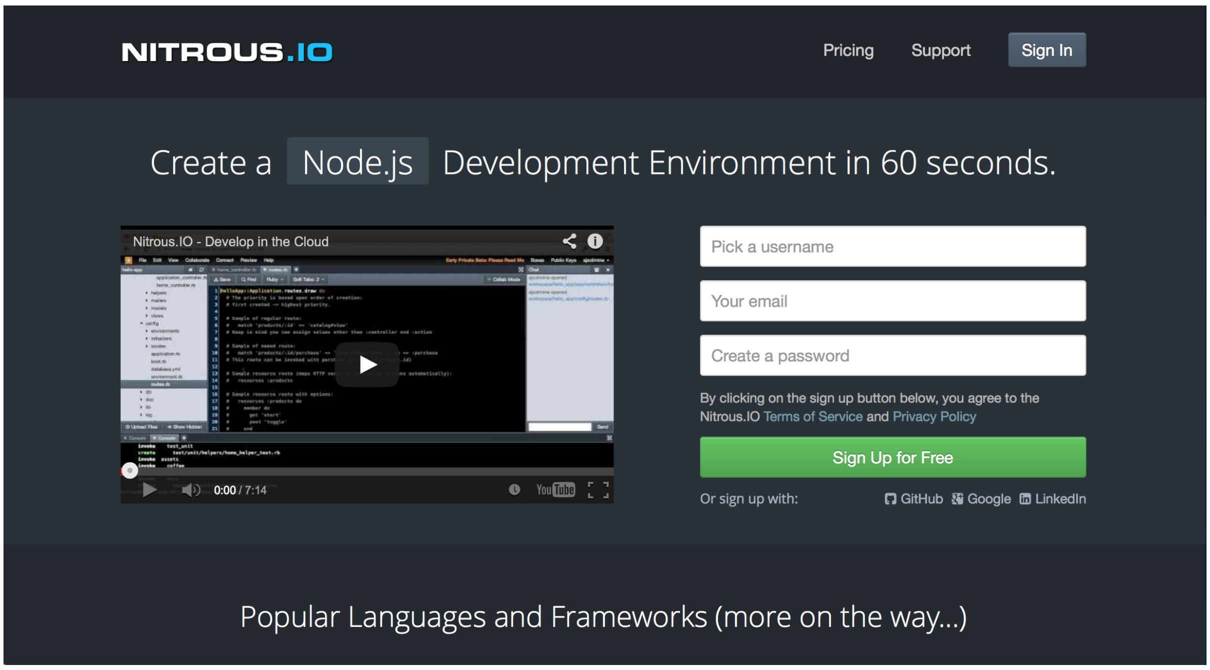
Nitrous.io

Signing Up

Open up your browser and go to Nitrous.io.

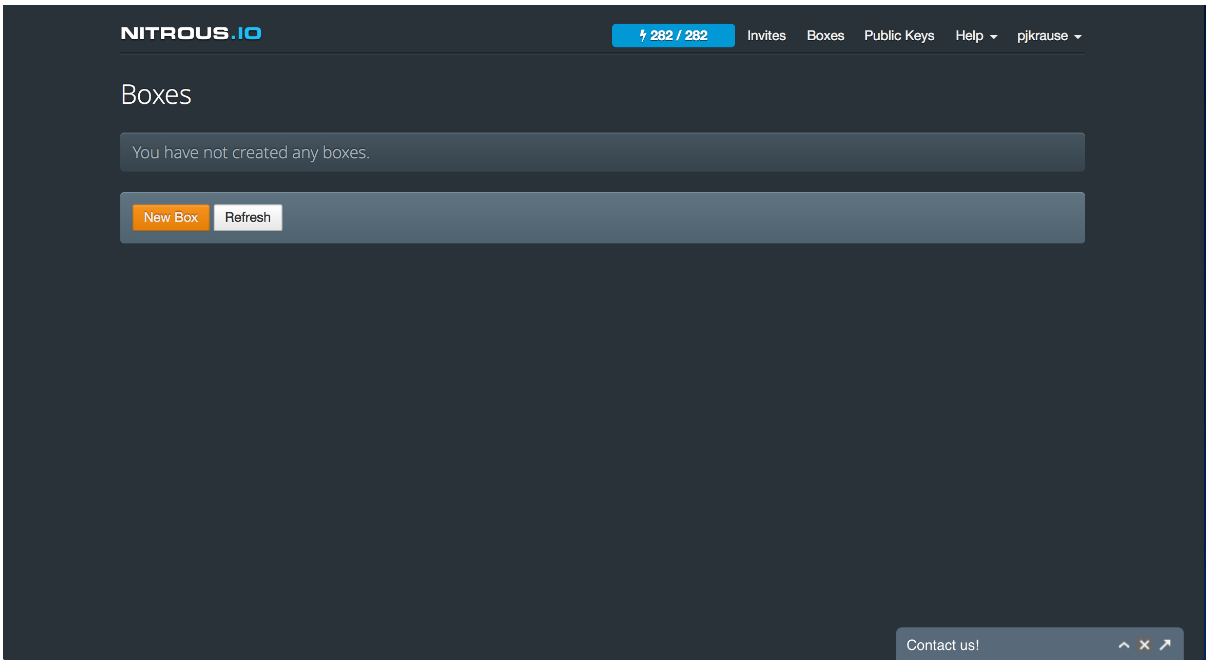

NITROUS.IO
Pricing
Support
Sign In
Create a Node.js Development Environment in 60 seconds.
Nitrous.IO - Develop in the Cloud
Pick a username
Your email
Create a password
By clicking on the sign up button below, you agree to the Nitrous.IO Terms of Service and Privacy Policy
Sign Up for Free
Or sign up with:
GitHub
Google
LinkedIn
Popular Languages and Frameworks (more on the way...)

Click on "Sign Up for Free" and follow the instructions.

Creating a Box

Nitrous supports many different programming languages. We are going to work with Ruby and Rails. Even more exciting, we are going to learn how to build web applications using Rails.

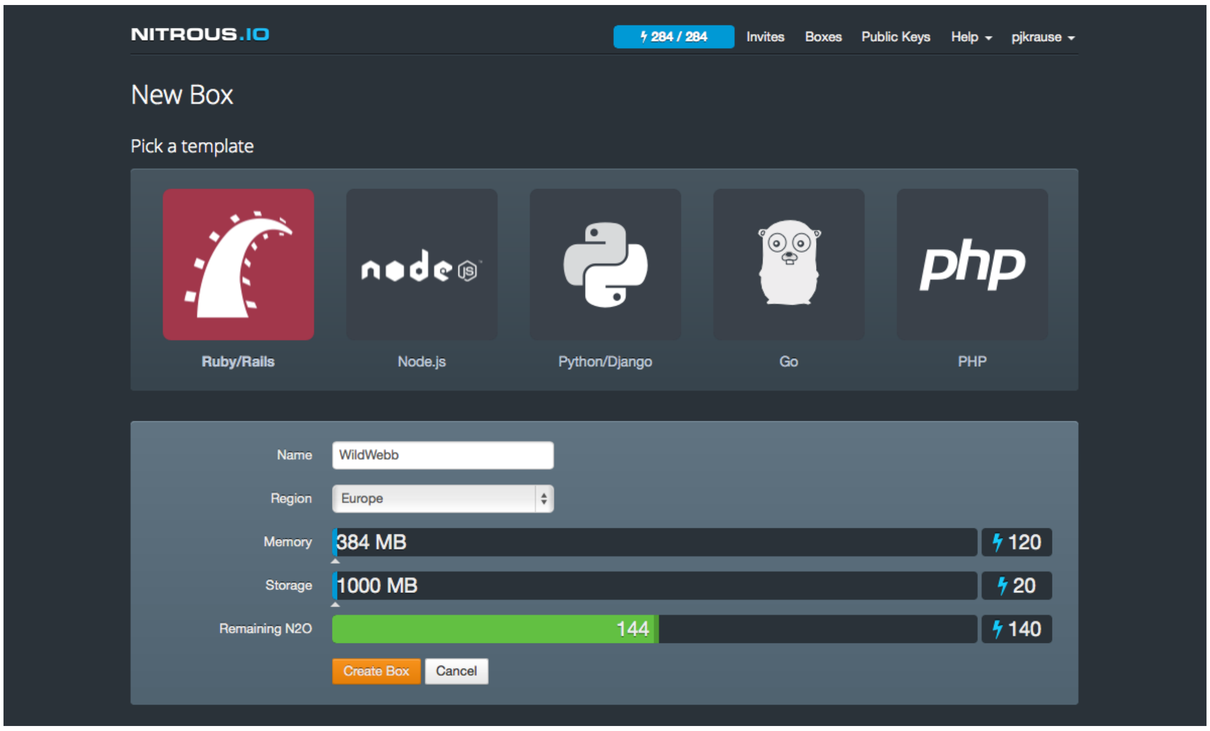
Once you have signed up and logged in, you will see a screen like the one on the next page:



