Ruby on Rails 3.2 Release Notes

January 13, 2015

Highlights in Rails 3.2:

- Faster Development Mode
- New Routing Engine
- Automatic Query Explains
- Tagged Logging

These release notes cover only the major changes. To learn about various bug fixes and changes, please refer to the change logs or check out the list of commits in the main Rails repository on GitHub.

1 Upgrading to Rails 3.2

If you're upgrading an existing application, it's a great idea to have good test coverage before going in. You should also first upgrade to Rails 3.1 in case you haven't and make sure your application still runs as expected before attempting an update to Rails 3.2. Then take heed of the following changes:

1.1 Rails 3.2 requires at least Ruby 1.8.7

Rails 3.2 requires Ruby 1.8.7 or higher. Support for all of the previous Ruby versions has been dropped officially and you should upgrade as early as possible. Rails 3.2 is also compatible with Ruby 1.9.2.

Note that Ruby 1.8.7 p248 and p249 have marshalling bugs that crash Rails. Ruby Enterprise Edition has these fixed since the release of 1.8.7-2010.02. On the 1.9 front, Ruby 1.9.1 is not usable because it outright segfaults, so if you want to use 1.9.x, jump on to 1.9.2 or 1.9.3 for smooth sailing.

1.2 What to update in your apps

- Update your Gemfile to depend on
  - `rails = 3.2.0`
  - `sass-rails ~> 3.2.3`
  - `coffee-rails ~> 3.2.1`
  - `uglifier >= 1.0.3`

- Rails 3.2 deprecates `vendor/plugins` and Rails 4.0 will remove them completely. You can start replacing these plugins by extracting them as gems and adding them in your Gemfile. If you choose not to make them gems, you can move them into, say, `lib/my_plugin/*` and add an appropriate initializer in `config/initializers/my_plugin.rb`.
- There are a couple of new configuration changes you'd want to add in `config/environments/development.rb`:

  ```
  # Raise exception on mass assignment protection for Active Record models
  config.active_record.mass_assignment_sanitizer = :strict

  # Log the query plan for queries taking more than this (works
  # with SQLite, MySQL, and PostgreSQL)
  config.active_record.auto_explain_threshold_in_seconds = 0.5
  ```

  The `mass_assignment_sanitizer` config also needs to be added in `config/environments/test.rb`:

  ```
  # Raise exception on mass assignment protection for Active Record models
  config.active_record.mass_assignment_sanitizer = :strict
  ```

1.3 What to update in your engines

Replace the code beneath the comment in `script/rails` with the following content:

```
ENGINE_ROOT = File.expand_path('../..', __FILE__)
ENGINE_PATH = File.expand_path('../../lib/your_engine_name/engine', __FILE__)

require 'rails/all'
require 'rails/engine/commands'
```

2 Creating a Rails 3.2 application

```
# You should have the 'rails' RubyGem installed
$ rails new myapp
$ cd myapp
```

2.1 Vendoring Gems

Rails now uses a `Gemfile` in the application root to determine the gems you require for your application to start. This `Gemfile` is processed by the Bundler gem, which then installs all your dependencies. It can even install all the dependencies locally to your application so that it doesn't depend on the system gems.

More information: Bundler homepage

2.2 Living on the Edge

`Bundler` and `Gemfile` makes freezing your Rails application easy as pie with the new dedicated `bundle` command. If you want to bundle straight from the Git repository, you can pass the `--edge` flag:

```
$ rails new myapp --edge
```

If you have a local checkout of the Rails repository and want to generate an application using that, you can pass the `--dev` flag:

```
$ ruby /path/to/rails/railties/bin/rails new myapp --dev
```

3 Major Features

3.1 Faster Development Mode & Routing

Rails 3.2 comes with a development mode that's noticeably faster. Inspired by Active Reload, Rails reloads classes only when files actually change. The performance gains are dramatic on a larger application. Route recognition also got a bunch faster thanks to the new Journey engine.

3.2 Automatic Query Explains

Rails 3.2 comes with a nice feature that explains queries generated by Arel by defining an `explain` method in `ActiveRecord::Relation`. For example, you can run something like `puts Person.active.limit(5).explain` and the query Arel produces is explained. This allows to check for the proper indexes and further optimizations.

Queries that take more than half a second to run are *automatically* explained in the development mode. This threshold, of course, can be changed.

3.3 Tagged Logging

When running a multi-user, multi-account application, it's a great help to be able to filter the log by who did what. TaggedLogging in Active Support helps in doing exactly that by stamping log lines with subdomains, request ids, and anything else to aid debugging such applications.

