ChiliProject - Bug # 511: Encoding of strings coming out of SQLite

Status:	Closed	**Priority:**	Normal
Author:	Gary Verhaegen	**Category:**	Libraries
Created:	2011-07-07	**Assignee:**	Eric Davis
Updated:	2011-07-22	**Due date:**	
Remote issue URL:			
Affected version:	2.0.0		

Description:

The current Gemfile restricts the sqlite3-ruby gem to below 1.3; a quick search on this website did not turn out any explanation as to why this is so.

The problem is that v1.3 of the sqlite3 gem is the one that introduces correct string encoding handling, which means that for all prior versions, as far as rails is concerned, sqlite3 only outputs ASCII-8 strings. This is not a problem in ruby18, but in ruby19 any page extracting any non-ASCII-7 character from the database will then trip up the ERB templating system with a cryptic error message that does not mention the database at all.

The fix is easy, since sqlite3 v1.3 and above honor the Encoding.default_internal property; I have sent a pull request to chiliproject at github with a fix.

Associated revisions

2011-07-22 11:35 pm - Eric Davis

[#511] Fix string encodings coming from sqlite3 in MRI 1.9.x

History

2011-07-07 08:44 am - Gregor Schmidt

Thanks for your investigations.

I'm not sure, why we stick to sqlite3 < 1.3. I think it is related to 1.8.6 support. But since, we are no longer support 1.8.6 with ChiliProject 2.0, it might very well be, that the reasons to do so, have vanished.

So I see 2 options here.

1. Your proposed solution to use a newer version on Ruby 1.9

2. Remove the limitation and make it work for all Rubies.

Actually I would prefer the second, but I currently do not have the time to run the tests on at least 1.8.7 and 1.9.2 with sqlite3 1.3.3. Also it would be great, if it would not totally break on JRuby 1.6.x

If there are reasons, why we need to go with option 1, I would prefer, if you used the "platform mechanism of bundler":http://gembundler.com/man/gemfile.5.html#PLATFORMS-platforms-. This way, bundler will be aware of platform differences and can resolve dependency cross-platform, but install only for the local one.

Would you have the time to test option 2 ?

2011-07-07 08:45 am - Holger Just

Cited from https://github.com/finnlabs/chiliproject-gemfile which served as the basis for ChiliProject's current Gemfile:

bq. Target operating system is currently Debian Stable [i.e. Lenny --Holger]. This may result in the need
to use older versions of some gems, than technically possible. This is most
notably the following gems
* sqlite3-ruby must be older than 1.3
* rmagick must be older than 2.0

2011-07-08 06:22 pm - Eric Davis

- Category set to Libraries

The current Debian stable is "squeeze" now.

* libsqlite3-ruby is currently at 1.2.4 in squeeze (stable) and 1.3.3 in wheezy (testing)
* librmagick-ruby is currently at 2.13.1 in squeeze (stable) and wheezy (testing)

I think updating rmagick is good. Until Debian's stable updates, could we use sqlite3 <1.3 with MRI 1.8.x and any version with MRI 1.9?

2011-07-08 06:27 pm - Gregor Schmidt

I'll quickly prepare a pull request, that updates rmagick and adds special handling for sqlite in 1.9.

2011-07-08 07:22 pm - Gregor Schmidt

Ok. I'm giving up. It is not possible to express the things, Eric laid out using bundler's API.

Something like the following is not possible.

<pre>
gem "sqlite3", :platform => :ruby_19
gem "sqlite3-ruby", "< 1.3", :require => "sqlite3", :exclude_platform => :ruby_19
</pre>

This way, bundler would be able to properly resolve versions for all dependencies. But as I said, it is not possible to express things that way. The only option would be to use the initial approach, proposed by Gary Verhaegen. On the other hand, this is generally not desirable. See "the bundler FAQ":http://gembundler.com/rationale.html#faq-3 for a discussion - there it is explained for groups, but the same goes for platforms. Whenever we are hiding dependencies with conditionals, we are running into new kinds of problems.

I was able to install sqlite3 1.3 properly on Debian Squeeze (with RVM), so technically, there seems to be no reason to stick to 1.3. The only one would be, that the system's gem (installed via apt-get) would satisfy the requirement and bundler would not try to install a newer version. This might save you from installing build tools on your app server.

So to put it another way: I'm not able to fulfill Eric's request, without hurting bundler's basic way of doing things. I, personally, would not face any disadvantages from updating to sqlite3 1.3, since I am generally avoiding the package maintainers versions of ruby and gems anyway, and the gem seems to build fine on Debian Stable.

Again. I'm giving up on this issue. I'll let smarter people - having more experience in installation and dependency management - decide.

2011-07-08 07:31 pm - Felix Schäfer

I'm not sure where I already said that, but I'd be in favor of having a more generic Gemfile and add the necessary changes for distro-specific changes in the tutorials and/or as comments.

2011-07-12 01:28 am - Eric Davis

Couldn't we do this?

<pre><code class="ruby">
gem "sqlite3", :platforms => :mri_19
gem "sqlite3-ruby", "< 1.3", :require => "sqlite3", :platforms => :mri_18
</code></pre>

This is how I'm loading ruby-debug for 1.8 vs 1.9 "locally":https://github.com/edavis10/local/blob/master/Gemfile. (I don't know if there is a difference between mri_18 and ruby_18)

If not, what would be the simplest thing that could work? Have our Gemfile require Debian testing and add docs for changing it for Debain stable?

2011-07-12 06:06 am - Gregor Schmidt

Gnahh. I totally missed the surrounding @platforms :mri@ block. In that case, would can be pretty sure, that we've only got @mri_18@ and @mri_19@. So your code block should actually work.

Now I'm realizing, that we've got not database adapters for rubinius defined. I should shed a light on that in the future.

Whoever is committing this should leave a comment or link to this issue, so that we are able to find out, why we are using an older sqlite3 on Ruby 1.8.

2011-07-22 09:38 pm - Eric Davis

- Target version set to 2.1.0

- Assignee set to Eric Davis

- (deleted custom field) set to 2.0.0

- Status changed from Ready for review to Closed

I've updated the sqlite gem in the Gemfile for 1.9.x (commit:9c47208). The old version was sending strings back with the incorrect encoding.

I've also added a few commits that should force re-encoding some of the SCM data after it comes out of the database (commit:d2724ef89 and commit:bbb4f63ae).