115
- overload method, 234**
- override keyword, 245**
- overriding**
 - class methods, 122
 - events, 253-254
 - methods, 121-122, 245-251
 - private methods, 122
 - properties, 251-253

P

- pages. See web pages**
- panels, websites, 321**
- parameters**
 - default values, 92
 - Inherits, 489-490
 - initParams, 407
 - load_assembly method, 209
 - methods, 495
 - multiple-parameter replacement, 89
 - onerror, 407
 - out, 226
 - params, 227
 - ref, 226-227
 - scope, 466
 - source, 407
 - special methods, 493
 - special types, 93-94
- params parameter, 227**
- parentheses (), 45**
- ParseDate library, 134**
- parse method, 307**
- Parse_Tree library, 134**
- Pascal casing in .NET code, 212**
- passing**
 - data between windows, 312-314
 - data in and out of threads, 164
 - filenames, 208
 - variables to and from IronRuby, 469-470
- pass method, 165**
- PATH Environment Variable, configuring, 29**
- PathName library, 134**
- paths**
 - full, loading .NET assemblies, 208
 - searching, 464
- PATH variable, 332**
- pattern match operator, 65, 112**
- patterns, design, 186-202**
 - builder, 196-199
 - command, 190-192
 - iterator, 188-190
 - observer, 194-196
 - singleton, 192-194
 - strategy, 186-188
- Perl, 1, 5**

ping library, 134
 references, 147
positioning
 arguments with default values, 92
 block arguments, 94
posting form data, 145
PP library, 134
precedence, operators, 64
preparing
 environments
 ASP.NET MVC, 363-365
 for data access, 260-259
 RoR, 331-332
 Silverlight, 402
 MySQL databases, 266
PrettyPrint library, 134
previous layout lists, applying, 358-360
printf method, 52-53
-print method, 48
print method, modifying separators, 172
priority, threads, 164-165
private binding mode, 213-214
PrivateBinding property, 463
private methods, 223
 overriding, 122
 testing, 428
procs, 97-99, 496
 flow, 100-101
-profile, 33
programming
 interfaces, Ruby, 506-508
 languages, 13. *See also* languages
 metaprogramming, 9-10
programs, Hello World!, 48
progress_proc, 146
project files, creating, 333
projects
 extensions, 481
 structures, Cucumber, 445-446
 Visual Studio, creating extensions, 502
properties
 CLR, 228-229
 files, accessing, 173-174
 forms, formatting, 287-289
 NavigationWindow, 314-315
 overriding, 251-253
 redefining, 252
 RubyClassAttribute class, 488-489
 RubyMethodAttribute class, 492
 RubyModuleAttribute, 483-484
 ScriptRuntimeSetup class, 463
proxies, 146
PStore library, 134
Public folder, 334
public/images folder, 334
public/javascripts folder, 334
public key tokens, assemblies, 18
public methods, 222
public/stylesheets folder, 334

Q

queries
 databases, 263-264, 268
 LINQ, 279-280
queues, 168

R

Racc/parser library, 134
RadRails, 37
raise_error RSpec expression matcher, 441
raising exceptions, 83-85
Rake, 184-185
rakefiles, 184-185
ranges
 arrays, converting, 59
 loops, 74
 Ruby, 59-60
rational library, 134
 references, 152
Readbytes library, 134
read-evaluate-print loop. See REPL
reading
 code, 467
 files, 158, 170-172
 XML documents, 154
receiving
 exception objects, 79
 messages, 283
Recorder class, implementation, 38-39
records, MySQL
 deleting, 269
 inserting, 269
redefining properties, 252
redirect results, 373-374
redo keyword, 75
references
 assemblies, adding, 367
 standard libraries, 135-158
reflection, 176
 CLR objects, 237
 living objects, finding, 176-177
 methods, invoking, 178-180
 objects, investigating, 177-178
 variables, configuring dynamically, 178-180
ref parameter, 226-227
Refresh button, adding, 360-361
regular classes, 239-242
regular expressions, 60-62
regular methods, 247-248
remove_EventName method, 253
removing assemblies, 366
replacing
 multiple-parameters, 89
 Silverlight content, 412

REPL (read-evaluate-print-loop), 10-11

- console mode, 31
- WPF, 329-330

reportErrors key, 408**Representational State Transfer. See REST requirements**

- assemblies, Chat class, 282
- libraries, RSpec, 436-437
- .NET Framework assemblies, 208

require method, 131, 207, 236**rescue statement, 78-80****ResizeMode values, 311-312****resources**