4 Documentation

From Rails 3.2, the Rails guides are available for the Kindle and free Kindle Reading Apps for the iPad, iPhone, Mac, Android, etc.

5 Railties

- Speed up development by only reloading classes if dependencies files changed. This can be turned off by setting `config.reload_classes_only_on_change` to false.

- New applications get a flag `config.active_record.auto_explain_threshold_in_seconds` in the environments configuration files. With a value of `0.5` in `development.rb` and commented out in `production.rb`. No mention in `test.rb`.
- Added `config.exceptions_app` to set the exceptions application invoked by the `ShowException` middleware when an exception happens. Defaults to `ActionDispatch::PublicExceptions.new(Rails.public_path)`.
- Added a `DebugExceptions` middleware which contains features extracted from `ShowExceptions` middleware.
- Display mounted engines' routes in `rake routes`.
- Allow to change the loading order of railties with `config.railties_order` like:

  ```
  config.railties_order = [Blog::Engine, :main_app, :all]
  ```
- Scaffold returns 204 No Content for API requests without content. This makes scaffold work with jQuery out of the box.
- Update `Rails::Rack::Logger` middleware to apply any tags set in `config.log_tags` to `ActiveSupport::TaggedLogging`. This makes it easy to tag log lines with debug information like subdomain and request id – both very helpful in debugging multi-user production applications.
- Default options to `rails new` can be set in `~/.railsrc`. You can specify extra command-line arguments to be used every time `rails new` runs in the `.railsrc` configuration file in your home directory.
- Add an alias `d` for `destroy`. This works for engines too.
- Attributes on scaffold and model generators default to string. This allows the following: `rails g scaffold Post title body:text author`
- Allow scaffold/model/migration generators to accept "index" and "uniq" modifiers. For example,

  ```
  rails g scaffold Post title:string:index author:uniq price:decimal{7,2}
  ```

  will create indexes for `title` and `author` with the latter being an unique index. Some types such as decimal accept custom options. In the example, `price` will be a decimal column with precision and scale set to 7 and 2 respectively.
- Turn gem has been removed from default Gemfile.
- Remove old plugin generator `rails generate plugin` in favor of `rails plugin new` command.
- Remove old `config.paths.app.controller` API in favor of `config.paths["app/controller"]`.

5.1 Deprecations

- `Rails::Plugin` is deprecated and will be removed in Rails 4.0. Instead of adding plugins to `vendor/plugins` use gems or bundler with path or git dependencies.

6 Action Mailer

- Upgraded `mail` version to 2.4.0.
- Removed the old Action Mailer API which was deprecated since Rails 3.0.

7 Action Pack

7.1 Action Controller

- Make `ActiveSupport::Benchmarkable` a default module for `ActionController::Base,` so the `#benchmark` method is once again available in the controller context like it used to be.
- Added `:gzip` option to `caches_page`. The default option can be configured globally using `page_cache_compression`.
- Rails will now use your default layout (such as "layouts/application") when you specify a layout with `:only` and `:except` condition, and those conditions fail.

```
class CarsController
  layout 'single_car', :only => :show
end
```

  Rails will use `layouts/single_car` when a request comes in `:show` action, and use `layouts/application` (or `layouts/cars`, if exists) when a request comes in for any other actions.

- `form_for` is changed to use `#{action}_#{as}` as the css class and id if `:as` option is provided. Earlier versions used `#{as}_#{action}`.
- `ActionController::ParamsWrapper` on Active Record models now only wrap `attr_accessible` attributes if they were set. If not, only the attributes returned by the class method `attribute_names` will be wrapped. This fixes the wrapping of nested attributes by adding them to `attr_accessible`.
- Log "Filter chain halted as CALLBACKNAME rendered or redirected" every time a before callback halts.
- `ActionDispatch::ShowExceptions` is refactored. The controller is responsible for choosing to show exceptions. It's possible to override `show_detailed_exceptions?` in controllers to specify which requests should provide debugging information on errors.
- Responders now return 204 No Content for API requests without a response body (as in the new scaffold).
- `ActionController::TestCase` cookies is refactored. Assigning cookies for test cases should now use `cookies[]`

```
cookies[:email] = 'user@example.com'
get :index
assert_equal 'user@example.com', cookies[:email]
```

To clear the cookies, use `clear`.

```
cookies.clear
get :index
assert_nil cookies[:email]
```

We now no longer write out HTTP_COOKIE and the cookie jar is persistent between requests so if you need to manipulate the environment for your test you need to do it before the cookie jar is created.

