Redmine - Defect #2626

Error during translation of the "is too short" and "is too long" messages

2009-01-30 17:53 - Andrew R Jackson

Status:	Closed	**Start date:**	2009-01-30
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Translations	**Estimated time:**	0.00 hour
Target version:	0.8.2		
Resolution:	Fixed	**Affected version:**	

Description

PROBLEM:
This class of message is causing ActionView::TemplateError translation value error ("too few arguments") and a resulting 500 HTTP error in 0.8.0.

REASON:
The cause is in the error_message_for() method of app/helpers/application_helper.rb--prior to the call to l() to obtain a localized msg, the msg Array is forcibly transformed into msg.first which is the String to display to the user.

This breaks the call to GLoc's l() method when the String has dynamically filled fields which *used* to follow the String in msg, but were stripped off. The actual error occurs in GLoc's internal _l() method.

GLoc is present at vendor/plugins/gloc-1.1.0/

MY FIX:
Repair application_helper.rb to not do that transformation. In fact, fix it so that if msg is just String, it is turned into an Array with just the String (opposite of what is there) and call l(*msg) instead of l(msg).

PATCH (changes 2 lines):

```
--- broken/app/helpers/application_helper.rb   2009-01-26 09:49:24.000000000 -0600
+++ app/helpers/application_helper.rb   2009-01-30 10:17:35.000000000 -0600
@@ -524,11 +525,12 @@
       full_messages = []
       object.errors.each do |attr, msg|
         next if msg.nil?
-        msg = msg.first if msg.is_a? Array
+        msg = [msg] if(msg.is_a?(String))
         if attr == "base"
           full_messages << l(msg)
         else
-          full_messages << "&#171; " + (l_has_string?("field_" + attr) ? l("field_" + attr) : obj
ect.class.human_attribute_name(attr)) + " &#187; " + l(msg) unless attr == "custom_values"
+          full_messages << "&#171; " + (l_has_string?("field_" + attr) ? l("field_" + attr) : obj
ect.class.human_attribute_name(attr)) + " &#187; " + l(*msg) unless attr == "custom_values"
         end
       end
       # retrieve custom values error messages
```

My env:

```
  mysql-5.0.67
  rails (2.1.2)
  ruby 1.8.7 (2008-08-08 patchlevel 71) [x86_64-linux]
  redmine 0.8.0
```

Related issues:

Has duplicate Redmine - Defect #2787: validation fails when project name exce...	**Closed**	**2009-02-20**

Associated revisions

Revision 2486 - 2009-02-20 19:55 - Jean-Philippe Lang

Fixes "too few arguments" error on activerecord error translation (#2626).

History

#1 - 2009-01-30 23:16 - Andrew R Jackson

Ok, had some others patching their Redmine installations to fix this same issue and obviously I did something stupid to mess up the patch (to be honest, maybe the 'fix' is dumb too, but it's working for us and we're goal-oriented, heh).

Here's a proper one I think and this one also preemptively fixes the other l(msg) calls in error_messages_for() :

```
--- broken/app/helpers/application_helper.rb   2009-01-30 16:07:47.000000000 -0600
+++ app/helpers/application_helper.rb        2009-01-30 16:03:36.000000000 -0600
@@ -524,11 +525,11 @@
       full_messages = []
       object.errors.each do |attr, msg|
         next if msg.nil?
-        msg = msg.first if msg.is_a? Array
+        msg = [msg] if(msg.is_a?(String))
         if attr == "base"
-          full_messages << l(msg)
+          full_messages << l(*msg)
         else
-          full_messages << "&#171; " + (l_has_string?("field_" + attr) ? l("field_" + attr) : object.class.hu
man_attribute_name(attr)) + " &#187; " + l(msg) unless attr == "custom_values"
+          full_messages << "&#171; " + (l_has_string?("field_" + attr) ? l("field_" + attr) : object.class.hu
man_attribute_name(attr)) + " &#187; " + l(*msg) unless attr == "custom_values"
         end
       end
       # retrieve custom values error messages
@@ -536,8 +537,8 @@
         object.custom_values.each do |v|
           v.errors.each do |attr, msg|
             next if msg.nil?
-            msg = msg.first if msg.is_a? Array
-            full_messages << "&#171; " + v.custom_field.name + " &#187; " + l(msg)
+            msg = [msg] if(msg.is_a?(String))
+            full_messages << "&#171; " + v.custom_field.name + " &#187; " + l(*msg)
           end
         end
       end
```

#2 - 2009-02-01 21:15 - Jean-Philippe Lang

It was reported in #2200 but I'm not able to reproduce this problem using ruby 1.8.7/rails 2.1.2.
Which language are you using? Did you made any change to this lang file or to other file?

