

Ruby master - Feature #4615

Add IO#split and iterative form of String#split

04/26/2011 02:21 PM - yimutang (Joey Zhou)

Status:	Rejected	
Priority:	Normal	
Assignee:	matz (Yukihiro Matsumoto)	
Target version:		
Description		
<pre>=begin file.each_line(sep=\$/) { line ... } can be used to iterate on lines. But the separator is just a string or nil, it cannot be a regexp. Sometimes I may want to iterate on "sentences", which are strings separated by (simply say) punctuations ".,!?". So if I can write it like this: file.split(/[,;!?]/) { sentence ... } I think it will be very convenient. You may say I can write it like this: file.gets(nil).split(/[,;!?]/).each { sentence ... } But this code will: (1) slurp in the whole file; (2) create a temporary array. If the file is a big one, those 2 steps seem both expensive and unnecessary. So I suggest a flexible IO#split: (also available for File and ARGV) io.split(pattern=\$/) { field ...} -> io # default pattern is \$/, not \$; io.split(pattern=\$/) -> enumerator # not array (I think adding a new method is better, rather than modifying the IO#each_line, making it accept regexp as argument.) Well, String#split has only one form: str.split(pattern=\$;, limit=0) -> array Maybe add a iterative form, when with a block: str.split(pattern=\$;, limit=0) { field ... } -> str Joey Zhou =end</pre>		
Related issues:		
Related to Ruby master - Feature #4780: String#split with a block		Closed

History

#1 - 04/26/2011 02:56 PM - naruse (Yui NARUSE)

- Status changed from Open to Assigned
- Assignee set to matz (Yukihiro Matsumoto)

```
=begin
```

```
=end
```

#2 - 04/26/2011 06:13 PM - naruse (Yui NARUSE)

```
=begin
```

Just a thought,

String#split drops a separator.

In this use case, you want to drop the separator?

Anyway on 1.9.2, StringScanner#scan_until seems the one you want.

=end

#3 - 04/27/2011 10:31 AM - yimutang (Joey Zhou)

=begin

Yes, I've made a mistake. The split regexp should be `/(?<=[.!?])/` if I want to iterate on "sentences".

Well, the key points here are: (1)more flexible separator; (2)iterative idiom.

Ruby is just like Perl. `$/` in Perl is just a string too. I saw in perldoc that "the value of `$/` is a string, not a regex. `awk` has to be better for something." Maybe `awk` can set the record separator to a regexp?

So, if there's an `IO#split` or `IO#each` can take a regexp as separator, I think it's powerful.

`IO` and `String` classes have a few same methods: `#each_line` `#each_char` `#each_byte`, maybe a `#split` for `IO` is OK.

and I think `((str.split(pattern) {|filed| ...}))` is a pure Ruby idiom:)

Thank you for telling me that `StringScanner` has such a method. I'm not familiar with the standard libs. Thank you:)

=end

#4 - 04/27/2011 12:32 PM - nobu (Nobuyoshi Nakada)

=begin

Use `scanf.rb`.

=end

#5 - 04/27/2011 06:30 PM - matz (Yukihiro Matsumoto)

- Status changed from Assigned to Rejected

=begin

Use `scanf`, or read then split. Besides that `File#split` does not well describe the method's behavior (read then split). It makes me feel it splits the file contents into several files.

matz.

=end

#6 - 04/27/2011 07:24 PM - yimutang (Joey Zhou)

=begin

Well, how about `((string.split {|filed| ...}))` ?

=end

#7 - 04/27/2011 11:54 PM - nobu (Nobuyoshi Nakada)

=begin

It should be another feature.

=end