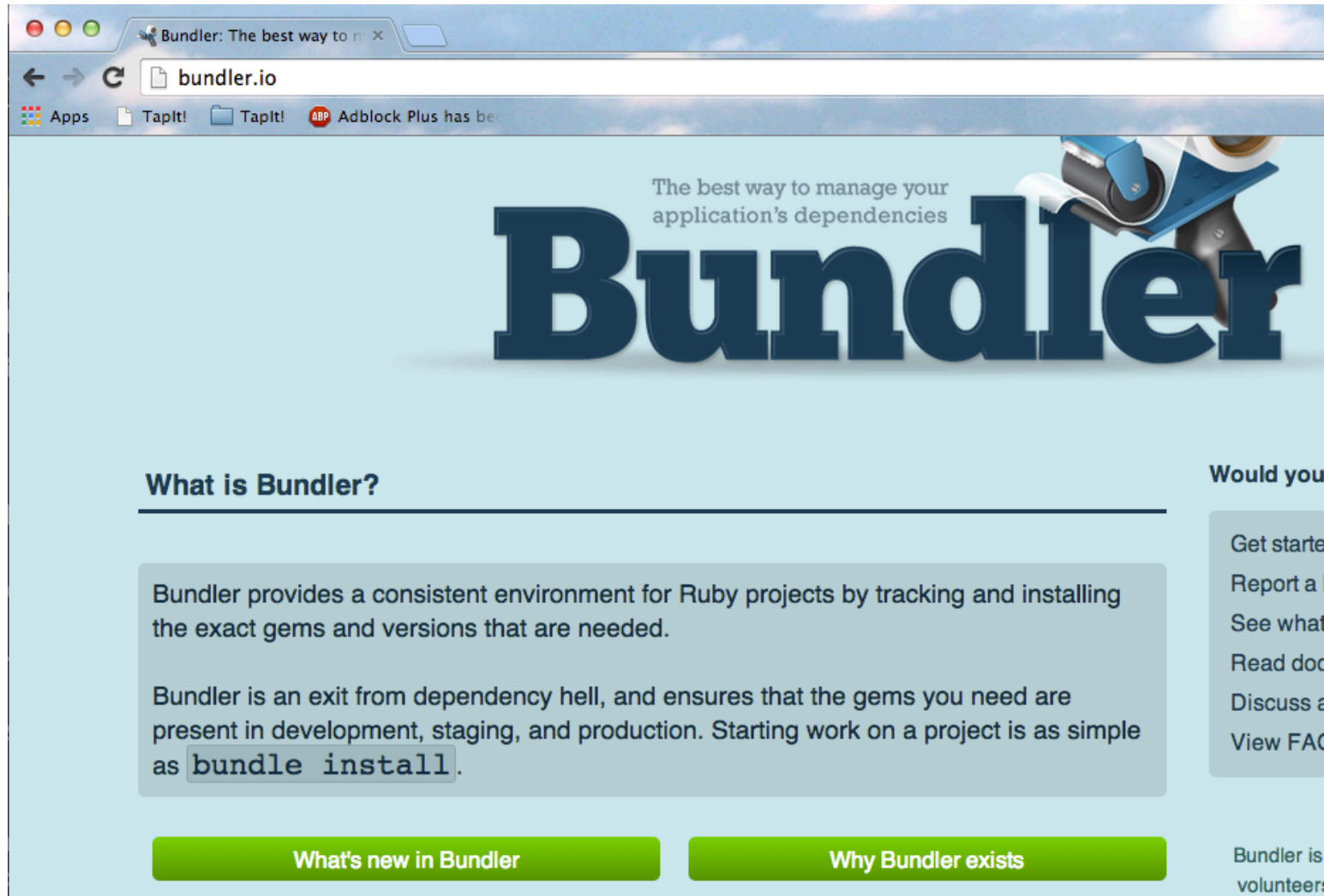




Bundler

Bundler



The screenshot shows a web browser window with the address bar displaying "bundler.io". The page features a large, stylized "Bundler" logo in the center, with the tagline "The best way to manage your application's dependencies" above it. To the right of the logo is an illustration of a robotic arm. Below the logo, there is a section titled "What is Bundler?" followed by a paragraph explaining that Bundler provides a consistent environment for Ruby projects by tracking and installing the exact gems and versions that are needed. Another paragraph states that Bundler is an exit from dependency hell and ensures that the gems you need are present in development, staging, and production. Starting work on a project is as simple as `bundle install`. On the right side, there is a sidebar with the heading "Would you" and a list of links: "Get started", "Report a", "See what", "Read doc", "Discuss a", and "View FAQ". At the bottom, there are two green buttons: "What's new in Bundler" and "Why Bundler exists".

Bundler: The best way to manage your application's dependencies

What is Bundler?

Bundler provides a consistent environment for Ruby projects by tracking and installing the exact gems and versions that are needed.

Bundler is an exit from dependency hell, and ensures that the gems you need are present in development, staging, and production. Starting work on a project is as simple as `bundle install`.