Boxes

You have not created any boxes.

Your free account will give you enough Nitrous to create just one box, but this will be sufficient for this course. So go ahead and click on "New Box"!



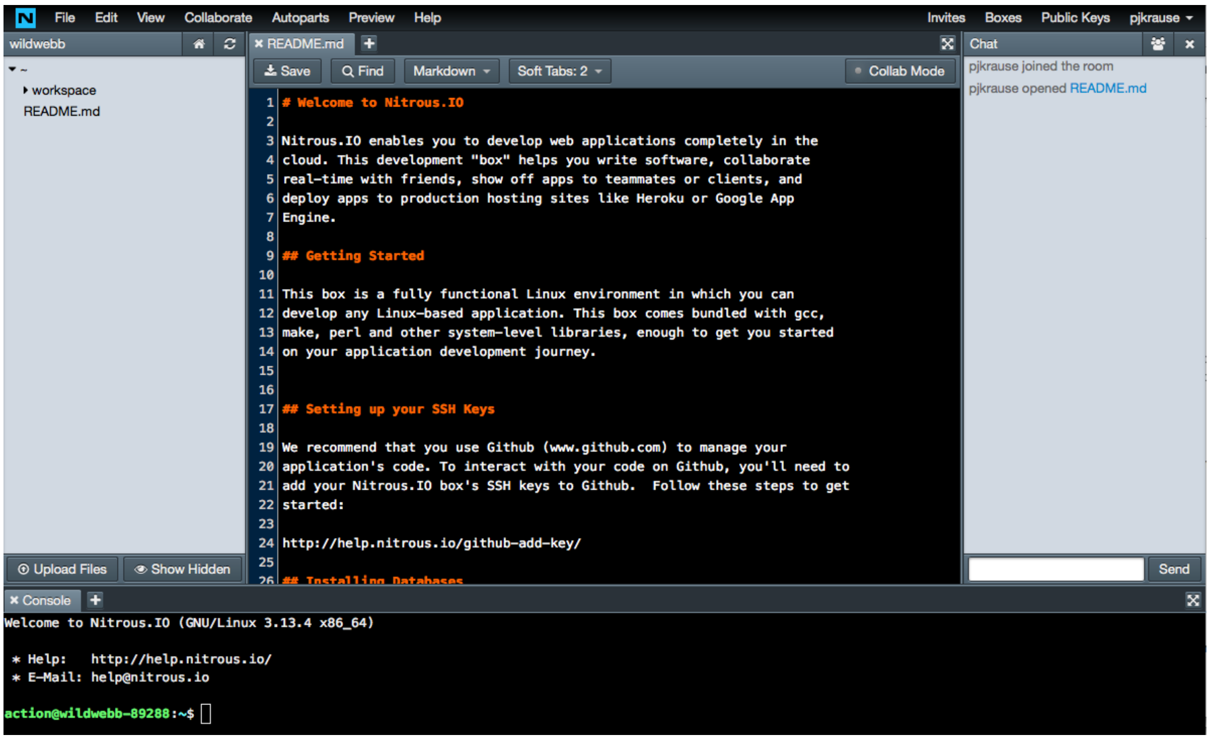
New Box

Pick a template

Name: WildWebb
Region: Europe
Memory: 384 MB — 120
Storage: 1000 MB — 20
Remaining N2O: 144 — 140

You can give your box a name. You can also select the region you live in so Nitrous will be running on a server near to you. But most of all, make sure you click on the BIG Ruby/Rails button so that Nitrous knows to set up all the things you need!

Once you have done that, hit "Create Box" and you will see this:



```
# Welcome to Nitrous.IO

Nitrous.IO enables you to develop web applications completely in the
cloud. This development "box" helps you write software, collaborate
real-time with friends, show off apps to teammates or clients, and
deploy apps to production hosting sites like Heroku or Google App
Engine.

## Getting Started

This box is a fully functional Linux environment in which you can
develop any Linux-based application. This box comes bundled with gcc,
make, perl and other system-level libraries, enough to get you started
on your application development journey.


## Setting up your SSH Keys

We recommend that you use Github (www.github.com) to manage your
application's code. To interact with your code on Github, you'll need to
add your Nitrous.IO box's SSH keys to Github.  Follow these steps to get
started:

http://help.nitrous.io/github-add-key/

## Installing Databases
```

pjkrause joined the room
pjkrause opened README.md

```
Welcome to Nitrous.IO (GNU/Linux 3.13.4 x86_64)

 * Help:   http://help.nitrous.io/
 * E-Mail: help@nitrous.io

action@wildwebb-89288:~$
```

How cool is that! Nitrous has set you up with a complete desktop in your browser with a finder space on the left side where you can see all your files and folders (although we only have one folder called "workspace" just now), a big area for editing your code, a console along the bottom where you can run your code and do all sorts of exciting things. There is even a chat room where you can talk to your friends while you code!

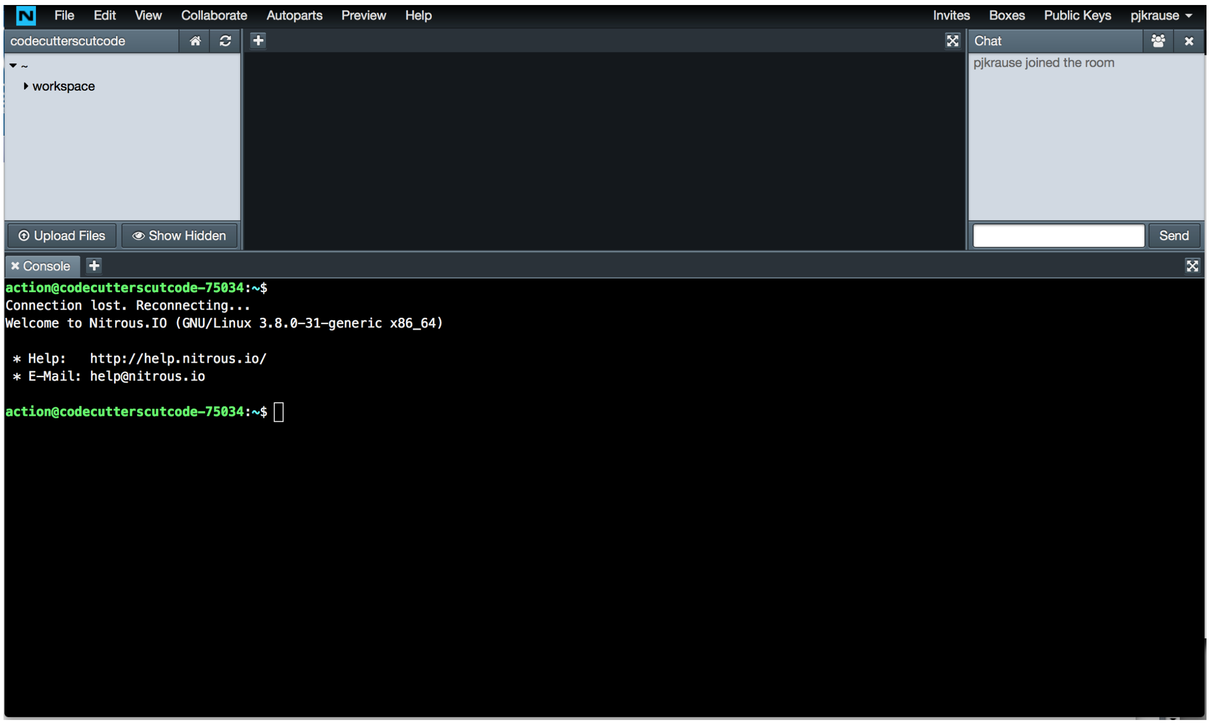
There's loads of stuff in there and you don't need to do a thing to keep it all up to date - Nitrous will do that for you. This is one of the BIG advantages of "working in the cloud".

