

Build a Sinatra API Using TDD, Heroku, and Continuous Integration with Travis



Darren Jones(<http://www.sitepoint.com/author/djones/>)

December 09, 2013



Want cutting-edge content?

Success! 2 subscribers.

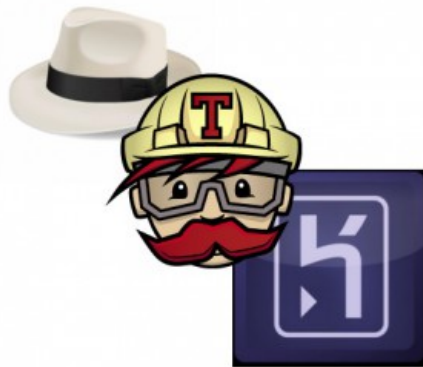


Get the best of Ruby plus exclusive deals and freebies in your inbox!

Claim your free ebook!

you@email.com

Get **Rolls Deep Dive** when you start a free 14-day SitePoint Premium trial



Get this deal!
Learn more about our services

This post was inspired by [this brilliant video \(http://ascii.io/a/4348\)](http://ascii.io/a/4348), where [Konstantin Haase \(http://rkh.im/\)](http://rkh.im/), the maintainer of [Sinatra \(http://www.sinatrarb.com/\)](http://www.sinatrarb.com/) builds a fully working app and deploys it on [Heroku \(http://heroku.com/\)](http://heroku.com/) with tests on Travis (where he works).

I decided to do a similar example that walks through each step of building an API service that demonstrates a typical Sinatra development cycle workflow – write your tests, write your code, push the code to GitHub, and check it works when deployed to Heroku with continuous integration on Travis.

The post will go from start to finish and cover the following:

Writing tests using MiniTest (<http://mattsears.com/articles/2011/12/10/minitest-quick-reference>)

Building a [Sinatra](http://www.sinatrarb.com/) (<http://www.sinatrarb.com/>) API (http://en.wikipedia.org/wiki/Application_programming_interface) that returns information in JSON (<http://en.wikipedia.org/wiki/JSON>) format

Using Bundler (<http://bundler.io/>) to manage dependencies

Using Git (<http://git-scm.com/>) to manage version control

Hosting Code on GitHub (<http://github.com/>)

Deploying and host the site on Heroku (<http://heroku.com/>)

Performing Continuous Integration with Travis CI (<https://travis-ci.org/>)

Define the Problem

The application I want to build is called Number Cruncher. It will return information about numbers, such as its factors, whether it is odd or even, and if it's prime.

The first thing I want to do is monkey-patch the `Integer` class to add a method that returns the factors of a number as an array. I also want to add a `prime?` method to test if a number is prime. Once I've done this, I'll use Sinatra to provide an API that will return information about a number in JSON format.

Number Cruncher is running live on [Heroku \(http://number-cruncher.herokuapp.com/24\)](http://number-cruncher.herokuapp.com/24)

You can see the Continuous Integration tests running on [Travis \(https://travis-ci.org/daz4126/number_cruncher\)](https://travis-ci.org/daz4126/number_cruncher)

The code is on [GitHub \(https://github.com/daz4126/number_cruncher\)](https://github.com/daz4126/number_cruncher)

Testing

Before we write any code, we should write some tests that cover the description above. First of all, create a file called 'number_cruncher.rb' and another called 'test.rb'. Then add the following MiniTest setup code to 'test.rb':

```
ENV['RACK_ENV'] = 'test'
require 'minitest/autorun'
require 'rack/test'
require_relative 'number_cruncher.rb'

include Rack::Test::Methods

def app
  Sinatra::Application
end
```

Now, we'll write a spec to test some of the features described above. Add the following code to the bottom of 'test.rb':

```

describe "Number Cruncher" do

  it "should return the factors of 6" do
    6.factors.must_equal [1,2,3,6]
  end

  it "should say that 2 is prime" do
    assert 2.prime?
  end

  it "should say that 10 is not prime" do
    refute 10.prime?
  end

  it "should return json" do
    get '/6'
    last_response.headers['Content-Type'].must_equal 'application/json;charset=utf-8'
  end

  it "should return the correct info about 6 as json" do
    get '/6'
    six_info = { number: 6, factors: 6.factors, odd: 6.odd?, even: 6.even?, prime: 6.prime? }
    six_info.to_json.must_equal last_response.body
  end

end

```

The first test is for the `factors` method that I plan to add to the `Integer` class. We check to see that if the integer 6 calls the `factors` method followed by if the factors of 6 are returned as an array.

