Ruby master - Feature #4016

Consider adding Age Unicode property

11/03/2010 07:25 AM - ammar (Ammar Ali)

Status:	Closed	
Priority:	Normal	
Assignee:		
Target version:		

Description

=begin
Please consider adding the Age Unicode property to Oniguruma. This property is useful when it is necessary to determine availability of code points in a given text against a given Unicode version. Latest info: http://unicode.org/Public/UNIDATA/DerivedAge.txt

mini:bin ammar$ ./ruby -v
ruby 1.9.3dev (2010-11-02 trunk 29669) [i386-darwin9.8.0]
mini:bin ammar$ ./ruby -e 'p /\p{age=3.0}/'
-e:1: invalid character property name {age=3.0}: /\p{age=5.2}/
mini:bin ammar$ ./ruby -e 'p /\p{Age=3.0}/'
-e:1: invalid character property name {Age=3.0}: /\p{Age=5.2}/

Thanks.
=end

History

#1 - 11/03/2010 07:33 AM - ammar (Ammar Ali)

=begin
Sorry for the incorrect output, copy paste error on my part. Output should have been:

mini:bin ammar$ ./ruby -v
ruby 1.9.3dev (2010-11-02 trunk 29669) [i386-darwin9.8.0]
mini:bin ammar$ ./ruby -e 'p /\p{age=5.2}/'
-e:1: invalid character property name {age=5.2}: /\p{age=5.2}/
mini:bin ammar$ ./ruby -e 'p /\p{Age=5.2}/'
-e:1: invalid character property name {Age=5.2}: /\p{Age=5.2}/

But I think you get the idea.

Thanks
=end

#2 - 11/04/2010 01:13 PM - naruse (Yui NARUSE)

=begin
2010/11/3 Ammar Ali redmine@ruby-lang.org:

> Please consider adding the Age Unicode property to Oniguruma.
> This property is useful when it is necessary to determine availability of code points in
> a given text against a given Unicode version.
> Latest info: http://unicode.org/Public/UNIDATA/DerivedAge.txt

I agree with your idea and plan to implement it.
If you have a patch, I'll merge it.

--
NARUSE, Yui
naruse@airemix.jp

=end

#3 - 11/05/2010 07:24 PM - duerst (Martin Dürst)

=begin
On 2010/11/04 13:12, NARUSE, Yui wrote:

> 2010/11/3 Ammar Aliredmine@ruby-lang.org:

> Please consider adding the Age Unicode property to Oniguruma.
> This property is useful when it is necessary to determine availability of code points in
> a given text against a given Unicode version.
> Latest info: http://unicode.org/Public/UNIDATA/DerivedAge.txt

I agree with your idea and plan to implement it.
If you have a patch, I'll merge it.

Great, thanks!

I think this is one more step towards making Ruby versions and Unicode
versions more independent. Currently, it's impossible to use a new
Unicode version with an older version of Ruby.

Regards, Martin.

--
#-# Martin J. Dürst, Professor, Aoyama Gakuin University
#-# http://www.sw.it.aoyama.ac.jp mailto:duerst@it.aoyama.ac.jp

=end

#4 - 11/06/2010 07:28 AM - runpaint (Run Paint Run Run)

- File uni-age.patch added

=begin
I've started this. What forms of property name do we want to support?

- \p{age=5.2} / \p{age=6} This is the style suggested by Unicode, but we don't support \p{name=value} for any other properties.
- \p{age 5.2} / \p{age 6} This is what I've implemented.
- \p{5.2} / \p{6} This is consistent with our current convention.

DerivedAge.txt states:
<>

I've implemented this by assigning to each age in DerivedAge.txt the codepoints of that age or younger. However, consider \p{age 4.2}. 4.2 is an invalid age, so we currently raise a SyntaxError, but we could treat this as equivalent to \p{age 4.1}. Which behaviour is preferred?

Anyway, with the current patch applied:

U+0x0220, LATIN CAPITAL LETTER N WITH LONG RIGHT LEG, was assigned in Unicode 3.2:

```
/\p{age 6.0}/u =~ ?\u0220 #=> 0
/\p{age 5.2}/u =~ ?\u0220 #=> 0
/\p{age 5.1}/u =~ ?\u0220 #=> 0
/\p{age 5.0}/u =~ ?\u0220 #=> 0
/\p{age 4.0}/u =~ ?\u0220 #=> 0
/\p{age 4.1}/u =~ ?\u0220 #=> 0
/\p{age 3.2}/u =~ ?\u0220 #=> 0
/\p{age 3.1}/u =~ ?\u0220 #=> nil
/\p{age 3.0}/u =~ ?\u0220 #=> nil
```

U+0x0620, ARABIC LETTER KASHMIRI YEH, was assigned in Unicode 6:

```
/\p{age 6.0}/u =~ ?\u0620 #=> 0
/\p{age 5.2}/u =~ ?\u0620 #=> nil
/\p{age 5.1}/u =~ ?\u0620 #=> nil
/\p{age 5.0}/u =~ ?\u0620 #=> nil
```

When we agree on what syntax should be supported, I'll update the patch and run some tests.
=end

#5 - 11/06/2010 07:46 AM - ammar (Ammar Ali)

- File unicode-age.patch added

=begin

Run Paint Run Run beat me by a few minutes :)

In addition to the syntax possibilities outlined by Run Paint, there is also the possibility of

- \p{age: 5.0} Used by perl, but this is also inconsistent with the current convention.
    - \p{5.2} / \p{6} This is consistent with our current convention.