Working at the Console

Linux is there to help you

Any computer program needs to run inside an operating system. That operating system acts as an environment for the program to live in, so that it can communicate with files, printers, mouse and keyboard, for example. If we are aiming to build a web application, then we really need an operating system that is good at multi-tasking. There are lots of things that it needs to do at the same time - it might be saving some date to a file file for one user, while doing some sums for another, and waiting for a third one to fill in a form. Unix is a very stable multi-tasking operating system that was developed by some talented computer scientists at Bell Labs in the 1960s. Unfortunately, it is a bit expensive. So instead, we will use Linux, which is an open source version of Unix. The kernel of Linux was first released in 1991 by Linus Torvalds, and it has developed since then into an operating system that is very widely used by the web application development and deployment community.

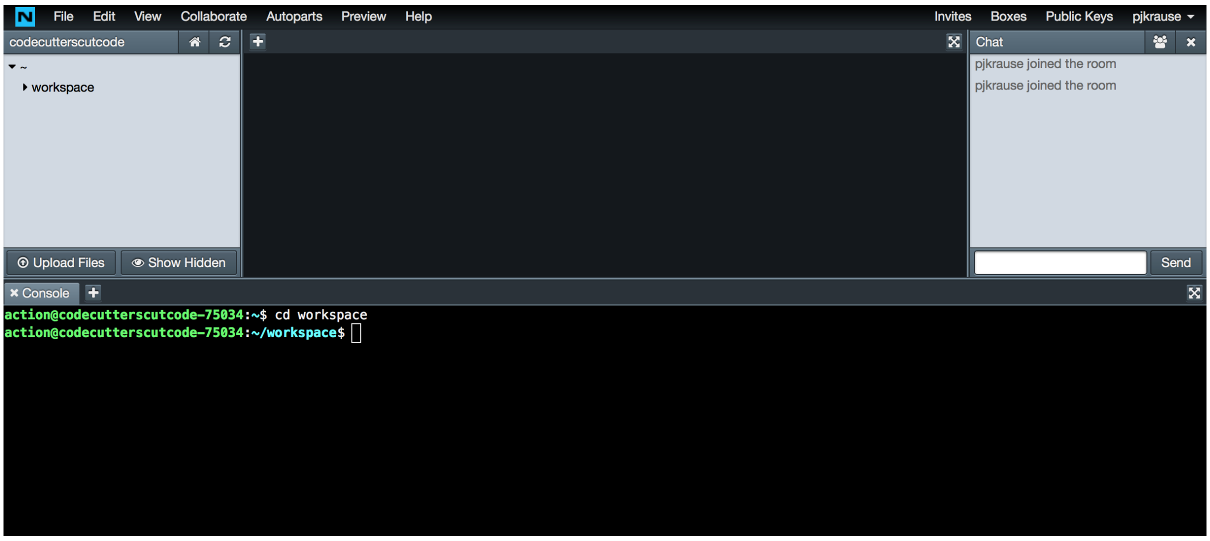
The Console at the bottom of your Nitrous desktop is your window into the Linux operating system. These is where you type in commands to get Linux to do useful things for you. We can make the window a bit bigger, which is worth doing for this chapter and the next as we are not going to use the editor for a little while. Just move the cursor so that it is over the line that divides the editor window from the console and a double sided arrow will appear. Then click on the line and drag it upwards so that your Nitrous desktop looks something like this:



```
action@codecutterscutcode-75034:~$
Connection lost. Reconnecting...
Welcome to Nitrous.IO (GNU/Linux 3.8.0-31-generic x86_64)

 * Help:   http://help.nitrous.io/
 * E-Mail: help@nitrous.io

action@codecutterscutcode-75034:~$
```

Computers try to hide low-level commands for everyday use. However, if you are developing programs then you really need to learn to talk directly to the operating system through the console. It's a little bit harder to get started as you need to learn some commands instead of just clicking on things, but you soon start to remember the commands you need.

Let's try one or two. Nitrous will have started the console session in the "root" folder, or "directory". As you can see from the finder window, you only have one other directory (I am going to stick with the term "directory" instead of folder from now onwards as this is the terminology of professionals!). This is called "workspace" and we will soon be putting some files into it. So, let's move the console to that folder. You use the command "cd" for changing directory (fairly obvious really!). So type this at the console and press the "return" key. You will see that the prompt in the console reminds you of the directory that you have just moved to (see the picture on the next page).



pjkrause joined the room
pjkrause joined the room

```
action@codecutterscutcode-75034:~$ cd workspace
action@codecutterscutcode-75034:~/workspace$
```

Is you add in new directories, you gradually build up a tree-like structure starting from the "root". Actually, it is a bit like an upside-down tree as you will find you think about moving down the tree from root to a sub-directory like "workspace". You can always move up a level by typing:

```
$ cd ..
```

(The dollar sign represents the prompt in the console so you don't actually type that - it is already there). Try it now, and you will end up back in the root directory.

I'll let you into another little secret - Linux can recognise some of the words you might use without you need to type them. This is known as "tab autocompletion". So, for example, in the root directory there is only one sub-directory and this starts with a "w" (for workspace). So if you now type at the prompt

```
$ cd w
```

and before pressing the return key, you press the tab key, Linux will complete this for you so you will see:

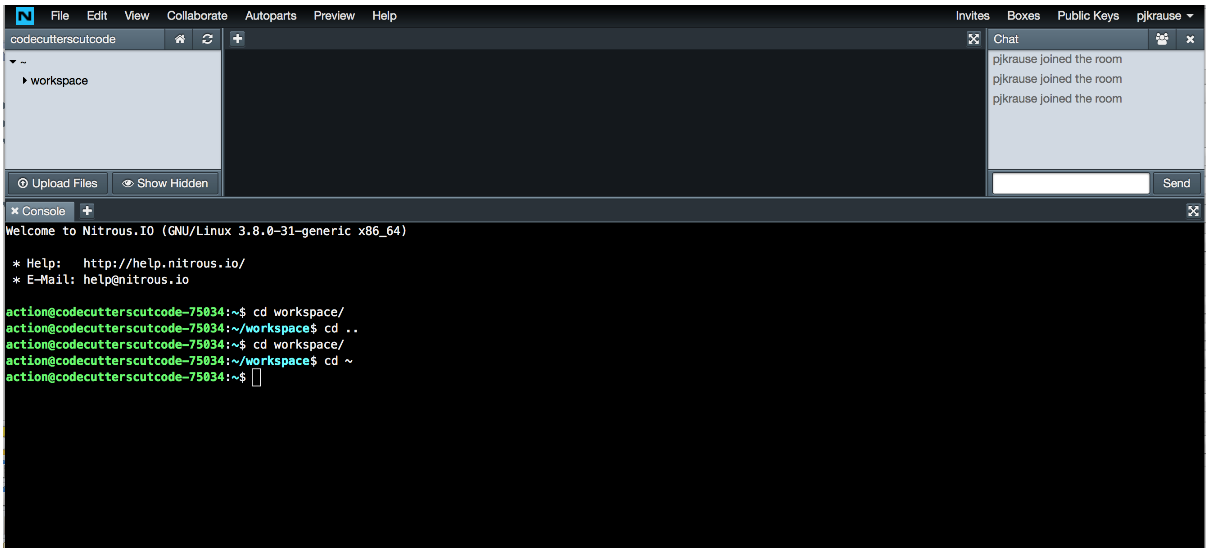
$ cd workspace

You can then press "return" and the directory will change.

You will find that as time goes by, your directory structure will build up to become quite complex. Fortunately, there is a quick way "home". As you may have seen from the command prompt, Linux recognises "~" as a shorthand for the root directory. If you type

$ cd ~

from any position in your directory structure, Linux will take you back to the root directory. Try it now.



pjkrause joined the room
pjkrause joined the room
pjkrause joined the room

```
Welcome to Nitrous.IO (GNU/Linux 3.8.0-31-generic x86_64)

 * Help:   http://help.nitrous.io/
 * E-Mail: help@nitrous.io

action@codecutterscutcode-75034:~$ cd workspace/
action@codecutterscutcode-75034:~/workspace$ cd ..
action@codecutterscutcode-75034:~$ cd workspace/
action@codecutterscutcode-75034:~/workspace$ cd ~
action@codecutterscutcode-75034:~$
```

After all that, you should see a trail of commands rather like the above picture. That's all for Linux for now. Let's do a little programming.

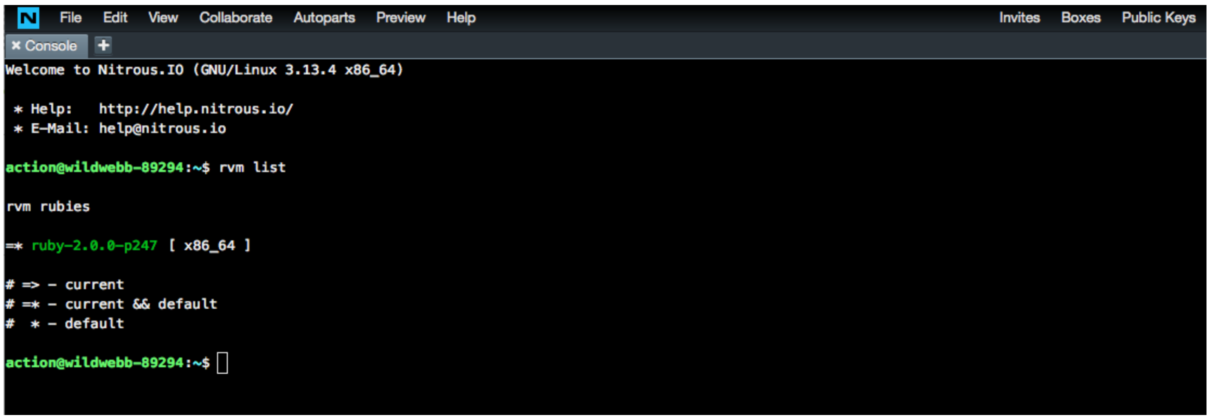
RVM

Ruby Version Manager

The Ruby Version Manager (RVM) is a tool for managing, installing and working with multiple Ruby environments. It will work on any *-nix operating system. You can find out all about it at: https://rvm.io

If you are a Windows fan and at some point wanting to move to managing Ruby on your local machine then try Pik (https://www.ruby-toolbox.com/projects/pik)

RVM is already installed in Nitrous.io so lets start using it just to get a feel for managing the very basics of your Ruby environment. The first thing to do is to list the Rubies that are currently installed:

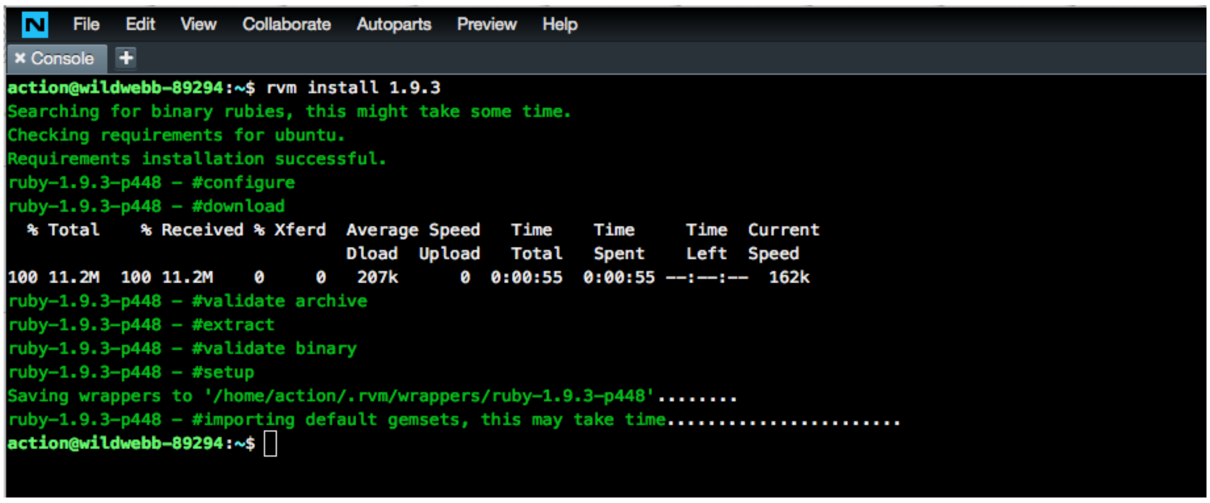


```
Welcome to Nitrous.IO (GNU/Linux 3.13.4 x86_64)

 * Help:   http://help.nitrous.io/
 * E-Mail: help@nitrous.io

action@wildwebb-89294:~$ rvm list

rvm rubies

=* ruby-2.0.0-p247 [ x86_64 ]

# => - current
# =* - current && default
#  * - default

action@wildwebb-89294:~$ 
```

Note that the editor is not visible. I clicked the icon in the top right hand side of the console to toggle it into full screen as we are not going to use the editor for a while.

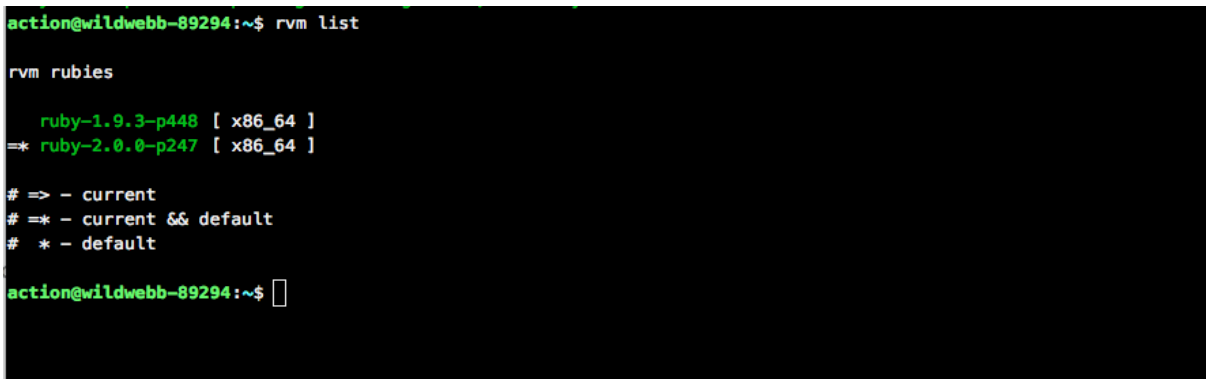
You see that when I typed the command “rvm list” I found out that version 2.0.0 was the only one installed and that it was the current and default Ruby (of course).