Next, test the `prime?` method that will also be added to the `Integer` class. First of all, check to see if 2 returns `true` and then if 10 returns `false`.

The last two tests are specific to the API. The fourth test checks to see if the app is returning JSON. This is done by performing a `get` request with `'6'` as a parameter. The `last_response.headers` are checked for a content-type of `'application/json;charset=utf-8'`. In the last test, we check that the correct info is returned by comparing a JSON string to the `last_response.body` method.

To run our tests, enter the following in a terminal prompt:

```
$ ruby test.rb
```

This produces the following output:

```

# Running:
EEEEEE
Finished in 0.008729s, 572.8333 runs/s, 0.0000 assertions/s.
 5 runs, 0 assertions, 0 failures, 5 errors, 0 skips

```

All five of our tests produce errors, which is expected since we haven't written any code yet. Let's see if we can get those tests passing.

Number Cruncher Code

Start by adding the `factors` and `prime?` methods to the `Integer` class. Add the following code to 'number_cruncher.rb':

```
class Integer
  def factors
    square_root = self**0.5
    (1..square_root).map{ |n| [n,self/n] if self/n*n == self }.compact.flatten.sort
  end

  def prime?
    self.factors.size == 2 ? true : false
  end
end
```

Try running the tests again.

```
$ ruby test.rb
```

This produces the following output:

```
5 runs, 3 assertions, 0 failures, 2 errors, 0 skips
```

That's better – our first three tests pass, but the last two still fail, because we haven't implemented the API yet. We'll use Sinatra to do this. This involves requiring the `sinatra` and `json` gems, so add the following to 'number_cruncher.rb':

```
require 'sinatra'
require 'json'
```

We will also need a route that people can use to access the API. This is a simple `GET` request that accepts the a number as a parameter. It will return information about the number:

```
get '/:number' do
  content_type :json
  number = params[:number].to_i
  { number: number, factors: number.factors, odd: number.odd?, even: number.even?, prime: number.prime? }.to_json
end
```

First, change the content type to JSON using the handy `content_type` helper method that Sinatra provides. Then grab the number from the `params` hash and convert it to an integer (everything entered in the route is a string). Then create a hash of information about the number including its factors. The hash includes if the number is odd or even and whether it is prime

using our new `Integer` methods. The hash is converted to JSON using the `to_json` method provided by the `json` gem. Because this is the last line of the route handler, it will be returned automatically.

Let's run the tests again:

```
$ ruby test.rb
```

This time, we get the following output:

```
5 runs, 5 assertions, 0 failures, 0 errors, 0 skips
```

That's great, our code works! We can run it through it's paces by checking out some facts about the fifth [Fermat number](http://en.wikipedia.org/wiki/Fermat_number) (http://en.wikipedia.org/wiki/Fermat_number) using `curl`. First, start a server:

```
$ ruby number_cruncher.rb
```

Now, open another terminal window and enter:

```
$ curl http://localhost:4567/4294967297
```

This gives us the following information, which backs up what Euler found out in 1732:

```
{"number":4294967297,"factors":[1,641,6700417,4294967297],"odd":true,"even":false,"prime":false}
```

Version Control with Git

It's always useful to keep your code under version control. Git is virtually ubiquitous in the Ruby community and for good reason (it's awesome!). First, make sure that Git is installed on your system, then run the following commands:

```
$git init
```

This should give you a message similar to the one below:

```
Initialized empty Git repository in /path/to/number_cruncher/.git/
```

Use the `add .` command to add all the files in the directory:.

```
git add .  
git commit -m 'initial commit'
```

You'll get a message similar to the one below, detailing which files have been changed:

```
[master (root-commit) 6674d0c] initial commit  
2 files changed, 56 insertions(+)  
create mode 100644 number_cruncher.rb  
create mode 100644 test.rb
```

Since we are using Git for our version control, it makes sense to host our code on GitHub. If you haven't created an account, then head over there and [set one up \(https://github.com/signup/free\)](https://github.com/signup/free). There is a useful gem called 'hub' for interacting with GitHub via the terminal. To install it just enter the following to a terminal:

```
$ gem install hub
```

