String#each_chunk and #chunks

Status: Open
Priority: Normal
Assignee: 
Target version: 

Description
String#each_chunk iterates chunks of specified size in String.
String#chunks is a shorthand for str.each_chunk(n).to_a.

present:
```
str = <<EOS
20190101 20190102
20190103 20190104
EOS

str.scan(/.{1,9}/m) do |chunk|
  p chunk #=> "20190101 
end
```
```
str.scan(/.{1,9}/m) do |chunk|
  chunk.strip!
  p chunk #=> "20190101"
end
```
```
str.scan(/.{1,9}/m) #=> ["20190101 ", "20190102\n", "20190103 ", "20190104\n"]
str.scan(/.{1,9}/m).map(&:strip) #=> ["20190101", "20190102", "20190103", "20190104"]
```

proposal:
```
str = <<EOS
20190101 20190102
20190103 20190104
EOS

str.each_chunk(9) do |chunk|
  p chunk #=> "20190101 
end
```
```
str.each_chunk(9, strip: true) do |chunk|
  p chunk #=> "20190101"
end
```
```
str.chunks(9) #=> ["20190101 ", "20190102\n", "20190103 ", "20190104\n"]
str.chunks(9, strip: true) #=> ["20190101", "20190102", "20190103", "20190104"]
```

History

#1 - 02/06/2019 03:30 AM - shyouhei (Shyouhei Urabe)
Why the String#scan example you showed is not suitable for you? Tell us what makes you happy with the proposal.

#2 - 02/06/2019 12:39 PM - mame (Yusuke Endoh)
I like the proposal itself. I don't think that chunks is a good name, though.

To take every n characters, I often write str.scan(/.{# n}/m), but it looks a bit cryptic. In this case str.chunks(n) is simpler.

I dislike strip: true. It is too ad-hoc. Does it also support lstrip: true, rstrip: true, chop: true, chomp: true, etc? In principle, one method should do one thing, IMO.
I am also not so sure if this feature is needed. But if I wanted such feature, I would ask to let String#scan take similar arguments as String[]. That is, let the first argument point to the starting position, and an optional second argument to be the length. Since we want to capture multiple matches unlike with [], passing a single index for the first argument does not make much sense, but now we have Enumerator::ArithmeticSequence. So we should be able to do

```ruby
str.scan((0..).step(9)) #=> ['20190101
', '20190102
', '20190103
', '20190104
']
str.scan((0..).step(9), 8) #=> ['20190101', '20190102', '20190103', '20190104']
```

This requires more concrete real world example.

Here is a use case


Because I didn't know /....../ should be /....../m I wasted at least 2 hours of debugging.

I wish for both each_chunk or each_slice and/or each_unpack.

I wonder if we should have consistency with slice and each_slice from Array. But honestly, I don't care, just if it's available.

Is size in characters or bytes?

I like String#each_slice and #slices.

Is size in characters or bytes?

Considering consistency with #slice, it is better to have size as characters.

I think String#each_slice(n_chars) would make sense, since it's like str.chars.each_slice(9) { |a| a.join }.

#each_slice and #slices seems fine to me as well; I think it is also a better name than chunks.

I also wanted something like # each_slice.

For example, use it when you want to fix the width of the output.

```ruby
puts "abcdefgijklmnopqrstuvwxyz".each_slice(5).map { |s| "#{s}<br>" }  
# output:
# abcde<br>
# fghij<br>
# klmno<br>
# pqrst<br>
# uvwxy<br>
# z<br>
```

Is size in characters or bytes?

Considering consistency with #slice, it is better to have size as characters.

I think that there may be multiple String#each_slice_xxx like String#each Xxx.

(e.g. Defined String#each_slice_byte , String#each_slice_char and more...)
Also, I think that String#each_slice may be equivalent to String#each_slice_char.

#12 - 08/29/2019 04:43 AM - matz (Yukihiro Matsumoto)
As @shyouhei (Shyouhei Urabe) mentioned, we'd like to hear the real-world use-case. Extracting fixed-width records may be the purpose. I'm curious about the OP's opinion.

Matz.

#13 - 08/29/2019 05:05 AM - usa (Usaku NAKAMURA)
Just an idea, this method may be useful to treat data of fixed-length record format if it accepts multi column lengths, such as

```ruby
records = []
fixed_length_records_data.each_slice(7, 10, 20) do |zip, tel, name|
  records.push({zip: zip, tel: tel, name: name})
end
```

#14 - 12/25/2019 04:28 AM - naruse (Yui NARUSE)
- Target version deleted (2.7)

Files

<table>
<thead>
<tr>
<th>file</th>
<th>size</th>
<th>date</th>
<th>author</th>
</tr>
</thead>
<tbody>
<tr>
<td>patch.diff</td>
<td>6.56 KB</td>
<td>02/06/2019</td>
<td>Glass_saga (Masaki Matsushita)</td>
</tr>
</tbody>
</table>