

Ruby on Rails

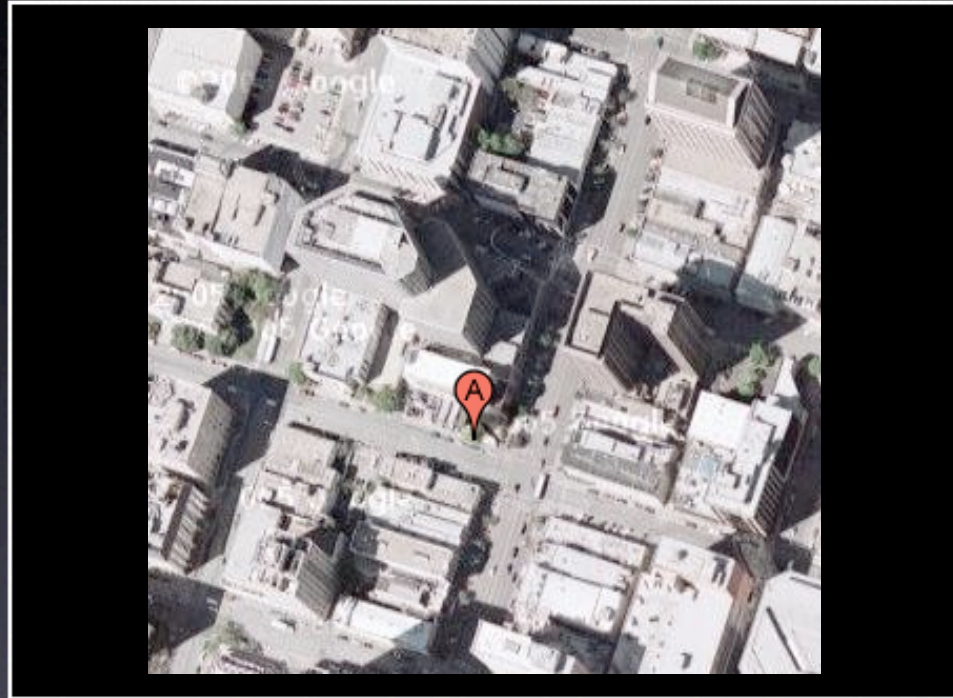


Austin User Group

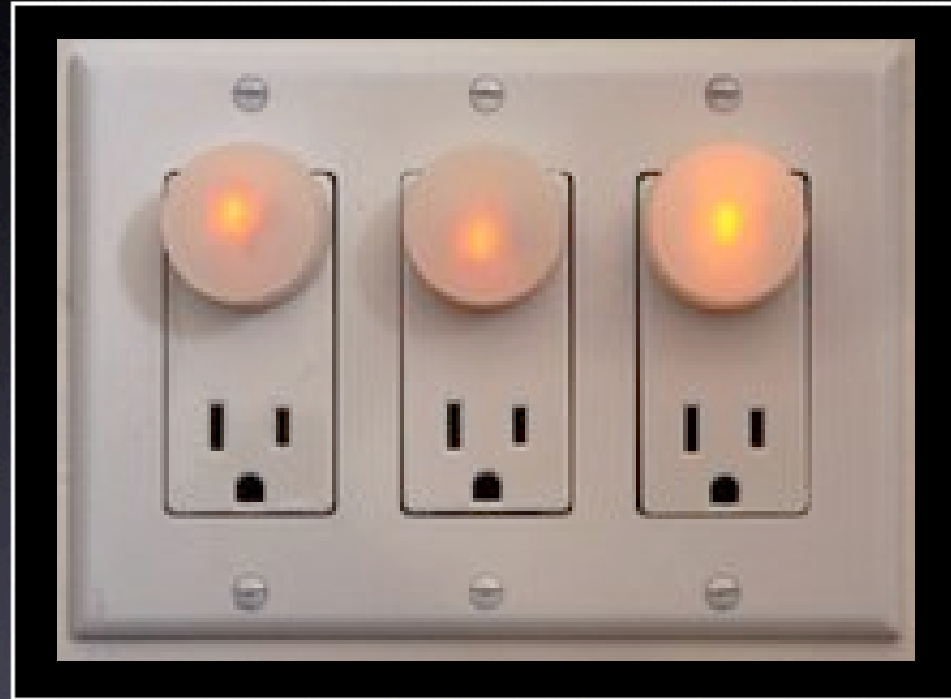
What's New in Rails 1.0?

Damon Clinkcales

<http://damonclinkcales.com/>



Birds-Eye View



Plugins

script/plugin commands

discover	Discover repository
list	List available plugins
install	Install from a known repository or URL
update	Update installed plugins
remove	Uninstall plugins
source	Add a source repos.
unsource	Remove a repos.
sources	List currently configured repos.



A Cleaner Environment

Let's have a look



Fast CGI

spawner -p 9100 -i 10 # starts 10 instances counting from 9100 to 9109

Description:

The spawner is a wrapper for spawn-fcgi that makes it easier to start multiple FCGI processes running the Rails dispatcher. The spawn-fcgi command is included with the lighttpd web server, but can be used with both Apache and lighttpd (and any other web server supporting externally managed FCGI processes).

reaper # restarts the default dispatcher

Description:

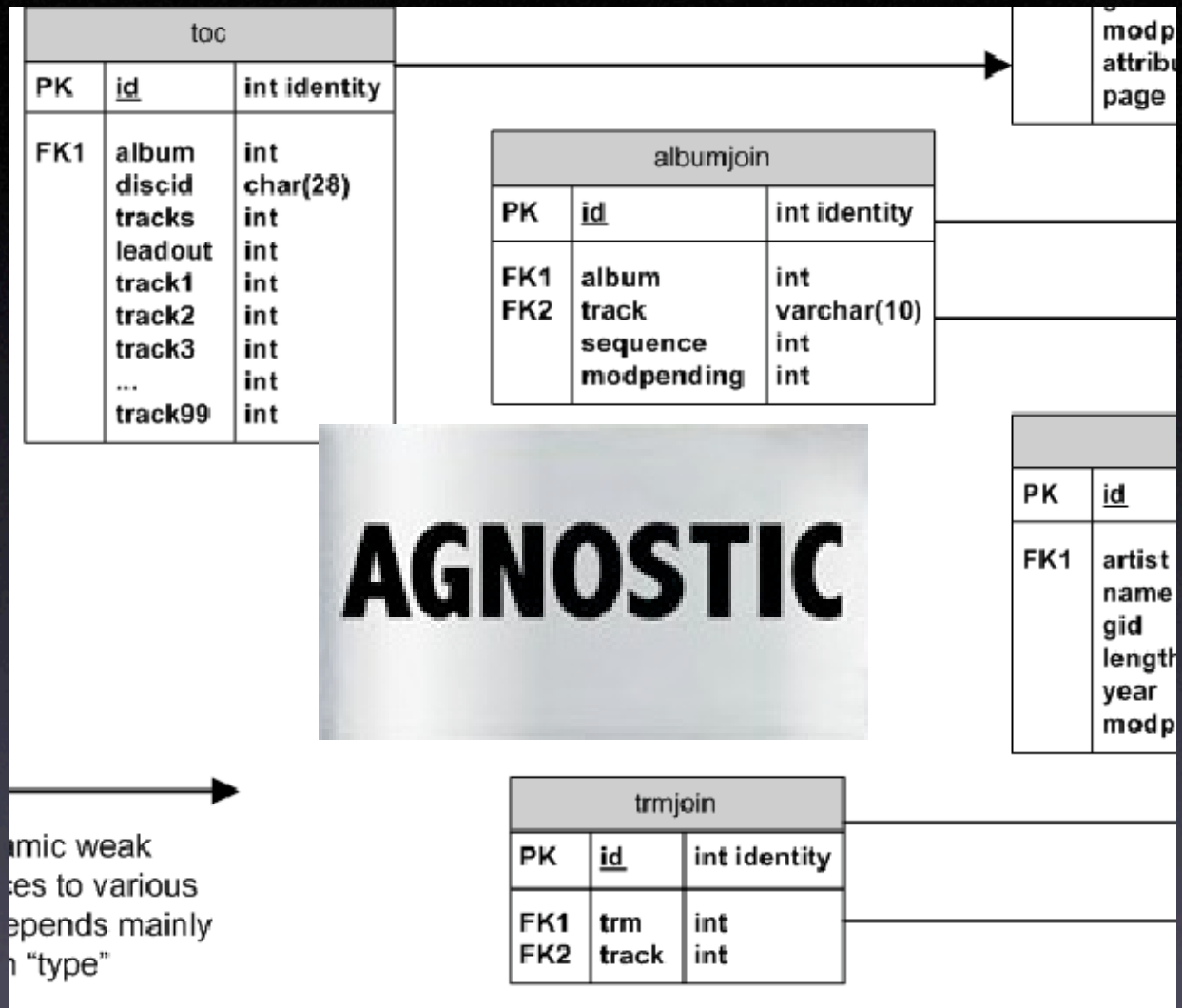
The reaper is used to restart, reload, gracefully exit, and forcefully exit FCGI processes running a Rails Dispatcher. This is commonly done when a new version of the application is available, so the existing processes can be updated to use the latest code.

spinner -i 3 -d

only run the spawner every 3 seconds and
detach from the terminal to become a daemon

Description:

The spinner is a protection loop for the spawner, which will attempt to restart any FCGI processes that might have been exited or outright crashed. It's a brute-force attempt that'll just try to run the spawner every X number of seconds, so it does pose a light load on the server.



There is no spoon

An example

```
create_table "alias", :force => true do |t|
  t.column "name", :string, :limit => 30
  t.column "value", :string, :limit => 80
  t.column "external", :boolean
  t.column "requireslogin", :boolean

  end
```

```
create_table "api_keys", :force => true do |t|
  t.column "updated_on", :timestamp
  t.column "created_on", :timestamp
  t.column "lock_version", :integer, :default => 0, :null => false
  t.column "api_key", :string, :limit => 40
  t.column "name", :string, :limit => 40

  end
```



Hey...I know you!

Session store can be enabled simply by uncommenting its definition and you can create the matching database table with:

```
rake create_session_table
```

```
class ApplicationController < ActionController::Base
  session :off
end
```

```
class PostsController < ActionController::Base
  session :off, :only => :feed

  def feed
    # won't start or use sessions
  end
end
```

```
class PostsController < ActionController::Base
  session :off, :if => Proc.new({ |request| request.post? })
end
```



Testing Changes

Fewer SQL queries for test setup improves test execution speed by 2-5x

```
# use transactional table type in mySQL  
alter table products type=InnoDB;
```

```
# Use products(:rails_book) instead of @rails_book
```

```
#new rails defaults in class Test::Unit::TestCase  
self.use_transactional_fixtures = true  
self.use_instantiated_fixtures = false
```

Resources

- **General Rails Advice** - <http://wiki.rubyonrails.com/>
- **Performance Tips** - <http://railsexpress.de/blog/>
- **Code Snippets** - <http://bigbold.com/snippets/tag/rails>
- **Agile Web Dev** - <http://pragmaticprogrammer.com/titles/rails/>
- **Testing Tips** - <http://clarkware.com/cgi/blosxom/2005/10/24#Rails10FastTesting>
- **Upgrading from 0.13.x** - <http://manuals.rubyonrails.org/read/book/19>

Ruby on Rails



Austin User Group