Now you should be able to create a new repository for you code using the following command:

```
$ hub create number_cruncher
```

This should give you the following output:

```
Updating origin  
created repository: daz4126/number_cruncher
```

Push the code to our new GitHub repository, which is done like so:

```
$ git push origin master
```

If all goes to plan, you'll see something similar to the following output:

```
Counting objects: 4, done.  
Delta compression using up to 4 threads.  
Compressing objects: 100% (4/4), done.  
Writing objects: 100% (4/4), 914 bytes, done.  
Total 4 (delta 0), reused 0 (delta 0)  
To git@github.com:daz4126/number_cruncher.git  
* [new branch]      master -> master
```

Bundler

Next we're using Bundler to manage our dependencies. This involves creating a file called 'Gemfile' that contains the following code:

```
source 'https://rubygems.org'
gem "sinatra"
gem "json"
gem "rack-test", :group => :test
```

To use Bundler, use the following command in the terminal:

```
$ bundle install
```

You should see the following information as Bundler gets and installs all the necessary gems:

```
Fetching gem metadata from https://rubygems.org/.....
Fetching gem metadata from https://rubygems.org/..
Resolving dependencies...
Using json (1.8.0)
Using rack (1.5.2)
Using rack-protection (1.5.0)
Using rack-test (0.6.2)
Using tilt (1.4.1)
Using sinatra (1.4.3)
Using bundler (1.3.5)
Your bundle is complete!
Use `bundle show [gemname]` to see where a bundled gem is installed.
```

Since we've made some changes to our code, we need to add and commit these to git before pushing the updated code to GitHub. This is easily achieved with the following three commands in the terminal:

```
$ git add .
$ git commit -m 'Added Gemfile and bundled gems'
$ git push origin master
```

Deploy to Heroku

Now it's time to deploy our code to a live server using the [Heroku \(http://heroku.com/\)](http://heroku.com/) service. For this, we will need to create a "rackup" file. Create a file called 'config.ru' that contains the following code:

```
require './number_cruncher'
run Sinatra::Application
```

To test that this works, run the command that Heroku will use in a terminal:

```
$ bundle exec rackup
```

This has the effect of starting a server, and you'll see some output similar to that shown below:

```
[2013-08-29 13:27:36] INFO WEBrick 1.3.1
[2013-08-29 13:27:36] INFO ruby 2.0.0 (2013-06-27) [i686-linux]
[2013-08-29 13:27:36] INFO WEBrick::HTTPServer#start: pid=4188 port=9292
```

Once again, we have made some changes to the code, so do the add, commit and push dance with Git:

```
$ git add .
$ git commit -m 'Added config.ru'
$ git push origin master
```

Now it's time to create the app in Heroku. If you haven't created an account, then head over there and [get one](https://id.heroku.com/signup) (<https://id.heroku.com/signup>). You also need to make sure that you have the [Heroku toolbelt](https://toolbelt.heroku.com/) (<https://toolbelt.heroku.com/>) installed. Once that is done, enter the following command in a terminal:

```
$ heroku create
```

Git is used to deploy code to Heroku using push, like so:

```
$ git push heroku master
```

If all goes well, you'll see something similar to the output below:

```
Creating number-cruncher... done, stack is cedar
http://number-cruncher.herokuapp.com/ | git@heroku.com:number-cruncher.git
Git remote heroku added
```

Make a note of the URL created for your app and then test it out by paying it a visit, with your favorite number as a parameter:

<http://number-cruncher.herokuapp.com/10>

Continuous Integration with Travis

Continuous Integration is the practice of running tests on one centralized repository of code. Travis provides this service by running tests on the code in a GitHub repository every time the code is pushed to GitHub. This means that you can always check on the status of your code. You can even configure it to deploy your code once a build is successful!

To get started, install the travis gem:


```
$ gem install travis
```

You also need to sign in to [Travis \(https://travis-ci.org/\)](https://travis-ci.org/) using your GitHub account.