- deleting, 345
- panels, 321
- standard libraries, 159
- web pages, generating, 354-355

respond_to RSpec expression matcher, 442**REST (Representational State Transfer), 339-340****Restrictions property, 484, 486****results**

- filters, 390-392
- redirects, 373-374
- views, 372-373

retrieving

- pages, 144
- Silverlight elements, 412-414
- WPF elements, 308

retry keyword, 81**return keyword, 90, 99****return values**

- controller actions, 371
- methods, 90-91
- threads, 162

Rexml library, 134

- references, 153

-r "library," 33**Root folder, 30****root-visual property, 412****RoR (Ruby on Rails), 330-331**

- applications, creating, 332-337
- components, 340-342
- database configuration, 334-337
- environments
 - navigating, 342-346
 - preparing, 331-332
- folders, 333-334
- guidelines, 339-331
- servers, running, 337
- web pages
 - creating, 346-354
 - formatting database-driven pages, 354-361

rotating automatic logs, 144**routes**

- ASP.NET MVC, 385-387
- customizing, 386-387
- RoR, 341
- URLs, 371

RSpec, 435-444

- behavior, creating, 438
- code, injecting, 442-444

examples, 439

- expectation methods, 439-442
- expression matchers, 441-442
- installing, 436
- library requirements, 436-437
- running, 437

RSS library, 134**Rubinius, 6****Ruby**

- accessors, 107-109
- arrays, 54-57
- blocks, 96-97
- classes, 101-126
- code, naming, 212
- code-containing structures, 86
- constants, 63-64
- control structures, 64-77
- dates and times, 62-63
- features, 6-11
- Hello World!, 48
- history of, 5-6
- implementation, 6
- Lambdas, 99-100
- naming conventions, 484
- .NET, loading assemblies, 207-210
- numbers, 48-50
- objects, 237
- overview of, 2-5, 25-26
- process, 97-99
- programming interfaces, 506-508
- ranges, 59-60
- regular expressions, 60-62
- symbols, 58
- syntax, 43-47
- text, 50-54
- threads, 161-169
- type differences, CLR and, 211
- variables, 48-64

RubyClassAttribute class properties, 488-489**RubyClass parameter, 492****RubyContext parameter, 492****RubyGems, 183**

- gems
 - applying, 183-184
 - finding, 185
 - installing, 183
 - limitations, 185

Ruby in Steel, 34-35**RubyMethodAttribute class properties, 492****RubyMethodAttributes class, 494-495****RubyMine, 36-37****RubyModuleAttribute properties, 483-484****Ruby on Rails. See RoR****Ruby Spec, 6****rules**

- behavior, Cucumber, 443-457
- runtime components, 24

run method, 300

- synchronization, 169

running

- assemblies, 16
- REPL console mode, 31
- RSpec tests, 437
- servers, RoR, 337
- Silverlight applications, 405
- tagged features and scenarios, 454
- unit testing, 434-435
- XAML, 307, 411-412

runtime

- components, 22, 23-24

DLR. See DLR

- ScriptRuntime class, 462-463

S**Samples folder, 30****satisfy RSpec expression matcher, 442****scaffold command, 344, 345****Scanf library, 134****scenarios**

- Cucumber, 447-452
- hooks, 454-455
- outlines, 451-452
- tagging, 453

SciTE, 37**scope parameter, 466****script/destroy command, 345****ScriptEngine class, 23**

- C#/VB.Net, 463-465

Script folder, 334**script/generate command, 343-345****script/generate controller command, 346****ScriptRuntime class, 23**

- C#/VB.Net, 462-463

ScriptRuntimeSetup class, 462**ScriptScope class, 23**

- C#/VB.Net, 465-466

script/server command, 342**ScriptSource class, 23****sealed classes, 243****sealed methods, 250-251****searching**

- gems, 185
- living objects, 176-177
- paths, 464
- standard libraries, 159

security

- CAS, 19
- .NET Framework, 19

selectors, methods, 374**self keyword, 118****semicolon (;), 44****sending messages, 283****servers, running RoR, 337****services, sockets, 149-152****Set library, 134****setter methods, 117****settings. See configuration; formatting****setup method, unit testing, 432-433****shapes, WPF, 322-323****sharing views, 380-382****Shell library, 134****shifting operators, 113****short names, assemblies, 18****Silverlight, 22, 401**