Now suppose I want to work on a project that was started before Ruby 2 was released. I will need to use the earlier version of Ruby. Let’s first use RVM to install that earlier version (1.9.3 in this hypothetical scenario).



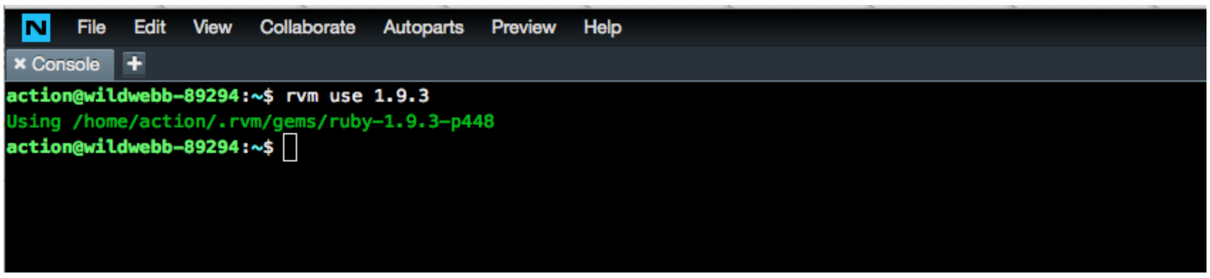
```
action@wildwebb-89294:~$ rvm install 1.9.3
Searching for binary rubies, this might take some time.
Checking requirements for ubuntu.
Requirements installation successful.
ruby-1.9.3-p448 - #configure
ruby-1.9.3-p448 - #download
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 11.2M  100 11.2M    0     0   207k      0  0:00:55  0:00:55 --:--:--  162k
ruby-1.9.3-p448 - #validate archive
ruby-1.9.3-p448 - #extract
ruby-1.9.3-p448 - #validate binary
ruby-1.9.3-p448 - #setup
Saving wrappers to '/home/action/.rvm/wrappers/ruby-1.9.3-p448'........
ruby-1.9.3-p448 - #importing default gemsets, this may take time.......................
action@wildwebb-89294:~$
```

As you see, all I had to do was type “rvm install 1.9.3” and the RVM got on with the job. Now if we type “rvm list” we have some more info.



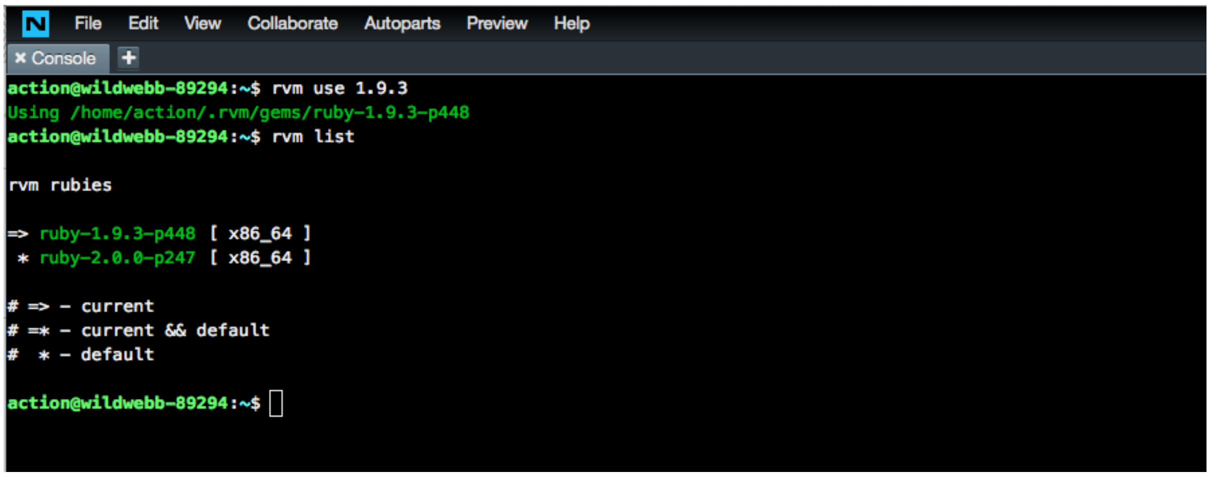
```
action@wildwebb-89294:~$ rvm list

rvm rubies

   ruby-1.9.3-p448 [ x86_64 ]
=* ruby-2.0.0-p247 [ x86_64 ]

# => - current
# =* - current && default
#  * - default

action@wildwebb-89294:~$
```

We are not there yet, though. As you can see, we need to send another command to change the current Ruby version.



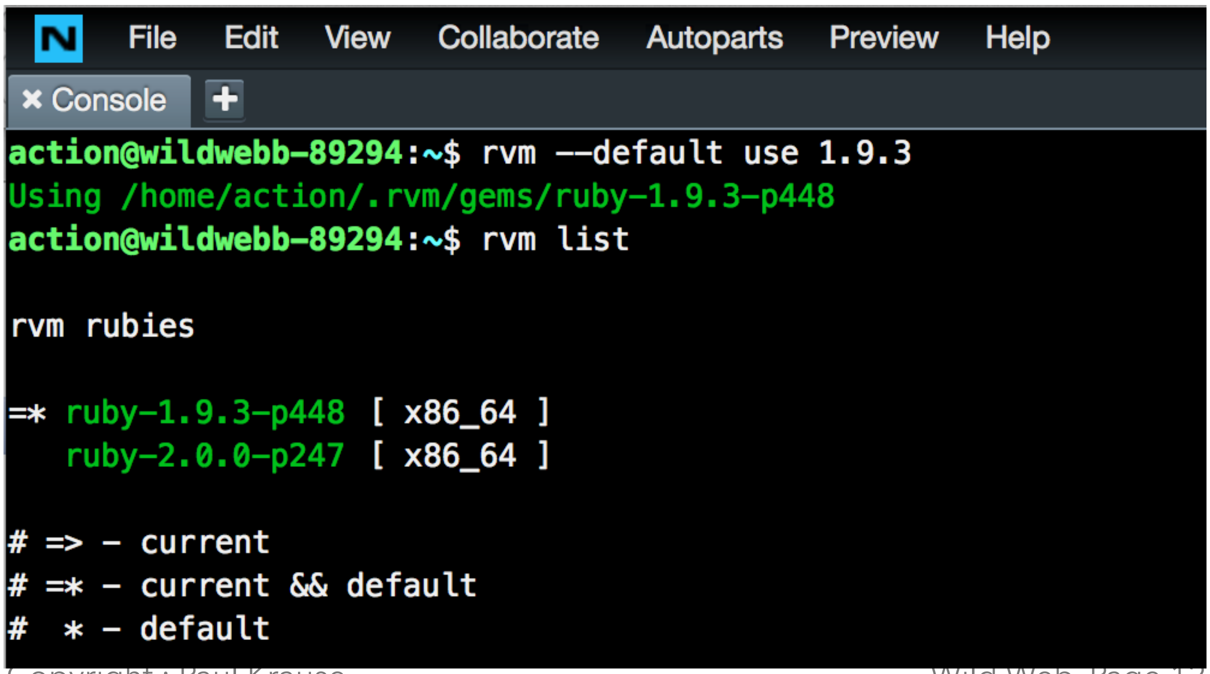
```
action@wildwebb-89294:~$ rvm use 1.9.3
Using /home/action/.rvm/gems/ruby-1.9.3-p448
action@wildwebb-89294:~$
```