Would you

- Get started
- Report a
- See what
- Read doc
- Discuss a
- View FAQ

What's new in Bundler

Why Bundler exists

Bundler is volunteers

Bundler

- Lets you specify gems (and associated gem dependencies) for this Rails app inside `Gemfile` (in the root of your Rails app)
- Preferred way to manage gem dependencies in Rails
- Run `$bundle install` or simply `$bundle` after specifying a new gem in the Gemfile
- Run `$bundle update` when modifying a version of a gem

Bundler (Continued)

- You can instruct Rails (through `Gemfile`) to only load certain gems in specific Rails environments

```
group :development, :test do  
  gem 'webrat'  
  gem 'some_other_gem'  
end
```

Bundler – which version of gem?

- If you don't specify – gets the latest version
- Can specify an exact version or an approximate version

```
gem "nokogiri"  
gem "rails", "3.0.0.beta3"  
gem "rack", ">=1.0"  
gem "thin", ">= 1.1.0", "< 2.0"  
gem "thin", "~>1.1"
```

Same as above

"Approximately greater than" (What?)
A.k.a. Pessimistic Version Constraint

Drop the final digit, then increment to
get the upper limit version number

gem "test", ">= 2.2.0", "< 2.3.0"
is the same as
gem "test", "~> 2.2.0"

Bundler (Continued) - require

- Occasionally, the name of the gem to be used inside `require` statement is different than the name of the gem