#3 - 2009-02-02 16:51 - Andrew R Jackson

I see you are also using Ruby 1.8.7 and Rails 2.1.2 like we are.

We haven't changed any language files or a language setting. The default is English, right? Not being familiar, I'm guessing it's using lang/en.yml as a source of message Strings as part of localization. And the GLoc library to effect that localization.

Redmine was obtained via: "svn co svn://rubyforge.org/var/svn/redmine/trunk ."

This included vendor/plugins/gloc-1.1.0/

I believe I saw the other bug and maybe another, but they didn't seem to recognize that the code was causing *any* message with format fields (%d, %f, etc) that gets translated via GLoc's l() method to throw an exception. So it was a much broader problem.

I was able to replicate their bug and others like it with Ruby 1.8.7 and Rails 2.1.2. I was thinking folks seeing this problem might like to have a "fix" to the code that has worked for us. We put the "fix" in our Redmine install notes, since any Redmine install we did had this problem.

I don't suppose you have some other version of GLoc in addition to the one in vendor/, perhaps that works a little differently?

Or perhaps your error_message_for() in application_helper.rb hasn't got the offending "msg = msg.first if msg.is_a? Array" lines?

I see that lang/en.yml has the relevant messages:

- text_caracters_maximum: %d characters maximum.
- text_caracters_minimum: Must be at least %d characters long.

Those are going to throw exceptions if l(msg) is called after doing "msg = msg.first if msg.is_a? Array". When msg was an Array, it had the message String *and* the appropriate argument to fill the %d. My so-called fix just makes sure l() receives both the String and the arguments to fill it (and thus not crash).

Thanks for taking a look, Jean-Philippe.
I have to admit that we just started trying out Redmine and it's great! Also I have to admit that I'm not familiar with Rails, but I can read code and make stuff work (maybe not the right way, but *a* way)

#4 - 2009-02-02 18:26 - Jean-Philippe Lang

Thanks for your feedback.
I understand why the missing arguments raise an error but I can't see why it occurs.

The 2 strings you mention (text_caracters_maximum and text_caracters_minimum) are never used to display validation error messages. They're just used to display tips on various form and they're always called with the appropriate argument.

The validation errors use these string instead:

- activerecord_error_too_long: is too long
- activerecord_error_too_short: is too short

And as you can see, there's no parameter to supply.

#5 - 2009-02-02 18:54 - Andrew R Jackson

Hi Jean-Philippe,

Jean-Philippe Lang wrote:

> The 2 strings you mention (text_caracters_maximum and text_caracters_minimum) are never used to display validation error messages. They're just used to display tips on various form and they're always called with the appropriate argument.
>
> The validation errors use these string instead:
>
> - activerecord_error_too_long: is too long
> - activerecord_error_too_short: is too short

Hmm, I see...no, it's the 2nd part of the message that is the problem. Let me find two I know used to cause exceptions. Here we go:

The 'too short' message from Change Password form:

« Password » is too short (minimum is 4 characters)

The too long and a too short message from Create New Project Form:

« Name » is too long (maximum is 30 characters)
 « Identifier » can't be blank
 « Identifier » is too short (minimum is 2 characters)

It is the "maximum" and "minimum" part of the messages that cause the error. Debugging shows that these have %d where the integers go, but no arguments to fill the %d are available in GLoc._l()

> And as you can see, there's no parameter to supply.

Yes, I see that. Um, where does the 2nd half of the message come from? Is that a standard Rails thing I'm not familiar with, but that is being retrieved as msg by the iterator object.errors.each do |attr, msg| in error_messages_for()? Because debugging shows that msg is an Array containing the String and a numeric argument to fill the format field (such as 4, 30, etc).