Now if we list the Ruby versions again, we will see that 1.9.3 is the current version (but not the default):



```
action@wildwebb-89294:~$ rvm use 1.9.3
Using /home/action/.rvm/gems/ruby-1.9.3-p448
action@wildwebb-89294:~$ rvm list

rvm rubies

=> ruby-1.9.3-p448 [ x86_64 ]
 * ruby-2.0.0-p247 [ x86_64 ]

# => - current
# =* - current && default
#  * - default

action@wildwebb-89294:~$
```

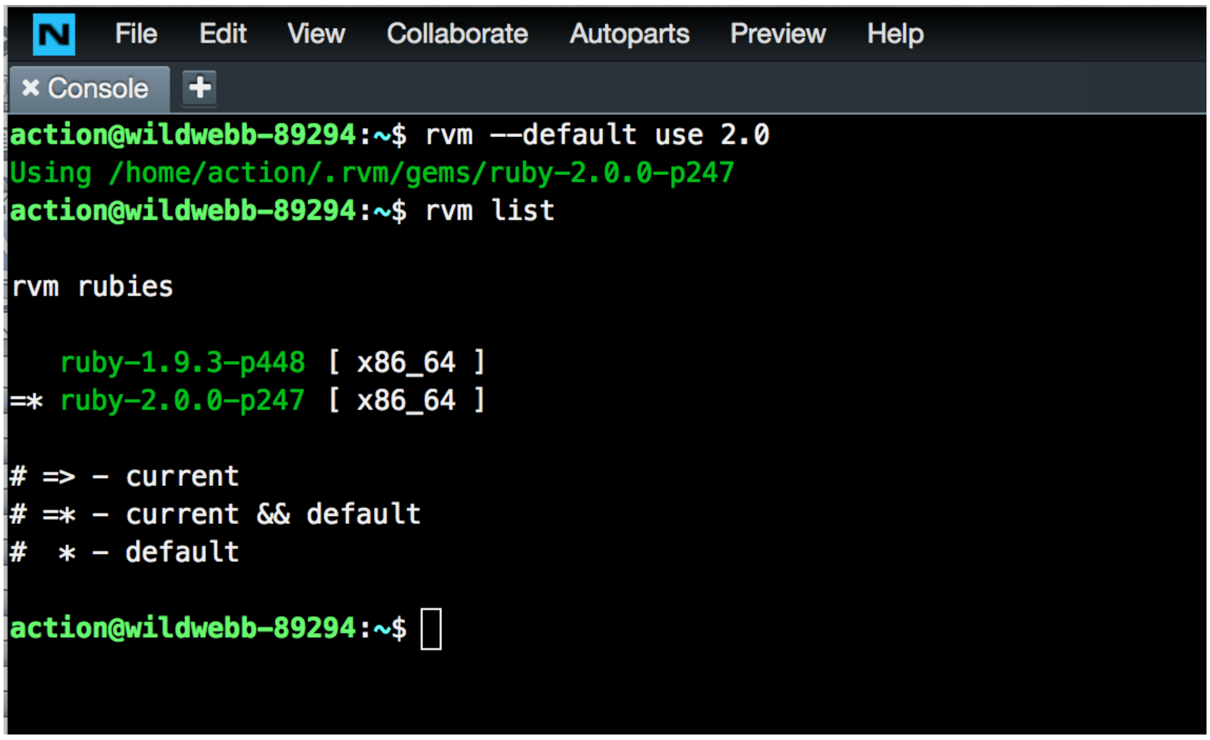
This means that Ruby 1.9.3 will only be used for the duration of the current console session. When I login again, we will be back to Ruby 2. That’s a bit risky if I am going to be working on this old project for a while, so lets change 1.9.3 to the default Ruby:



```
action@wildwebb-89294:~$ rvm --default use 1.9.3
Using /home/action/.rvm/gems/ruby-1.9.3-p448
action@wildwebb-89294:~$ rvm list

rvm rubies

=* ruby-1.9.3-p448 [ x86_64 ]
   ruby-2.0.0-p247 [ x86_64 ]

# => - current
# =* - current && default
#  * - default
```

What I did here was to set the "–default" flag for the "use" action (there are two minus signs before the word "default"), and then enter "rvm list" to show the status of my Rubies.

That is all for now, although there is much, much more that can be done with the RVM. We will come back to it later but for now let's just reset everything back to Ruby 2 because that is the version we are going to stick with for the time being:

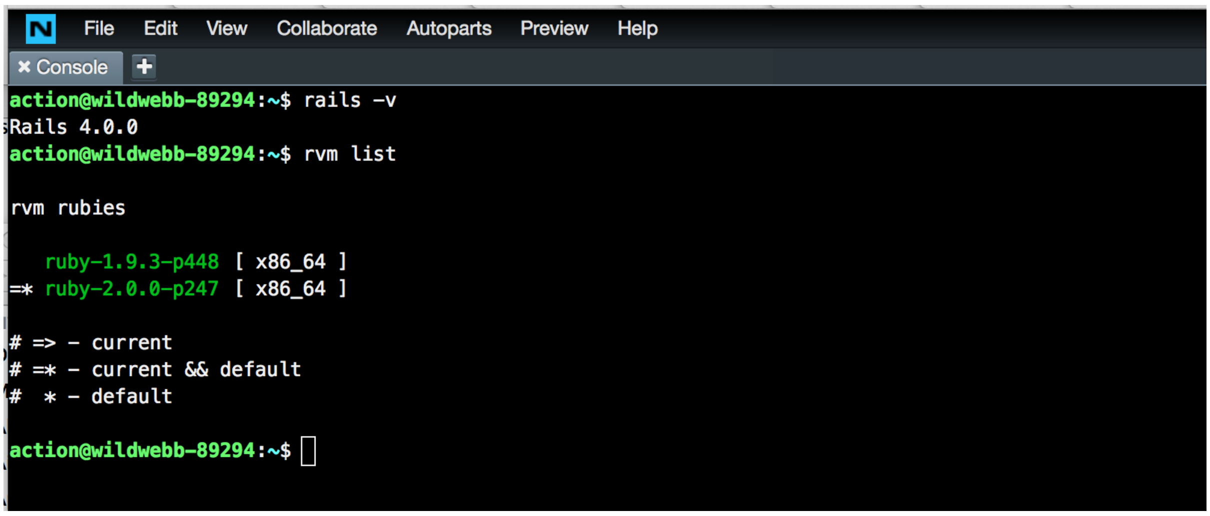


```
action@wildwebb-89294:~$ rvm --default use 2.0
Using /home/action/.rvm/gems/ruby-2.0.0-p247
action@wildwebb-89294:~$ rvm list

rvm rubies

   ruby-1.9.3-p448 [ x86_64 ]
=* ruby-2.0.0-p247 [ x86_64 ]

# => - current
# =* - current && default
#  * - default

action@wildwebb-89294:~$
```

Updating Rails

Getting Ready to Start

It is always worth making sure that you have the latest versions of Ruby and of Rails when you are starting a new project. This chapter provides an easy recipe for doing just that. Let use first take a look at the environment that Nitrous provides you (as of 7 March 2014) when you start up a new Rails box:

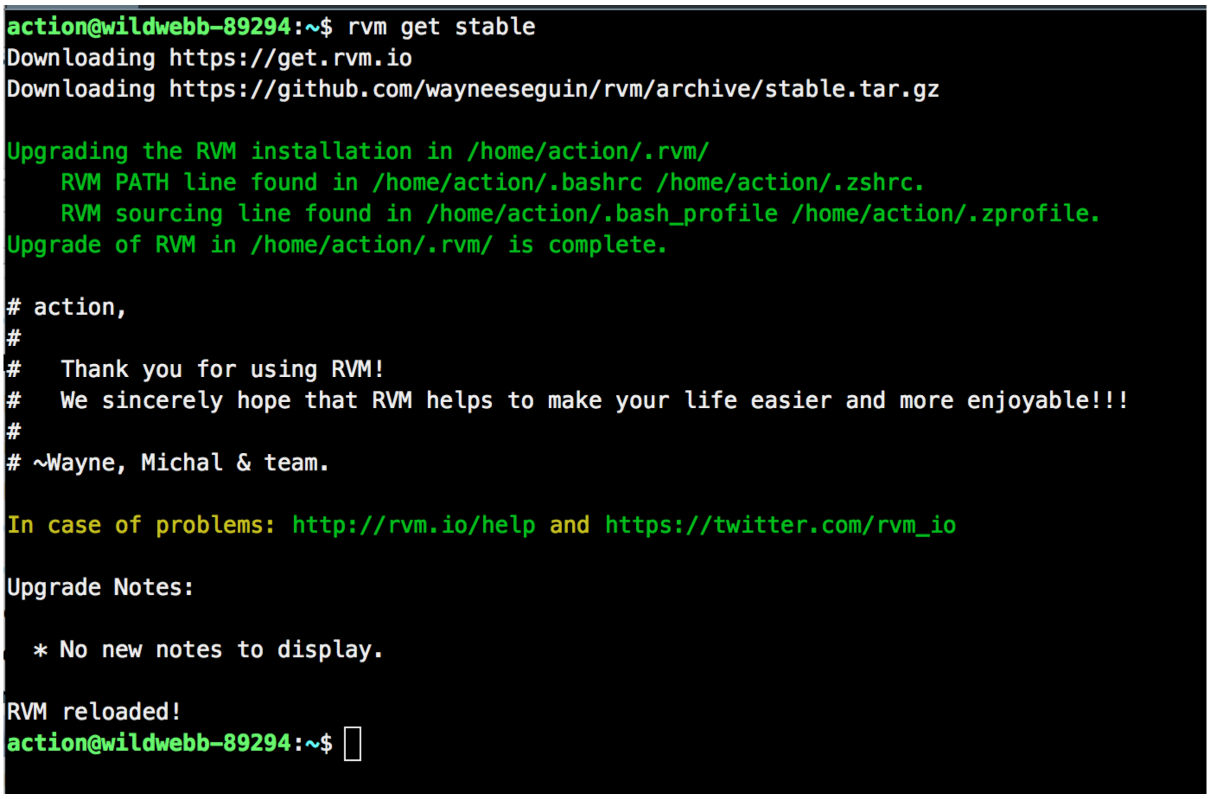


```
action@wildwebb-89294:~$ rails -v
Rails 4.0.0
action@wildwebb-89294:~$ rvm list

rvm rubies

   ruby-1.9.3-p448 [ x86_64 ]
=* ruby-2.0.0-p247 [ x86_64 ]

# => - current
# =* - current && default
#  * - default

action@wildwebb-89294:~$
```

As you will recall from the last chapter, Ruby version 2.0.0 is set as the default. In addition, I have checked the Rails version ("rails -v") and we see that version 4.0.0 is installed.

Now, the Rails team recommend usage of Ruby 2.1. It provides some performance advantages over 2.0 so it makes sense to use it with Rails. In addition, the current stable release of Rails is 4.0.3. This contains some important security fixes so it is important to use it.

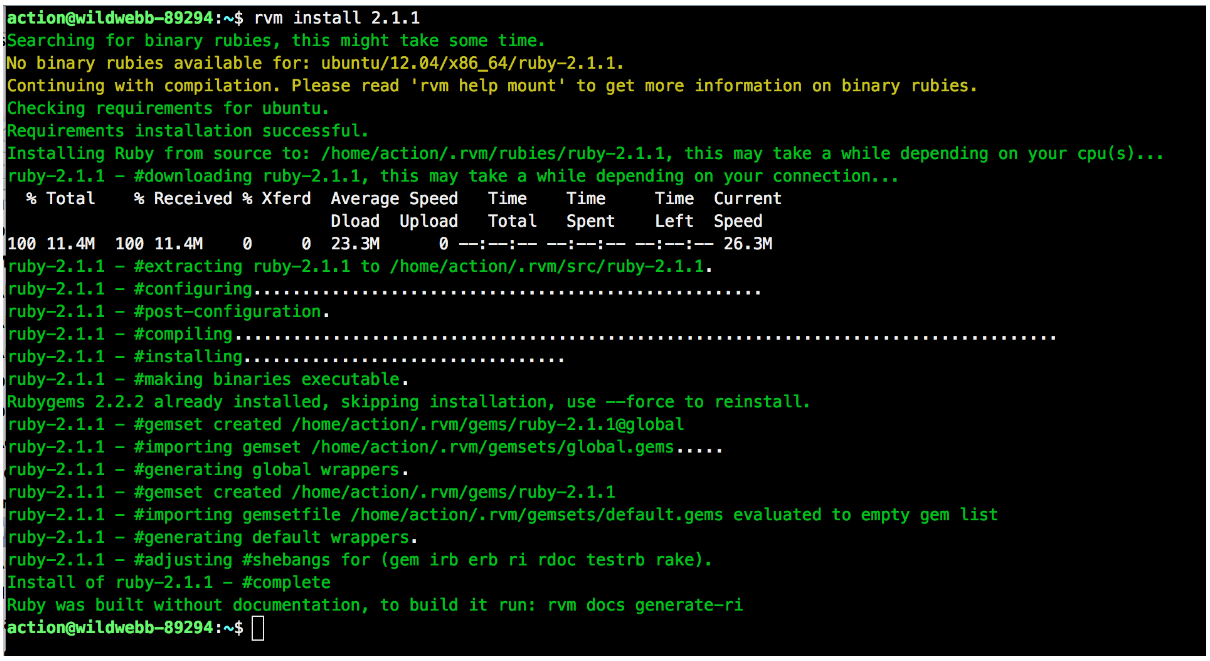
As before, we will use the RVM to install this updated version of Ruby. However, RVM itself needs to be updated to make sure that it is a version that "knows all about" the latest version of Ruby. We do that with "rvm get stable":



```
action@wildwebb-89294:~$ rvm get stable
Downloading https://get.rvm.io
Downloading https://github.com/wayneeseguin/rvm/archive/stable.tar.gz

Upgrading the RVM installation in /home/action/.rvm/
    RVM PATH line found in /home/action/.bashrc /home/action/.zshrc.
    RVM sourcing line found in /home/action/.bash_profile /home/action/.zprofile.
Upgrade of RVM in /home/action/.rvm/ is complete.

# action,
#
#   Thank you for using RVM!
#   We sincerely hope that RVM helps to make your life easier and more enjoyable!!!
#
# ~Wayne, Michal & team.

In case of problems: http://rvm.io/help and https://twitter.com/rvm_io

Upgrade Notes:

  * No new notes to display.

RVM reloaded!
action@wildwebb-89294:~$
```

As it happens, I had already got an updated version of the RVM installed, but it you have just started with a new Nitrous box the chances are it will need to download and install an update. So you will see more messages than I have shown here while it does the necessary.

