Ruby master - Feature #707

Documentation for Enumerator chaining

11/03/2008 06:41 PM - candlerb (Brian Candler)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>drbrain (Eric Hodel)</td>
</tr>
<tr>
<td>Target version:</td>
<td>2.0.0</td>
</tr>
</tbody>
</table>

**Description**

```ruby
=begin
Enumerators now support a horizontal filter/chaining pattern:
generator.filter1 { ... }.filter2 { ... }.filter3 { ... }.consumer

The overall pattern for a filter is:

```ruby
Enumerator.new do |y|
  source.each do |input|     # filter INPUT
    ...
  y << output              # filter OUTPUT
end
end
```

This is extremely powerful. However it is not obvious to the newcomer that this is even possible. (Confusion may arise where people know Ruby 1.8's Enumerator, which cannot do this)

So I would just like to see this pattern documented with an example, e.g. under ri Enumerator.new

I have attached a possible example. Note that I have written my Fib generator in a style familiar from ruby-1.8, to emphasise that the Enumerator filter doesn't require a specially-written generator.

=end

**Associated revisions**

Revision a63b86e2 - 02/13/2012 09:19 PM - drbrain (Eric Hodel)

- enumerator.c: Document use of Enumerator.new for creating a lazy enumeration for filtering/chaining. [ruby-trunk - Feature #707]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@34586 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 34586 - 02/13/2012 09:19 PM - drbrain (Eric Hodel)

- enumerator.c: Document use of Enumerator.new for creating a lazy enumeration for filtering/chaining. [ruby-trunk - Feature #707]

Revision 34586 - 02/13/2012 09:19 PM - drbrain (Eric Hodel)

- enumerator.c: Document use of Enumerator.new for creating a lazy enumeration for filtering/chaining. [ruby-trunk - Feature #707]

Revision 34586 - 02/13/2012 09:19 PM - drbrain (Eric Hodel)

- enumerator.c: Document use of Enumerator.new for creating a lazy enumeration for filtering/chaining. [ruby-trunk - Feature #707]

Revision 34586 - 02/13/2012 09:19 PM - drbrain (Eric Hodel)

- enumerator.c: Document use of Enumerator.new for creating a lazy enumeration for filtering/chaining. [ruby-trunk - Feature #707]

Revision 34586 - 02/13/2012 09:19 PM - drbrain (Eric Hodel)

- enumerator.c: Document use of Enumerator.new for creating a lazy enumeration for filtering/chaining. [ruby-trunk - Feature #707]
enumerator.c: Document use of Enumerator.new for creating a lazy enumeration for filtering/chaining. [ruby-trunk - Feature #707]

History

#1 - 11/04/2008 12:05 AM - candlerb (Brian Candler)

=begin
Here is another (shorter and simpler)

class Enumerator
def filter(&blk)
sel.class.new do |y|
each do *[input]
blk.call(y, *input)
end
end
end
end

da = (1..1_000_000_000).to_enum
a.filter [{out,inp} out << inp if inp % 2 == 0 ].
filter [{out,inp} out << inp+100 ].
with_index.each [{inp,c} puts inp; break if c > 10 ]
=end

#2 - 11/29/2008 04:29 PM - ko1 (Koichi Sasada)

- Assignee set to matz (Yukihiro Matsumoto)

=begin
=end

#3 - 11/14/2009 03:04 AM - rogerdpack (Roger Pack)

=begin
Is this also possible in 1.8?

http://www.ruby-forum.com/topic/198804#new
http://github.com/trans/facets/blob/master/lib/core/facets/denumerator.rb

Brian do you think you could write the tutorial on chaining started here:
http://wiki.github.com/rdp/ruby_tutorials_core/enumerator
?
-r
=end

#4 - 11/14/2009 05:01 AM - candlerb (Brian Candler)

=begin

Is this also possible in 1.8?

Sure. Block-form Enumerator is surprisingly straightforward to implement, see
http://github.com/trans/facets/blob/master/lib/core/facets/enumerator.rb

But since 1.9 has it already, I think it's worth documenting the possibilities more clearly.
=end

#5 - 09/14/2010 04:51 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

=begin
=end

#6 - 02/07/2012 11:55 PM - mame (Yusuke Endoh)

- Assignee changed from matz (Yukihiro Matsumoto) to drbrain (Eric Hodel)
Drbrain, could you take over this documentation ticket?

--
Yusuke Endoh mame@tsg.ne.jp

#7 - 02/08/2012 03:38 PM - drbrain (Eric Hodel)
- Priority changed from 3 to Normal

I will work on it, thank you.

#8 - 02/11/2012 11:01 AM - drbrain (Eric Hodel)
- File enumerator.c.lazy.patch added

What do you think about this patch?

I chose not to monkey-patch Enumerator since I don't want the documentation to collide with #4890

Also, I think assigning the lazy enumerators helps illustrate the difference between lazy_select and select (select would never return).

#9 - 02/13/2012 08:06 PM - mame (Yusuke Endoh)

Look good to me. Please go ahead!

--
Yusuke Endoh mame@tsg.ne.jp

#10 - 02/14/2012 06:19 AM - drbrain (Eric Hodel)
- Status changed from Assigned to Closed
- % Done changed from 0 to 100

This issue was solved with changeset r34586.
Brian, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- enumerator.c: Document use of Enumerator.new for creating a lazy enumeration for filtering/chaining. [ruby-trunk - Feature #707]

Files

<table>
<thead>
<tr>
<th>File</th>
<th>Size</th>
<th>Date</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>fib.rb</td>
<td>676 Bytes</td>
<td>11/03/2008</td>
<td>candierb (Brian Candler)</td>
</tr>
<tr>
<td>enumerator.c.lazy.patch</td>
<td>2.7 KB</td>
<td>02/11/2012</td>
<td>drbrain (Eric Hodel)</td>
</tr>
</tbody>
</table>