**Ruby master - Feature #2645**

**Have a method in StringScanner which returns the position in characters rather than in bytes**

01/25/2010 07:57 PM - stefanocr (Stefano Crocco)

<table>
<thead>
<tr>
<th>Status</th>
<th>Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee</td>
<td>naruse (Yui NARUSE)</td>
</tr>
<tr>
<td>Target version</td>
<td>2.0.0</td>
</tr>
</tbody>
</table>

**Description**

```
In ruby 1.9, StringScanner#pos returns the position in number of bytes. I read on the ruby mailing list (http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-talk/352809) this happens because working with character-based indexes would be too slow. However, I think it would be nice if StringScanner also provided a method which returned the position in terms of characters (even if it would be slow). As I see it, the situation is the same as with StringScanner#get_byte and StringScanner#getch. I think this would be useful because, when using StringScanner, you're usually interested in the character rather than in bytes.
```

**Related issues:**

Is duplicate of Ruby master - Feature #1159: StringScanner に文字ベースでのインデックスを返すメ... Rejected 02/14/2009

---

**History**

#1 - 01/26/2010 02:30 AM - murphy (Kornelius Kalnbach)

```
+1...but what to name it?

* char_pos
* chpos
* index (like String#index)
```

by the way, the documentation for StringScanner#pos states:

In the 'terminated' position (i.e. the string is exhausted), this value is the length of the string.

This is not true:

```
irb(main):002:0> s = StringScanner.new("äöü"); s.scan(/.*/); s.pos => 6
irb(main):003:0> s.string.length => 3
```

#2 - 01/26/2010 05:06 PM - naruse (Yui NARUSE)

```
- Priority changed from Normal to 3

StringScanner's pos is related to IO#pos.

Feature#1159 is also about this. (but in Japanese)

A problem is:

```
ss = StringScanner.new("äöü")
ss.get_byte
ss.char_pos #=> what is this result?
```

And more, I doubt the use case. Can you tell us more detailed use case?

the documentation for StringScanner#pos states:

In the 'terminated' position (i.e. the string is exhausted), this value is the length of the string.

thanks, I fixed the doc.
```

```
03/19/2020 1/3
```
I had a similar problem: I wanted to extract a part of a StringScanner-backed string.

Consider the following use case:

- The StringScanner ss is used to arrive at a certain position.
- The current position is saved, ie. start_pos = ss.pos.
- ss is used to do some scanning, arriving at a new position: end_pos = ss.pos
- Extracting the string between start_pos and end_pos using ss.string[start_pos..end_pos] does not work in case the range contains multibyte characters.

My work-around is the following:

```ruby
# Extract the part of the StringScanner +strscan+ backed string specified by the +range+. This 
# method works correctly under Ruby 1.8 and Ruby 1.9.
def extract_string(range, strscan)
  result = nil
  if RUBY_VERSION >= '1.9'
    begin
      enc = strscan.string.encoding
      strscan.string.force_encoding('ASCII-8BIT')
      result = strscan.string[range].force_encoding(enc)
      ensure
        strscan.string.force_encoding(enc)
      end
    else
      result = strscan.string[range]
    end
  result
end
```

You can use String#byteslice.

gettalong wrote:

I had a similar problem: I wanted to extract a part of a StringScanner-backed string.

Consider the following use case:

- The StringScanner ss is used to arrive at a certain position.
- The current position is saved, ie. start_pos = ss.pos.
- ss is used to do some scanning, arriving at a new position: end_pos = ss.pos
- Extracting the string between start_pos and end_pos using ss.string[start_pos..end_pos] does not work in case the range contains multibyte characters.

You can use String#byteslice.
- Assignee set to naruse (Yui NARUSE)

#8 - 10/27/2012 05:23 AM - naruse (Yui NARUSE)
- Status changed from Feedback to Rejected