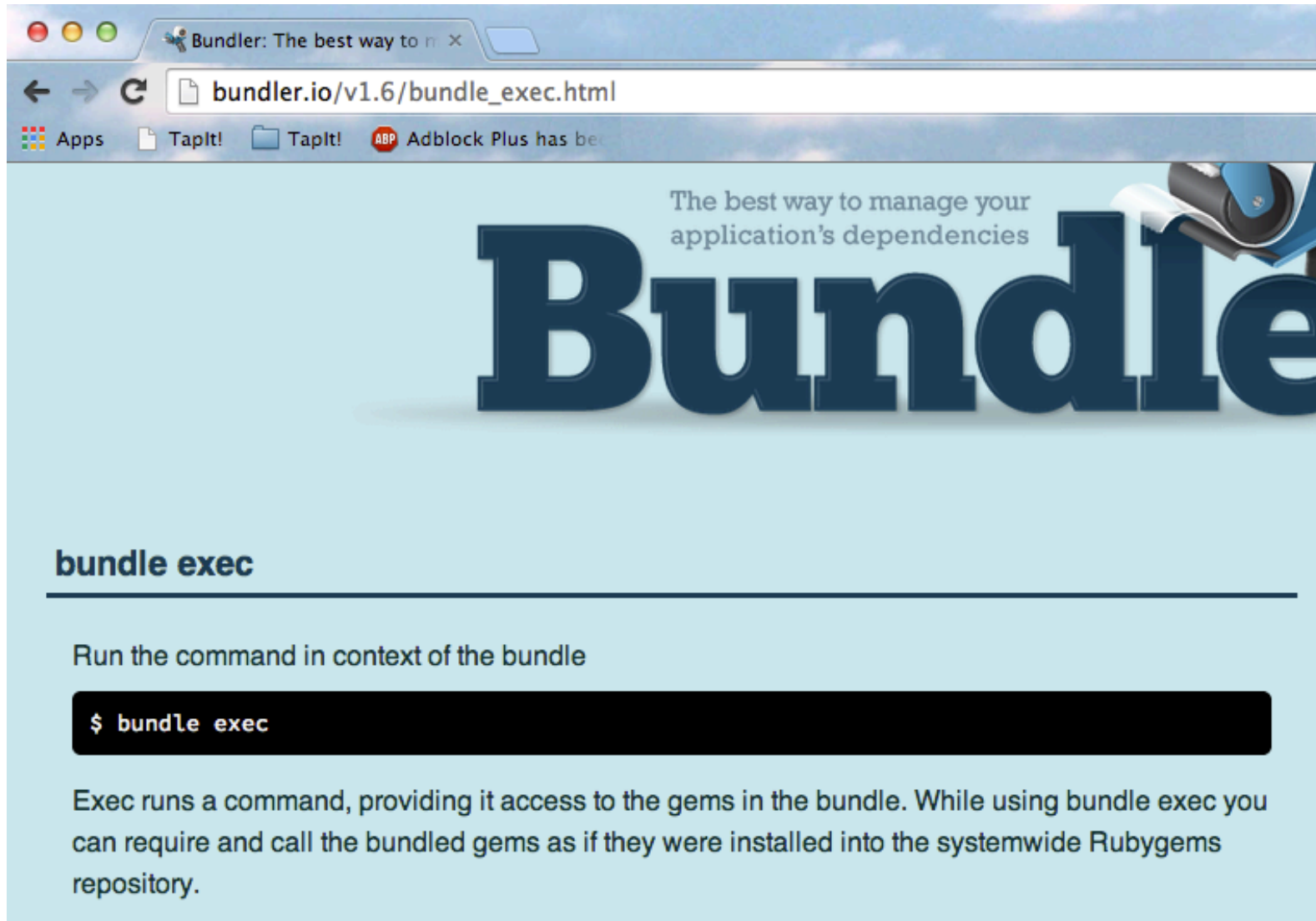
```
gem 'sqlite3-ruby', :require => 'sqlite3'
```

Gemfile - Example

```
source 'http://rubygems.org'
gem 'rails', '4.1.1'
# Bundle edge Rails instead:
# gem 'rails', github:'git://github.com/rails/rails.git'
gem 'sqlite3'
...
```

- Our app can even use a different version of Rails if you change the version and run `$bundle update`
- `$bundle` creates a `Gemfile.lock` file, which contains the gem versions your app is using with their associated dependencies

Bundle exec?



The screenshot shows a web browser window with the address bar displaying `bundler.io/v1.6/bundle_exec.html`. The page content features the Bundler logo and the tagline "The best way to manage your application's dependencies". Below this, the section "bundle exec" is highlighted with a horizontal line. The text "Run the command in context of the bundle" is followed by a code block containing the command `$ bundle exec`. A descriptive paragraph explains that `bundle exec` runs a command with access to the bundle's gems, allowing them to be used as if they were installed systemwide.

The best way to manage your application's dependencies

Bundle

bundle exec

Run the command in context of the bundle

```
$ bundle exec
```

Exec runs a command, providing it access to the gems in the bundle. While using bundle exec you can require and call the bundled gems as if they were installed into the systemwide Rubygems repository.