- animation, 417-418
- applications, 402-406
- chr tool, 404-406
- code, adding, 411-415
- controls, 411
- data binding, 419-422
- dynamic data, 420-421
- elements, retrieving, 412-414
- environments, preparing, 402
- event handling, 414
- folders, 30
- graphics, 415-417
- HTML, accessing, 414-415
- layouts, 410-411
- sl tool, 402-403
- static data, 419-420
- templates, 422
- web pages, adding, 406-408
- XAML, 409

single line comments, 43**single-quoted strings, 51****singleton**

- classes, 490
- library, 154
- methods, 96
- patterns, 192-194

Size property, 287**sl tool, 402-403****Smalltalk, 1, 5****socket library, 154****sockets, services, 149-152****source code, 29****source parameter, 407****special argument types, 225****special IronRuby methods, 231-236****special methods, 115-118****special .NET methods, 225****special parameters**

- methods, 493
- types, 93-94

sql.rb files, 264-265**SQL Server, 260**

- connections, opening, 262
- data access, contacting, 260-265
- design, 272-276
- installing, 332

SqlServerAccessor class, applying, 265**Stack** class, 38**StackPanel** control, 317-319, 410**StandardError**, 85

standard libraries, 131

applying, 131

available libraries, 132-135

MVC, 375

references, 135-158

searching, 159

standards

code, 47

.NET code, 211-213

REST, 339-340

starting, 24-25

threads, 161

start key, 408

start method, 161

StartPosition property, 288

startup events, 305

statements

case, 67-69

else, 81-82

if-else, 69

modifiers, 80

rescue, 78-80

writing, 44

yield, 76-77

states, threads, 165-167

static classes, 243

static data

binding to, 325-326

Silverlight, 419-420

static methods, 223, 248-249

steps

hooks, 454-455

implementation, 449-451

Storyboard element, 418

strategy pattern, 186-188

String class, 53

methods, 54, 234-235

String.new method, 51

strings

accessing, 53-54

connections

adding, 262

building, 261, 267

examples of, 261

defining, 50

delimiters, 50

double-quoted, 44-50

IronRuby code, executing from, 468-469

single-quoted, 51

time, helper classes, 349

strong names, loading .NET assemblies, 208-209

structs, inheritance from CLR, 243

structures

applications, WinForms, 282

classes, building, 260, 267

code file, 46-47

directories, 333-334

folders, sl tool, 403

projects, Cucumber, 445-446

Ruby

code-containing, 86

control, 64-77

str variable, 466

styles

Silverlight, 419-420

WPF, 326-327

stylesheets, adding, 350-351

subscribing events, 219-220

substrings, 53. *See also* strings

succ method, 116

suites, test, 433-434

superclass method implementation, invoking, 123

super keyword, 246

suppressing layout events, 286

switches

chr tool command-line, 405

command-line, 31

symbols, Ruby, 58

synchronization, threads, 167-169

syntax

aliasing, 215

comments, 43-44

LINQ, 279-280

Ruby, 43-47

Ruby in Steel, 35

SyntaxError, 80

System.String class, 254**System.Windows.Forms.Application** class, 300

T

tables, examples, 451-452

tags

Cucumber, 453-454

scenarios, hooks, 455

target environments, .NET environments, 482

target machines, executing code on, 16

TCPServer class, 156**TCPSocket** class, 156, 283

teardown method, unit testing, 432-433

templates

directories, sl tool, 403

IronRubyMvs Dll files, adding, 365

Silverlight, 422

WPF, 328-329

ternary operators, 69

Test folder, 334

testing

Cucumber, 443-457

private methods, 428

RSpec, 435-444

unit testing. *See* unit testing

Test::Unit, 427-433

text
 Ruby, 50-54
 words, modifying, 357
TextBlock element, 304
TextBlock.Loaded event, 418
Text property, 288
textual marshaling, 182
the_next_big_thing method, 92
Then step, 448
third-party libraries, MVC, 375
 references, 157
threads, 161-169
 exceptions within, 163
 priority, 164-165
 states, 165-167
 synchronization, 167-169
throw_symbol RSpec expression matcher, 442
time
 Ruby, 62-63
 strings, helper classes, 349
Tmp folder, 334
to_clr_type method, 231
ToDoListModel class, 370
tools, 30-34
 chr, 404-406
 GUI, installing, 260
 irake, 337
 Rake, 184-185
 sl, 402-403
-trace, 33
trees
 expressions, 23, 24
 inheritance, 49
types, 16
 accessors, 107
 declaring, 48
 defining, 44
 differences, CLR and Ruby, 211
 exceptions
 defining, 139
 handling, 80
 special argument, 225
 special parameter, 93-94