- `send_file` now guesses the MIME type from the file extension if `:type` is not provided.
- MIME type entries for PDF, ZIP and other formats were added.
- Allow `fresh_when/stale?` to take a record instead of an options hash.
- Changed log level of warning for missing CSRF token from `:debug` to `:warn`.
- Assets should use the request protocol by default or default to relative if no request is available.

7.1.1 Deprecations

- Deprecated implied layout lookup in controllers whose parent had an explicit layout set:

  ```
  class ApplicationController
    layout "application"
  end

  class PostsController < ApplicationController
  end
  ```

  In the example above, `PostsController` will no longer automatically look up for a posts layout. If you need this functionality you could either remove `layout "application"` from `ApplicationController` or explicitly set it to `nil` in `PostsController`.

- Deprecated `ActionController::UnknownAction` in favor of `AbstractController::ActionNotFound`.
- Deprecated `ActionController::DoubleRenderError` in favor of `AbstractController::DoubleRenderError`.
- Deprecated `method_missing` in favor of `action_missing` for missing actions.
- Deprecated `ActionController#rescue_action`, `ActionController#initialize_template_class` and `ActionController#assign_shortcuts`.

7.2 Action Dispatch

- Add `config.action_dispatch.default_charset` to configure default charset for `ActionDispatch::Response`.
- Added `ActionDispatch::RequestId` middleware that'll make a unique X-Request-Id header available to the response and enables the `ActionDispatch::Request#uuid` method. This makes it easy to trace requests from end-to-end in the stack and to identify individual requests in mixed logs like Syslog.
- The `ShowExceptions` middleware now accepts an exceptions application that is responsible to render an exception when the application fails. The application is invoked with a copy of the exception in `env["action_dispatch.exception"]` and with the `PATH_INFO` rewritten to the status code.
- Allow rescue responses to be configured through a railtie as in `config.action_dispatch.rescue_responses`.

7.2.1 Deprecations

- Deprecated the ability to set a default charset at the controller level, use the new `config.action_dispatch.default_charset` instead.

7.3 Action View

- Add `button_tag` support to `ActionView::Helpers::FormBuilder`. This support mimics the default behavior of `submit_tag`.

```
<%= form_for @post do |f| %>
  <%= f.button %>
<% end %>
```

- Date helpers accept a new option `:use_two_digit_numbers => true`, that renders select boxes for months and days with a leading zero without changing the respective values. For example, this is useful for displaying ISO 8601-style dates such as '2011-08-01'.
- You can provide a namespace for your form to ensure uniqueness of id attributes on form elements. The namespace attribute will be prefixed with underscore on the generated HTML id.

```
<%= form_for(@offer, :namespace => 'namespace') do |f| %>
  <%= f.label :version, 'Version' %>:
  <%= f.text_field :version %>
<% end %>
```

- Limit the number of options for `select_year` to 1000. Pass `:max_years_allowed` option to set your own limit.
- `content_tag_for` and `div_for` can now take a collection of records. It will also yield the record as the first argument if you set a receiving argument in your block. So instead of having to do this:

```
@items.each do |item|
  content_tag_for(:li, item) do
     Title: <%= item.title %>
  end
end
```

You can do this:

```
content_tag_for(:li, @items) do |item|
  Title: <%= item.title %>
end
```

- Added `font_path` helper method that computes the path to a font asset in `public/fonts`.

7.3.1 Deprecations

- Passing formats or handlers to render :template and friends like `render :template => "foo.html.erb"` is deprecated. Instead, you can provide :handlers and :formats directly as options: `render :template => "foo", :formats => [:html, :js], :handlers => :erb`.

7.4 Sprockets

- Adds a configuration option `config.assets.logger` to control Sprockets logging. Set it to `false` to turn off logging and to `nil` to default to `Rails.logger`.