Travis works by running a rake task, so create a file called 'Rakefile' that contains the following code:

```
task(:default) { require_relative 'test' }
```

This basically tells it to run the 'test.rb' file when the Rakefile is run. You can test if this works by entering the following into a terminal:

```
$ rake
```

You'll see the same output as when we run our tests:

```
Run options: --seed 58513
# Running tests:
.....
Finished tests in 0.092591s, 54.0009 tests/s, 54.0009 assertions/s.
5 tests, 5 assertions, 0 failures, 0 errors, 0 skips
```

You also need to add 'rake' to our Gemfile, edit it to look like the following:

```
source 'https://rubygems.org'
ruby '2.0.0'
gem "sinatra"
gem "json"

group :test do
  gem "rack-test"
  gem "rake"
end
```

You need to run `bundle install` again, since we've changed our Gemfile:

```
$ bundle install
```

Now, we are ready to set up Travis for our project. This is done by entering the following command in a terminal:

```
$ travis init ruby --rvm 2.0.0
```

This produces the following output:

```
repository not known to Travis CI (or no access?)
triggering sync: ..... done
.travis.yml file created!
daz4126/number_cruncher: enabled :)
```

A file called '.travis.yml' will be created that contains information about our app. A build on Travis is initialized when you push to your Github account, so do that now:

```
$ git add .
$ git commit -m 'Created a Rakefile and set up Travis'
$ git push origin master
```

To check the status of the build, run the following command in a terminal:

```
$ travis show
```

Travis can take a while to run a build, so you might get a message like the following:

```
no build yet for daz4126/Sinatra-API
```

But be patient, make a cup of tea, then come back and try again:

```
$ travis show
```

Hopefully, this time you get confirmation that the build has passed successfully:

```
Job #1.1: Created a Rakefile and set up Travis
State:      passed
Type:       push
Branch:     master
Compare URL: https://github.com/daz4126/number_cruncher/compare/4f55d5f9cd32...86c181d96f5d
Duration:   30 sec
Started:    2013-09-28 18:22:56
Finished:   2013-09-28 18:23:26
Allow Failure: false
Config:     rvm: 2.0.0
```



You'll also get a nice email from the lovely folks at Travis to confirm the successful build.

That's All Folks!

We've come to the end of this tutorial where we completed a full development cycle that can be repeated every time we want to add a new piece of code: Write tests, write code, commit changes to Git, push code to GitHub, test code using Travis, and deploy to Heroku. Rinse and repeat.

I hope you found it useful. I would love to hear any feedback about your own workflow in the comments below.

Tags: [travis](http://www.sitepoint.com/tag/travis/) (<http://www.sitepoint.com/tag/travis/>)

Was this helpful?  



Get our latest Ruby articles in your inbox, once a week, for free.

Enter your email

Get Updates



[Darren Jones](http://www.sitepoint.com/author/djones/) (<http://www.sitepoint.com/author/djones/>)

 (<https://twitter.com/daz4126>) 
(<https://plus.google.com/+DarrenJones4126>)

DAZ is the author of [Jump Start Sinatra](http://www.sitepoint.com/store/jump-start-sinatra/) (<http://www.sitepoint.com/store/jump-start-sinatra/>), a short book that helps you to get up to speed with Sinatra over a weekend. He's been using Sinatra since 2009. DAZ lives in Manchester, UK where he teaches maths and plays water polo.

6 Comments SitePoint

 Login ▾ Recommend Share

Sort by Best ▾

**RitterJack** • 2 years ago

Thx you! very clear and usefull !

2   • Share ›**daz4126** > RitterJack • 2 years ago

Thanks Jack, glad you enjoyed it!

  • Share ›**Yasser Jennani** • 2 years ago

Hello ,

thanks for this great tutorial , it's very useful

My question is when i pass a big number it takes a long time to load and i get an error msg but for small number it's fine .

  • Share ›**Seth** • 2 years ago

Something is wrong with the demo.

Internal Server Error

undefined method `detect' for nil:NilClass

WEBrick/1.3.1 (Ruby/2.0.0/2013-06-27) at number-cruncher.herokuapp.com:80  • Share ›**Guest** • 2 years ago

Can't see the app on heroku, it gives :

Internal Server Errorundefined method `detect' for nil:NilClass

  • Share ›**timmillwood** • 2 years ago

Great intro to TDD.

From someone who is often too lazy or doesn't get clients with the budget for TDD.