Even though this style is consistent with the current convention, it lacks enough information about the purpose and could be confusing. The token 'age' should be included in the property, IMHO.

The syntax used in the patch I submitted is \p{age=5.0}. I think the style suggested by Unicode (age=5.0) should be one of the supported styles if several styles are desirable and beneficial. Unfortunately I assumed that the suggested style will be the only one, so my patch hardcodes the '=' into the keywords file. This will require changes in enc/unicode.c to support other styles, for example ': ' => '=', and it might introduce a problem for older versions of gperf, but I'm not sure about the latter.

> DerivedAge.txt states:
> <<Note: When using the Age property in regular expressions,
>
> **an expression such as "\p{age=3.0}" matches all of the code points assigned in Version 3.0--that is, all the code points with a value less than or equal to 3.0 for the Age property.>>**

My patch implements this requirement.

> However, consider \p{age 4.2}. 4.2 is an invalid age, so we currently raise a SyntaxError, but we could treat this as equivalent to \p{age 4.1}. Which behaviour is preferred?

Allowing undefined Unicode ages, or non-existing versions, in the case of 4.2, might cause confusion. Allowing update version numbers for existing version numbers, such as 2.1.8 and 4.0.1 might be useful. In other cases a SyntaxError seems reasonable.

My patch does not modify enc/unicode.c, nor does it include enc/unicode/name2ctype.h (I just deleted it before running ./configure), but it does include enc/unicode/name2ctype.kwd to simplify evaluation. Some basic tests are included.

Thank you for your consideration.
=end

#6 - 11/08/2010 02:31 PM - naruse (Yui NARUSE)

=begin
Run Paint Run Run wrote:

> I've started this. What forms of property name do we want to support?
>
> - \p{age=5.2} / \p{age=6} This is the style suggested by Unicode, but we don't support \p{name=value} for any other properties.

This is suitable.
I want to support this style.

- \p{age 5.2} / \p{age 6} This is what I've implemented.
- \p{5.2} / \p{6} This is consistent with our current convention. They are not acceptable.

> I've implemented this by assigning to each age in DerivedAge.txt the
> codepoints of that age or younger. However, consider \p{age 4.2}.
> 4.2 is an invalid age, so we currently raise a SyntaxError,
> but we could treat this as equivalent to \p{age 4.1}.
> Which behaviour is preferred?

For exmaple invalid property, current ruby raises SyntaxError like following:
So SyntaxError is preferred.

```
% ./ruby -e 'p /\p{XXX}/'
-e:1: invalid character property name {XXX}: /\p{XXX}/
```

Ammar Ali wrote:

> Even though this style is consistent with the current convention,
> it lacks enough information about the purpose and could be confusing.
> The token 'age' should be included in the property, IMHO.

Agree.

> The syntax used in the patch I submitted is \p{age=5.0}.
> I think the style suggested by Unicode (age=5.0) should be one of the
> supported styles if several styles are desirable and beneficial.
> Unfortunately I assumed that the suggested style will be the only one,
> so my patch hardcodes the '=' into the keywords file.
> This will require changes in enc/unicode.c to support other styles,
> for example ': ' => '=',

Mentioned above, supporting key=value style is better,
but this is acceptable.

> and it might introduce a problem for older versions of gperf,
> but I'm not sure about the latter.

This is not a problem because a person who require gperf is only people
who hacks enc/unicode/name2ctype.h.

> > DerivedAge.txt states:
> > <<Note: When using the Age property in regular expressions,
> >
> > **an expression such as "\p{age=3.0}" matches all of the code points assigned in Version 3.0--that is, all the code points with a value less than or equal to 3.0 for the Age property.>>**
>
> My patch implements this requirement.

I'm wandering about this.
Perl supports Age property but it takes a strict interpretation.
http://perldoc.perl.org/perlunicode.html
http://perldoc.perl.org/perluniprops.html

So current implementation is *experimental* and may change in future.
=end

#7 - 11/08/2010 02:37 PM - naruse (Yui NARUSE)

- Status changed from Open to Closed

- % Done changed from 0 to 100

=begin
This issue was solved with changeset r29717.
Ammar, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

=end

Files

uni-age.patch	683 KB	11/06/2010	runpaint (Run Paint Run Run)
unicode-age.patch	98 KB	11/06/2010	ammar (Ammar Ali)