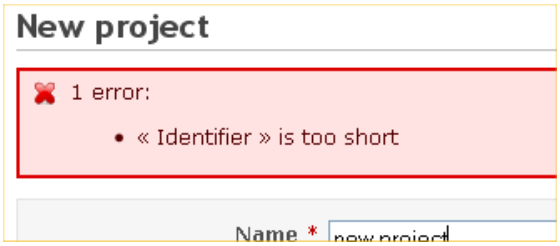
Thanks,
Andrew

#6 - 2009-02-02 19:04 - Jean-Philippe Lang

- File too_short_error.png added

AFAIK, "is too short (minimum is 4 characters)" is a default Rails 2.1 error message.
It should be overriden by source:/trunk/config/initializers/10-patches.rb.

Which gives:



#7 - 2009-02-02 19:36 - Andrew R Jackson

Jean-Philippe Lang wrote:

> AFAIK, "is too short (minimum is 4 characters)" is a default Rails 2.1 error message.
> It should be overriden by source:/trunk/config/initializers/10-patches.rb.

The 10-patches.rb we have looks like that, minus the AsynchronousMailer code you added 2 days ago (Revision 2339).

Although, I don't understand since it says:

10 :too_long => "activerecord_error_too_long",
 11 :too_short => "activerecord_error_too_short",

Which, to my ignorant eye, seems to indicate to use the messages that came with activerecord? How is that doing the overriding (I'm getting out of my depth here...I'm better at figuring out and fixing problems than talking about how Rails apps work).

I can verify that the ActiveRecord I got when installing Rails 2.1.2 gives me:

/ruby/gems/1.8/gems/activerecord-2.1.2/lib/active_record/validations.rb:34: :too_long => "is too long (maximum is %d characters)",

Which is what I'm seeing [now, with my fix; before it crashed], and you are not...I don't seem to have the overriding you do, even though the ActiveRecord::Errors.default_error_messages Hash in 10-patches.rb matches what you linked to. I wonder what I'm missing.

Question: isn't the default Rails "too short" message more informative and isn't it better to have that displayed? The fix I've put in, for example, doesn't break messages with no format fields and takes advantage of l() and _l() having an argument glob.

From this respect, I kind of like the « Password » is too short (minimum is 4 characters) I've got now, rather than a less informative version. Although maybe I've introduced some other problem...and why don't I see what you see sort of bothers me.

#8 - 2009-02-02 19:47 - Jean-Philippe Lang

> Which, to my ignorant eye, seems to indicate to use the messages that came with activerecord?

Not exactly, "activerecord_error_too_short" is not the actual error message (as you can see it's not the same as :too_long => "is too long (maximum is %d characters)" that you find in rails code). It's the key that is used to get the translated error message when doing l(msg) in ApplicationHelper#error_messages_for that overrides validation error messages display.

These hacks will be removed in Redmine 0.9 since Rails 2.2 has builtin support for internationalization :-)
But I can apply your patch for 0.8.x releases.

#9 - 2009-02-02 20:23 - Andrew R Jackson

Oh, duh, I see "activerecord_error_too_short" etc in my ./lang/en.yml with the values you mentioned. And in 10-patches.rb you've told it to use those keys and thus those values. Ok, that makes sense, I should have seen that.

I'm still confused why our installs would not be using your values--while I like the messages I have now, it suggests I've skipped some step or config setting, or I've got something wrong. When does 10-patches.rb and such get applied; every start-up of the app or when one does a db:migrate or something else?

Thanks for all the feedback and explanations, I appreciate it, Jean-Philippe.

#10 - 2009-02-20 19:55 - Jean-Philippe Lang

- Status changed from New to Resolved

- Target version set to 0.8.2

Fix applied in r2486.
The reason why this error occurs is still unknown.

#11 - 2009-02-21 20:29 - Jean-Philippe Lang

- Status changed from Resolved to Closed

Merged in 0.8 branch in r2513.

Files

too_short_error.png	3.1 KB	2009-02-02	Jean-Philippe Lang