FREE eBook

LEARNING rubygems

Free unaffiliated eBook created from
Stack Overflow contributors.

#rubygems

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: rubygems

It is an unofficial and free rubygems ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official rubygems.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with rubygems

Remarks

Gems can be used to extend or modify functionality in Ruby applications. Commonly they're used to distribute reusable functionality that is shared with other Rubyists for use in their applications and libraries. Some gems provide command line utilities to help automate tasks and speed up your work.

The RubyGems software allows you to easily download, install and use Ruby software packages on your system. The software package is called a `gem` and contains a packaged Ruby application or library.

Examples

Installation or Setup

If you don't have any *RubyGems* installed, there is still the pre-gem approach to getting software, doing it manually:

- Download from RubyGems
- Unpack into a directory and `cd` there
- Install with: `ruby setup.rb` *(you may need admin/root privilege)*

  ```
  sudo ruby setup.rb
  ```

For more details and other options, see:

```
ruby setup.rb --help
```

Installation on Linux

Make sure you have ruby installed before installing rubygems, and then:

Installing RubyGems Using apt-get on Ubuntu

```
sudo apt-get install rubygems
```

Installing RubyGems Using yum

```
sudo yum install rubygems
```

Manual Installation Method

```
wget https://rubygems.org/rubygems/rubygems-2.6.6.tgz
tar xvf rubygems-2.6.6.tgz
cd rubygems-2.6.6
sudo ruby setup.rb
```

Install specific precompiled ruby version and ruby gems (Ubuntu)

```
sudo apt-add-repository ppa:brightbox/ruby-ng
```

Hit `Enter` to confirm

```
sudo apt-get update
```

Then you can install your ruby version of choice (the ppa supports `ruby2.0` `ruby2.1` `ruby2.2` `ruby2.3` and legacy versions `ruby1.8` `ruby1.9.1`) Don't forget to include the respective `-dev` package for your version. Otherwise the development of native extensions such as `Nokogiri` or `mysql2` will fail.

```
sudo apt-get install ruby2.3 ruby2.3-dev ruby-switch
```

```
sudo ruby-switch set ruby2.3
```

```
ruby -v
```

```
>> ruby 2.3.1p112 (2016-04-26) [x86_64-linux-gnu]
```

Now you can install any desired gem systemwide for all users via `sudo gem install gemname`.

Note:

- This method installs and sets ruby and rubygems system-wide for all users
- Requires sudo to install gems (`sudo gem install rails`) but don't run `bundle` as root. If it needs to install gems it will ask for the sudo password.

Change gem source

```
# list gem sources:
gem sources -l
# remove default gem source:
gem sources -r https://rubygems.org/
# add other gem sources:
gem sources -a https://ruby.taobao.org/
```

Read Getting started with rubygems online: https://riptutorial.com/rubygems/topic/1039/getting-started-with-rubygems

Chapter 2: Anemone

Syntax

- Use Anemone::Core.new(url, options) to initialize the crawler
- Use on_every_page block to run code on every page visited
- Use .run method to start the crawl. No code beforehand will actually start any GET calls.

Parameters

Parameter	Details
url	URL (including protocol to be crawled)
options	optional hash, see all options here

Remarks

- The crawler will by only visit links that are on the same domain as the starting URL. This is important to know when dealing with content subdomains such as `media.domain.com` since they will be ignored when crawling `domain.com`
- The crawler is HTTP / HTTPS aware and will by default stay on the initial protocol and not visit other links on the same domain
- The `page` object in the `on_every_page` block above has a `.doc` method which returns the Nokogiri document for the HTML body of the page. This means you can use Nokogiri selectors inside the `on_every_page` block such as `page.doc.css('div#id')`
- Other information to start can be found here

Examples

Basic Site Crawl

```
pages = []
crawler = Anemone::Core.new(url, options)
crawler.on_every_page do |page|
  results << page.url
end
crawler.run
```

Read Anemone online: https://riptutorial.com/rubygems/topic/6445/anemone

Chapter 3: Change gem source in Gemfile

Examples

Use github repo for specific gem in Gemfile

Instead of

```
gem 'rails'
```

You can specify a github user/repo combination with

```
gem 'rails', github: 'rails/rails'
```

Make use of ref, branch, git for specific gem in gemfile

```
gem 'any gem',git: 'any repo',branch: 'specific branch of that repo',ref: 'reference no.'
```

ref specifies individual commit. **branch** specifies the git branch to pull from.

Read Change gem source in Gemfile online: https://riptutorial.com/rubygems/topic/6356/change-gem-source-in-gemfile

Chapter 4: JSON

Introduction

The `json` gem for Ruby allows for the parsing and creation of JSON.

Syntax

- JSON.parse(json_document_string) => returns a Hash of the JSON document
- JSON.generate(ruby_hash) => returns a JSON document in the form of a String

Parameters

Parameter	Details
json_document_string	A JSON document in the form of a String
ruby_hash	Any Hash object

Examples

Hash to JSON

```
require 'json'
data = {"test" => 123}
puts JSON.generate(data)
```

JSON to Hash

```
require 'json'
document = "{\"test\":123}"
puts JSON.parse(document)
```

Alternate JSON to Hash

```
require 'json'
data = JSON '{"test":23}'  # => {"test"=>23}
```

or

```
require 'json'
data = JSON['{"test":23}'] # => {"test"=>23}
```

Alternate Hash to JSON

```
require 'json'
document = JSON 'test'  => 23 # => "{\"test\":23}"
```

or

```
require 'json'
document = JSON['test' => 23] # => "{\"test\":23}"
```

Read JSON online: https://riptutorial.com/rubygems/topic/8894/json

Chapter 5: Maruku

Introduction

Maruku is a Markdown interpreter for Ruby. It allows for exporting Markdown to HTML and PDF.

Syntax

- Maruku.new(str) => parses Markdown into a Maruku object
- Maruku#to_html_document => returns the Maruku object as a full HTML document (as a string)

Parameters

Parameter	Usage
str	Markdown string to be parsed

Examples

Reading Markdown into Maruku

```
require 'maruku'

str = "
    # Title
    * List 1
    * List 2
    * List 3
"
markdown = Maruku.new(str)
puts markdown.inspect
```

Markdown to HTML

```
require 'maruku'

str = "
    # Title
    * List 1
    * List 2
    * List 3
"
markdown = Maruku.new(str)
puts markdown.to_html_document
```

Read Maruku online: https://riptutorial.com/rubygems/topic/8893/maruku

Credits

S. No	Chapters	Contributors
1	Getting started with rubygems	Community, halfelf, Killerpixler, Lahiru, mlabarca
2	Anemone	Killerpixler, Punknoodles
3	Change gem source in Gemfile	Divya Sharma, giniouxe, Killerpixler
4	JSON	Punknoodles
5	Maruku	Punknoodles