This is how i work with livereload:

1.) Get the gem

In your Gemfile:

```
group :development do
  gem 'guard-livereload', require: false
end
```

2.) `guard init livereload`, Which will generate a `Guardfile` at the root of your App.

3.) Opening your Guardfile it should look like this (Just the Guard-Livereload, if you run other guard plugins make sure they're below the livereload.)

```
guard 'livereload' do
  watch(%r{app/views/.+\.(erb)$})
  watch(%r{app/helpers/.+\.rb})
  watch(%r{public/.+\.(css|js|html)})
  watch(%r{config/locales/.+\.yml})
  watch(%r{(app|vendor)(/assets/\w+/(.+\.(css|js|html|png|jpg))).*}) { |m| "/asset
end
```

4.) Get the Livereload Chrome App from the Chrome Web Store

5.) Restart your server and open a separate tab and type-> `guard`

6.) In your Browser push the livereload button and it should link it (Browser Connected in the Guard Tab)

I wrote it extensively for other's which may stumble upon the same question. For your specific case read **#3**. Open your guardfile and make sure livereload is called first.

> Go watch the Railscast #264 Guard

When working with SSL, livereload doesn't like that quite well.

Rack-Livereload is a neat little gem which you can add to your project to get around the SSL problems. The gem inserts a piece or Rack middleware and basically connects to the livereload app to serve up the javascript through the existing (and SSL enabled) Rails server.

share improve this answer

edited Jul 19 '14 at 19:03

answered Jul 19 '14 at 18:06

The Mini John
4,717 ● 5 ● 34 ● 82