Ruby - Feature #3491

Pack missing directives for signed types with specified byte-order

06/28/2010 01:04 AM - emok (Erik M)

Status:	Closed	
Priority:	Normal	
Assignee:		
Target version:	1.9.3	

Description

=begin
The Array#pack and String#unpack have some directives where the byte-order is specified ("little endian" or "network order"), for unsigned long and for unsigned short. But when it comes to signed long and signed short there is no way to specify byte order, you can only use the platform-dependent directives.

This issue was mentioned on a mailing list in 2006: http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-talk/186884
and from what I understood it is a bug inherited from the pack and unpack functions in Perl. The conversation ended by matz having talked to the Perl lead programmer about coordination so both languages solve this issue the same way. But then I don't know if anything happened.

A possible solution
http://perldoc.perl.org/perlpacktut.html#Byte-order-modifiers mentions Perl 5.9.2 supporting ">" and "<" to specify byte order for the directives that usually use the platform-dependent order. That seems like a solution! The documentation at http://search.cpan.org/~dapm/perl-5.10.1/pod/perlfunc.pod#pack may be easier to read.

More about current status
The docs for Ruby 1.9 still seem to miss this feature. Also the 'N' and 'V' directives operate on unsigned longs, which I think is not clear from the documentation. For all the platform-dependent formats cCsSiIlL it has pairs like "Unsigned long" and "Long" (which is then signed), but for the platform-indepedent suddenly "Long" means "Unsigned long" (applies to formats nNvV). I think that is confusing.

As explained in http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-talk/165961 it is possible to work around this issue by unpacking with the unsigned directive and then making a function that can convert from unsigned to signed. For Array#pack it seems the function doesn't care if you input signed values, the same directive works for unsigned and signed data when packing.

I'm not sure if this is a bug or a feature request, but to be polite it made it a feature. I tested on ruby 1.8.7 (2010-01-10 patchlevel 249) [i486-linux].
=end

Related issues:

Related to Ruby - Feature #4084: pack should support 64bit network byte order...	**Closed**	**11/24/2010**
Has duplicate Ruby - Feature #3947: Array#pack□□□□□□□□□□□□□□□</>□□□	**Closed**	**10/14/2010**

Associated revisions

Revision 5825359dd87a26d5daf7a604583baa0ab48cc543 - 10/14/2010 01:12 PM - naruse (Yui NARUSE)

- pack.c (pack_pack): support endian modifiers: < and >.
  [ruby-dev:42376] Feature #3491
- pack.c (pack_unpack): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@29496 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 5825359d - 10/14/2010 01:12 PM - naruse (Yui NARUSE)

- pack.c (pack_pack): support endian modifiers: < and >.
  [ruby-dev:42376] Feature #3491
- pack.c (pack_unpack): ditto.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@29496 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

History

#1 - 10/14/2010 10:17 PM - naruse (Yui NARUSE)

- Status changed from Open to Closed

=begin
This issue was solved with changeset r29496.
Yui, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

=end

#2 - 10/15/2010 07:13 AM - naruse (Yui NARUSE)

- Target version set to 1.9.3

=begin
Sorry for late response.

Your point is right and it'a new feature.
I add it and it will be available in Ruby 1.9.3.
=end