8 Active Record

- Boolean columns with 'on' and 'ON' values are type cast to true.
- When the `timestamps` method creates the `created_at` and `updated_at` columns, it makes them non-nullable by default.
- Implemented `ActiveRecord::Relation#explain`.
- Implements `AR::Base.silence_auto_explain` which allows the user to selectively disable automatic EXPLAINs within a block.
- Implements automatic EXPLAIN logging for slow queries. A new configuration parameter `config.active_record.auto_explain_threshold_in_seconds` determines what's to be considered a slow query. Setting that to nil disables this feature. Defaults are 0.5 in development mode, and nil in test and production modes. Rails 3.2 supports this feature in SQLite, MySQL (mysql2 adapter), and PostgreSQL.
- Added `ActiveRecord::Base.store` for declaring simple single-column key/value stores.

```
class User < ActiveRecord::Base
  store :settings, accessors: [ :color, :homepage ]
end
```

```
u = User.new(color: 'black', homepage: '37signals.com')
u.color                          # Accessor stored attribute
u.settings[:country] = 'Denmark' # Any attribute, even if not specified with an accessor
```

- Added ability to run migrations only for a given scope, which allows to run migrations only from one engine (for example to revert changes from an engine that need to be removed).

  ```
  rake db:migrate SCOPE=blog
  ```

- Migrations copied from engines are now scoped with engine's name, for example `01_create_posts.blog.rb`.
- Implemented `ActiveRecord::Relation#pluck` method that returns an array of column values directly from the underlying table. This also works with serialized attributes.

  ```
  Client.where(:active => true).pluck(:id)
  # SELECT id from clients where active = 1
  ```

- Generated association methods are created within a separate module to allow overriding and composition. For a class named MyModel, the module is named `MyModel::GeneratedFeatureMethods`. It is included into the model class immediately after the `generated_attributes_methods` module defined in Active Model, so association methods override attribute methods of the same name.
- Add `ActiveRecord::Relation#uniq` for generating unique queries.

  ```
  Client.select('DISTINCT name')
  ```

  ..can be written as:

  ```
  Client.select(:name).uniq
  ```

  This also allows you to revert the uniqueness in a relation:

  ```
  Client.select(:name).uniq.uniq(false)
  ```

- Support index sort order in SQLite, MySQL and PostgreSQL adapters.
- Allow the `:class_name` option for associations to take a symbol in addition to a string. This is to avoid confusing newbies, and to be consistent with the fact that other options like `:foreign_key` already allow a symbol or a string.

  ```
  has_many :clients, :class_name => :Client # Note that the symbol need to be capitalized
  ```

- In development mode, `db:drop` also drops the test database in order to be symmetric with `db:create`.

- Case-insensitive uniqueness validation avoids calling LOWER in MySQL when the column already uses a case-insensitive collation.
- Transactional fixtures enlist all active database connections. You can test models on different connections without disabling transactional fixtures.
- Add `first_or_create`, `first_or_create!`, `first_or_initialize` methods to Active Record. This is a better approach over the old `find_or_create_by` dynamic methods because it's clearer which arguments are used to find the record and which are used to create it.

  ```
  User.where(:first_name => "Scarlett").first_or_create!(:last_name => "Johansson")
  ```

- Added a `with_lock` method to Active Record objects, which starts a transaction, locks the object (pessimistically) and yields to the block. The method takes one (optional) parameter and passes it to `lock!`.

  This makes it possible to write the following:

  ```
  class Order < ActiveRecord::Base
    def cancel!
      transaction do
        lock!
        # ... cancelling logic
      end
    end
  end
  ```

  as:

  ```
  class Order < ActiveRecord::Base
    def cancel!
      with_lock do
        # ... cancelling logic
      end
    end
  end
  ```

8.1 Deprecations

- Automatic closure of connections in threads is deprecated. For example the following code is deprecated:

  ```
  Thread.new { Post.find(1) }.join
  ```

  It should be changed to close the database connection at the end of the thread:

```
Thread.new {
  Post.find(1)
  Post.connection.close
}.join
```

Only people who spawn threads in their application code need to worry about this change.

- The `set_table_name`, `set_inheritance_column`, `set_sequence_name`, `set_primary_key`, `set_locking_column` methods are deprecated. Use an assignment method instead. For example, instead of `set_table_name`, use `self.table_name=`.

```
class Project < ActiveRecord::Base
  self.table_name = "project"
end
```

Or define your own `self.table_name` method:

```
class Post < ActiveRecord::Base
  def self.table_name
    "special_" + super
  end
end

Post.table_name # => "special_posts"
```

9 Active Model

- Add `ActiveModel::Errors#added?` to check if a specific error has been added.
- Add ability to define strict validations with `strict => true` that always raises exception when fails.
- Provide mass_assignment_sanitizer as an easy API to replace the sanitizer behavior. Also support both :logger (default) and :strict sanitizer behavior.

9.1 Deprecations

- Deprecated `define_attr_method` in `ActiveModel::AttributeMethods` because this only existed to support methods like `set_table_name` in Active Record, which are themselves being deprecated.
- Deprecated `Model.model_name.partial_path` in favor of `model.to_partial_path`.