U
UDPSocket class, 156
unary operators, 113
undefining class methods, 491
undef methods, 96
uniform resource locators. See URLs
unit testing, 424-425
 assertions, 428-431
 code, 426-427
 running, 434-435
 setup method, 432-433

teardown method, 432-433
 test suites, 433-434
 Test::Unit, 427-433
unless, 67
unsubscribing events, 220
until loops, 70
URLs (uniform resource locators), 371
UTC difference values, modifying, 358-359

V

-v, 33
validations, ASP.NET MVC, 396-398
values
 compatibility available, 485
 CSV, 132
 default parameter, 92
 methods, returning from, 90-91
 numeric, 48-50
 priority threads, 164-165
 ResizeMode, 311-312
 return, threads, 162
 variables, configuring, 44-45
variables
 \$LOAD_PATH, 210
 accessing from outside, 106-107
 classes, 103
 code standards, 47
 configuring dynamically, 178-180
 constants, 63. *See also* constants
 inside classes, 102-107
 instances, 103-104, 453
 local, 105
 PATH, 332
 Ruby, 48-64
 str, 466
 to and from IronRuby, passing, 469-470
 values, configuring, 44-45
VB.Net. See C#/VB.Net
Vendor folder, 334
versions
 assemblies, 18
 .NET Framework, 13, 14
 Ruby, release dates of, 6
vertical bars (|), 96
VES (Virtual Execution System), 17
video, adding, 418
viewing
 data templates, 423
 generated index pages, 356-357
 windows, 312
views
 ActionView, 341
 applications, creating, 382-385
 ASP.NET MVC validations, 397-398
 creating, 346-349

- helper methods, 380
- MVC, 339, 378-385
- results, 372-373
- sharing, 380-382

Virtual Execution System. See VES

virtual methods, 245-246

visibility

- control, 118-120
- initializing, 102

Visual Studio, 35

- designer, applying, 295-296
- extensions, creating projects, 502
- IronRubyMvs Dll files, adding, 365

W

-w, 33

web pages

- creating, 346-354
- database-driven pages, formatting, 354-361
- resources, generating, 354-355
- Silverlight, adding, 406-408
- references, 157

websites

- panels, 321
- standard libraries, 159

When step, 448

while loops, 70

whitespaces, 45

Window attribute, 309-310

windows

- content, 315-317
- NavigationWindow, 314-315
- passing data between, 312-314
- viewing, 312
- WPF, 309-310

Windows Forms. See WinForms

Windows operating system, 14

Windows Presentation Foundation. See WPF

WindowStyle attribute, 310-311

WinForms, 281

- assemblies, loading, 285
- chat, building, 285-299
- Chat class, building, 282-285
- controls, adding, 289-293
- execution code, writing, 300-301
- functionality, adding, 293-295
- overview of, 281-282

-W[level], 33

words, modifying, 357

words, Cucumber, 456

WPF (Windows Presentation Foundation), 301

- animation, 324-325
- brushes, 322-324
- content, 315-317
- data binding, 325-329
- elements, retrieving, 308

- events, handling, 308-309
- graphics, 321-325
- IronRuby fundamentals, 307-309
- layout controls, 317-321
- overview of, 303-305
- REPL, 329-330
- shapes, 322-323
- styles, 326-327
- templates, 328-329
- windows, 309-310
- XAML, 305-307

WrapPanel control, 318-319

writing

- code, 504-506
- execution code, 300-301
- files, 158, 172-173
- Hello World!, 48
- statements, 44
- tests, 427

X

XAML (eXtensible Application Markup Language), 305-307

- animation, 417-418
- elements, accessing, 412-414
- graphics, 415-417
- running, 307, 411-412
- Silverlight, 409

-X:ColorfulConsole, 32

-X:CompilationThreshold, 34

-X:ExceptionDetail, 34

XML (eXtensible Markup Language) documents

- generating, 153
- reading, 154

-X:NoAdaptiveCompilation, 34

-X:PassExceptions, 34

-X:PrivateBinding, 34

XRuby, 6

-X:ShowClrExceptions, 34

Y-Z

YAML library, 135

- references, 157
- textual marshaling, 182

yield keyword, 94

yield statement, 76-77

ZIP files, 27

Zlib library, 135, 158