  • Share › Subscribe Add Disqus to your site Add Disqus Add PrivacyCOURSES >([https://www.sitepoint.com/premium/?q=&content_types\[\]=Course&utm_source=sitepoint&utm_medium=rela](https://www.sitepoint.com/premium/?q=&content_types[]=Course&utm_source=sitepoint&utm_medium=rela)

4:07:23

Ruby 2.0Darren Jones

★★★★☆

(https://www.sitepoint.com/premium/course/2-0-2906/?utm_source=sitepoint&utm_medium=rela

55:00

Build your first Rails app

Andy Hawthorne

★★★★☆

https://www.sitepoint.com/premium/course/build-your-first-rails-app-2784/?utm_source=sitepoint&utm_medium=related-content

3:07:36

JavaScript: Next Steps

M. David Green

★★★★★

https://www.sitepoint.com/premium/course/javascript-next-steps-2921/?utm_source=sitepoint&utm_medium=related-content

BOOKS >

[https://www.sitepoint.com/premium/books/?q=&content_types\[\]=Book&utm_source=sitepoint&utm_medium=related-content](https://www.sitepoint.com/premium/books/?q=&content_types[]=Book&utm_source=sitepoint&utm_medium=related-content)



Jump Start Rails

Andy Hawthorne

★★★★☆

https://www.sitepoint.com/premium/book/jump-start-rails/?utm_source=sitepoint&utm_medium=related-content



Jump Start Sinatra

Darren Jones

★★★★★

https://www.sitepoint.com/premium/book/jump-start-sinatra/?utm_source=sitepoint&utm_medium=related-content

utm_source=sitepoint&utm_medium=rela



Rails Deep Dive

Glenn Goodrich

★★★★☆

(https://www.sitepoint.com/premium/boc-deep-dive/?utm_source=sitepoint&utm_medium=rela

SCREENCASTS >

([https://www.sitepoint.com/premium/q=&content_types\[\]=ScreenCast&](https://www.sitepoint.com/premium/q=&content_types[]=ScreenCast&)

Feature Tests with RSpec: Simulate User Behavior and Test Your Ruby App

(https://www.sitepoint.com/premium/tut-tests-with-rspec-simulate-user-behavior-and-test-your-ruby-app/?utm_source=sitepoint&utm_medium=rela

Testing Views, Routes, and Helpers for Problems in Your Rails App

(https://www.sitepoint.com/premium/tut-views-routes-and-helpers-for-problems-in-your-rails-app/?utm_source=sitepoint&utm_medium=rela

Ensure Functionality with RSpec Controller Testing

Ilya Bodrov-Krukowski
(https://www.sitepoint.com/premium/tut-functionality-with-rspec-controller-testing/?utm_source=sitepoint&utm_medium=rela

About

[Our Story \(/about-us/\)](#)

[Advertise \(/advertising/\)](#)

[Press Room \(/press/\)](#)

[Reference \(http://reference.sitepoint.com/css/\)](http://reference.sitepoint.com/css/)

[Terms of Use \(/legals/\)](#)

[Privacy Policy \(/legals/#privacy\)](#)

[FAQ \(https://sitepoint.zendesk.com/hc/en-us\)](https://sitepoint.zendesk.com/hc/en-us)

[Contact Us \(mailto:feedback@sitepoint.com\)](mailto:feedback@sitepoint.com)

[Contribute \(/write-for-us/\)](#)

Visit

[SitePoint Home \(/\)](#)

[Forums \(https://www.sitepoint.com/community/\)](https://www.sitepoint.com/community/)

[Newsletters \(/newsletter/\)](#)

[Premium \(/premium/\)](#)

[References \(/sass-reference/\)](#)

[Shop \(https://shop.sitepoint.com\)](https://shop.sitepoint.com)

[Versioning \(https://www.sitepoint.com/versioning/\)](https://www.sitepoint.com/versioning/)

Connect



[\(http://www.sitepoint.com/feed/\)](http://www.sitepoint.com/feed/)



[\(/newsletter/\)](/newsletter/)



[\(https://www.facebook.com/sitepoint\)](https://www.facebook.com/sitepoint)



[\(http://twitter.com/sitepointdotcom\)](http://twitter.com/sitepointdotcom)



[\(https://plus.google.com/+sitepoint\)](https://plus.google.com/+sitepoint)

© 2000 – 2016 SitePoint Pty. Ltd.