10 Active Resource

- Redirect responses: 303 See Other and 307 Temporary Redirect now behave like 301 Moved Permanently and 302 Found.

11 Active Support

- Added `ActiveSupport:TaggedLogging` that can wrap any standard `Logger` class to provide tagging capabilities.

```
Logger = ActiveSupport::TaggedLogging.new(Logger.new(STDOUT))

Logger.tagged("BCX") { Logger.info "Stuff" }
# Logs "[BCX] Stuff"

Logger.tagged("BCX", "Jason") { Logger.info "Stuff" }
# Logs "[BCX] [Jason] Stuff"

Logger.tagged("BCX") { Logger.tagged("Jason") { Logger.info "Stuff" } }
# Logs "[BCX] [Jason] Stuff"
```

- The `beginning_of_week` method in `Date`, `Time` and `DateTime` accepts an optional argument representing the day in which the week is assumed to start.
- `ActiveSupport::Notifications.subscribed` provides subscriptions to events while a block runs.
- Defined new methods `Module#qualified_const_defined?`, `Module#qualified_const_get` and `Module#qualified _const_set` that are analogous to the corresponding methods in the standard API, but accept qualified constant names.
- Added `#deconstantize` which complements `#demodulize` in inflections. This removes the rightmost segment in a qualified constant name.
- Added `safe_constantize` that constantizes a string but returns `nil` instead of raising an exception if the constant (or part of it) does not exist.
- `ActiveSupport::OrderedHash` is now marked as extractable when using `Array#extract_options!`.
- Added `Array#prepend` as an alias for `Array#unshift` and `Array#append` as an alias for `Array#<<`.
- The definition of a blank string for Ruby 1.9 has been extended to Unicode whitespace. Also, in Ruby 1.8 the ideographic space U'3000 is considered to be whitespace.
- The inflector understands acronyms.
- Added `Time#all_day`, `Time#all_week`, `Time#all_quarter` and `Time#all_year` as a way of generating ranges.

```
Event.where(:created_at => Time.now.all_week)
Event.where(:created_at => Time.now.all_day)
```

- Added `instance_accessor: false` as an option to `Class#cattr_accessor` and friends.
- `ActiveSupport::OrderedHash` now has different behavior for `#each` and `#each_pair` when given a block accepting its parameters with a splat.

- Added `ActiveSupport::Cache::NullStore` for use in development and testing.
- Removed `ActiveSupport::SecureRandom` in favor of `SecureRandom` from the standard library.

11.1 Deprecations

- `ActiveSupport::Base64` is deprecated in favor of `::Base64`.
- Deprecated `ActiveSupport::Memoizable` in favor of Ruby memoization pattern.
- `Module#synchronize` is deprecated with no replacement. Please use monitor from ruby's standard library.
- Deprecated `ActiveSupport::MessageEncryptor#encrypt` and `ActiveSupport::MessageEncryptor#decrypt`.
- `ActiveSupport::BufferedLogger#silence` is deprecated. If you want to squelch logs for a certain block, change the log level for that block.
- `ActiveSupport::BufferedLogger#open_log` is deprecated. This method should not have been public in the first place.
- `ActiveSupport::BufferedLogger's` behavior of automatically creating the directory for your log file is deprecated. Please make sure to create the directory for your log file before instantiating.
- `ActiveSupport::BufferedLogger#auto_flushing` is deprecated. Either set the sync level on the underlying file handle like this. Or tune your filesystem. The FS cache is now what controls flushing.

  ```
  f = File.open('foo.log', 'w')
  f.sync = true
  ActiveSupport::BufferedLogger.new f
  ```

- `ActiveSupport::BufferedLogger#flush` is deprecated. Set sync on your filehandle, or tune your filesystem.

12 Credits

See the full list of contributors to Rails for the many people who spent many hours making Rails, the stable and robust framework it is. Kudos to all of them.

Rails 3.2 Release Notes were compiled by Vijay Dev.

13 Feedback

You're encouraged to help improve the quality of this guide.

Please contribute if you see any typos or factual errors. To get started, you can read our documentation contributions section.

You may also find incomplete content, or stuff that is not up to date. Please do add any missing documentation for master. Make sure to check Edge Guides first to verify if the issues are already fixed or not on the master branch. Check the Ruby on Rails Guides Guidelines for style and conventions.

If for whatever reason you spot something to fix but cannot patch it yourself, please open an issue.

And last but not least, any kind of discussion regarding Ruby on Rails documentation is very welcome in the rubyonrails-docs mailing list.