Now we can use the RVM to install the latest Ruby:

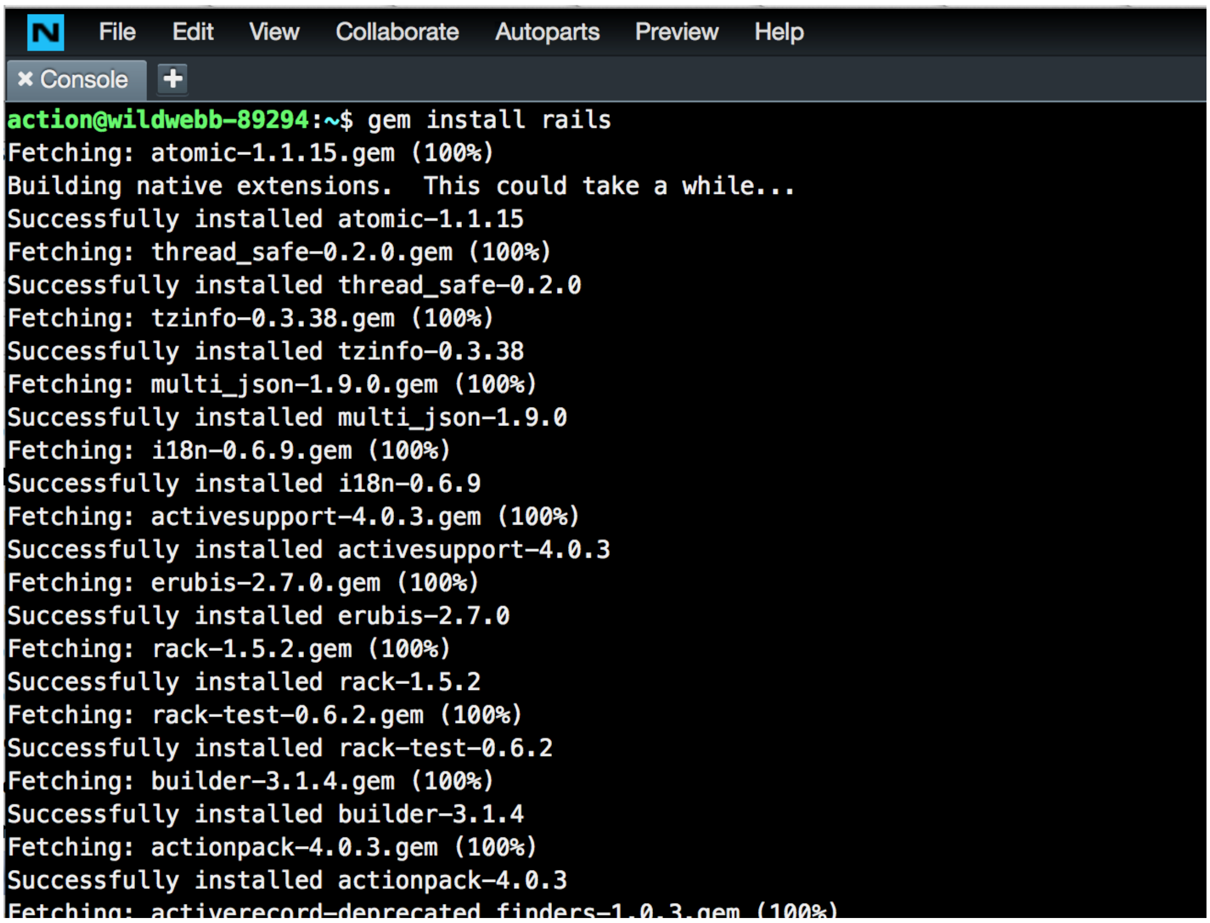


```
action@wildwebb-89294:~$ rvm install 2.1.1
Searching for binary rubies, this might take some time.
No binary rubies available for: ubuntu/12.04/x86_64/ruby-2.1.1.
Continuing with compilation. Please read 'rvm help mount' to get more information on binary rubies.
Checking requirements for ubuntu.
Requirements installation successful.
Installing Ruby from source to: /home/action/.rvm/rubies/ruby-2.1.1, this may take a while depending on your cpu(s)...
ruby-2.1.1 - #downloading ruby-2.1.1, this may take a while depending on your connection...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 11.4M  100 11.4M    0     0  23.3M      0 --:--:-- --:--:-- --:--:-- 26.3M
ruby-2.1.1 - #extracting ruby-2.1.1 to /home/action/.rvm/src/ruby-2.1.1.
ruby-2.1.1 - #configuring...................................................
ruby-2.1.1 - #post-configuration.
ruby-2.1.1 - #compiling..................................................................................
ruby-2.1.1 - #installing................................
ruby-2.1.1 - #making binaries executable.
Rubygems 2.2.2 already installed, skipping installation, use --force to reinstall.
ruby-2.1.1 - #gemset created /home/action/.rvm/gems/ruby-2.1.1@global
ruby-2.1.1 - #importing gemset /home/action/.rvm/gemsets/global.gems.....
ruby-2.1.1 - #generating global wrappers.
ruby-2.1.1 - #gemset created /home/action/.rvm/gems/ruby-2.1.1
ruby-2.1.1 - #importing gemsetfile /home/action/.rvm/gemsets/default.gems evaluated to empty gem list
ruby-2.1.1 - #generating default wrappers.
ruby-2.1.1 - #adjusting #shebangs for (gem irb erb ri rdoc testrb rake).
Install of ruby-2.1.1 - #complete
Ruby was built without documentation, to build it run: rvm docs generate-ri
action@wildwebb-89294:~$
```

It takes a little while (but it does tell you what it is doing), so now is a good time to make a cup of coffee…

Now that is done, it is a good time to install the latest version of Rails. Rails is prepared as a Ruby "gem". Gems are Ruby components that contain pre-packaged chunks of functionality. This is what makes coding in Ruby and Rails so cool - there is massive of prepackaged stuff available. With Ruby installed we can just ask for the latest version of Rails to be installed as a gem. All the gems that Rails itself depends on will also be installed.

As you will see on the next page, all we need to do is enter the command "gem install rails". This will automatically download the most recent stable release.



```
action@wildwebb-89294:~$ gem install rails
Fetching: atomic-1.1.15.gem (100%)
Building native extensions.  This could take a while...
Successfully installed atomic-1.1.15
Fetching: thread_safe-0.2.0.gem (100%)
Successfully installed thread_safe-0.2.0
Fetching: tzinfo-0.3.38.gem (100%)
Successfully installed tzinfo-0.3.38
Fetching: multi_json-1.9.0.gem (100%)
Successfully installed multi_json-1.9.0
Fetching: i18n-0.6.9.gem (100%)
Successfully installed i18n-0.6.9
Fetching: activesupport-4.0.3.gem (100%)
Successfully installed activesupport-4.0.3
Fetching: erubis-2.7.0.gem (100%)
Successfully installed erubis-2.7.0
Fetching: rack-1.5.2.gem (100%)
Successfully installed rack-1.5.2
Fetching: rack-test-0.6.2.gem (100%)
Successfully installed rack-test-0.6.2
Fetching: builder-3.1.4.gem (100%)
Successfully installed builder-3.1.4
Fetching: actionpack-4.0.3.gem (100%)
Successfully installed actionpack-4.0.3
Fetching: activerecord-deprecated_finders-1.0.3.gem (100%)
```

Many more dependent gems than you see here will be downloaded (including the Rails gem itself).

And now we are set to go. Remember that these settings only apply to the current Nitrous box. If you (terminate this one and) start a new one, you will have to do the same updates.