

---

# **cs263 Class Project Documentation**

***Release 0.5***

**Don Voita**

April 28, 2013



# CONTENTS

<b>1</b>	<b>Resources</b>	<b>3</b>
<b>2</b>	<b>Installation Notes</b>	<b>5</b>
<b>3</b>	<b>Running the Examples</b>	<b>7</b>



## Webbots with Ruby and Mechanize



# RESOURCES

- Video Tutorial: TBA
- Class Presentation: TBA
- Mechanize: <http://mechanize.rubyforge.org>
- Nokogiri: <http://nokogiri.org>
- Ruby enVironment Manager(RVM): <https://rvm.io/>





# INSTALLATION NOTES

## Software Used

- OS Fedora 18
- Ruby 1.9.3p392 (2013-02-22 revision 39386) [x86\_64-linux]
- RVM (1.19.6 stable)
- Mechanize (2.6.0)
- Nokogiri (1.5.9)

To install Ruby, you can use your system's Package Management tool. I am using Fedora 18 Linux, so am using Yum. The command looks like this:

```
$ sudo yum install ruby ruby-devel gcc libxml2 libxml2-devel libxslt libxslt-devel
```

To make working with Ruby easier, I recommend using RVM, the Ruby enVironment Manager. RVM is a great tool for managing multiple Ruby environments. With RVM it is easy to setup different projects to use different versions of Ruby and different gem sets. I will not go into great detail here, as using RVM is not necessary to try my examples. But I will cover how to get up and running with or without RVM.

First, without RVM, you can simply use your system's Package Manager to do the following:

```
$ sudo yum install rubygem-mechanize rubygem-nokogiri rubygem-mail
```

All done.

So why bother with RVM? If you only plan on using one version of Ruby and one set of gems, you are fine. However, say you are also developing another application on the same system and it happens to use a different version of Ruby. Now you have the kind of problem that RVM was designed to solve. To install it, do the following. You will need sudo access on the machine, (so this will not work for you on machines in CSIL.):

```
$ \curl -#L https://get.rvm.io | bash -s stable --autolibs=3 --ruby
```

Once this is done, you will need to do:

```
$ source /local/don/.rvm/scripts/rvm
$ type rvm | head -1      # The result should show 'rvm is a function'
```

I added the following to my *.bashrc* file to source *~/.rvm/scripts/rvm* when I login:

```
[[ -s "$HOME/.rvm/scripts/rvm" ]] && . "$HOME/.rvm/scripts/rvm" # Load RVM function
```

If everything works to this point, then doing the following will list which Rubies you have available:

```
$ rvm list
```

On my machine, I do not see version 1.9.3, but that is the version I want to use. So I run:

```
$ rvm install 1.9.3
$ rvm use 1.9.3
```

Now you are ready to install the Mechanize and Nokogiri gems. If you did not know their exact names, you can list any remotely available gems by running:

```
$ gem list --remote
```

Now do:

```
$ gem install mechanize
$ gem install nokogiri
$ gem install mail      # Used in one of my examples.
```

Now you are ready to use Ruby, Mechanize, and Nokogiri. In one of my example scripts, I use the Mail gem to add the ability to send the results of a web query via email, so I added it for good measure.

Note, if you have problems installing these gems, it could be due to missing header files on your system. To resolve these problems, it is usually a matter of installing the packages that supply the header files and trying again. As usual, Google is your best bet.

# RUNNING THE EXAMPLES

To run the examples, copy them to your machine. You can either do:

```
$ ruby example.rb
```

Or:

```
$ chmod 750 example.rb  
$ ./example.rb
```

It is really a matter of choice.