Ruby - Feature #12623

rescue in blocks without begin/end

07/24/2016 07:27 PM - Nondv (Dmitry Non)

Status:	Closed	
Priority:	Normal	
Assignee:		
Target version:		

Description

Hi there!

There's pretty nice feature that we can use "short rescue" in method definitions:

```
def my_method
  raise '1234'
rescue
   puts 'rescued'
end
```

What about using this in blocks?

```
loop do
  n = enumerator.next
  # do something
rescue StopIteration
end
```

P.S. actually I am not sure if this FR was not created earlier but I couldn't google it.
P.P.S sorry for my english

Related issues:

Is duplicate of Ruby - Feature #7882: Allow rescue/else/ensure in do..end	**Closed**
Has duplicate Ruby - Feature #12906: do/end blocks work with ensure/rescue/else	**Closed**

History

#1 - 07/24/2016 07:29 PM - Nondv (Dmitry Non)

- Description updated

#2 - 07/25/2016 06:18 PM - rosenfeld (Rodrigo Rosenfeld Rosas)

+1, I often want this and never understood why it only worked with methods.

#3 - 07/26/2016 01:59 AM - nobu (Nobuyoshi Nakada)

- Description updated

AFAIK, it has been proposed a few times.

An objection is that rescue in {} block feels weird.

#4 - 07/26/2016 01:00 PM - Nondv (Dmitry Non)

> An objection is that rescue in {} block feels weird.

Do you mean in one-line or multi-line form?

Multiline:

```
list.each { |x|
  # do something
rescue
```

```
  # do something
}
```

In single-line there's ambiguous case:

```
# not implemented
list.each { |x| action_1; action_2; rescue; 'error!' }

# this is valid
list.each { |x| action_1; action_2 rescue  'error!' }
```

But 2nd line looks bad anyway :(

So, what's problem of "feels weird"?
IMHO there're not many people that will use it like that (my first example in single-line).

#5 - 07/27/2016 04:23 AM - shyouhei (Shyouhei Urabe)

Dmitriy Non wrote:

```
list.each { |x|
  # do something
rescue
  # do something
}
```

-1. This is odd. I cannot remember any other language syntax that goes likes this. Java, C++, C# and all other language that use {} as block notations share this syntax to write exception handlings:

```
static void Main()
    {
        try
        {
            // something
        }
        catch (Exception e)
        {
            // something
        }
        finally
        {
            // something
        }
    }
```

#6 - 07/27/2016 07:52 AM - Nondv (Dmitry Non)

> I cannot remember any other language syntax

So, case statement in Ruby is different too.

+ it is not necessary to use this.
BTW AFAIK ruby-style-guide banned multiline {}

#7 - 07/27/2016 08:11 AM - duerst (Martin Dürst)

Nobuyoshi Nakada wrote:

> An objection is that rescue in {} block feels weird.

I feel the same way. I think it feels weird because in Ruby, program structures starting with a keyword (if/while/do/def/...) can contain keywords (else, rescue,...) and end with keywords (end), but symbols ({}, [],...) and keywords are not mixed.

This, combined with the fact that {} is mostly used single-line, may suggest that adding rescue to do blocks might work, but not for {} blocks.

#8 - 08/02/2016 01:52 AM - shyouhei (Shyouhei Urabe)

- Is duplicate of Feature #7882: Allow rescue/else/ensure in do..end added

#9 - 08/04/2016 06:50 AM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Closed

Closing duplicated issue. Please continue discussing at Issue #7882.

#10 - 11/07/2016 02:29 AM - shyouhei (Shyouhei Urabe)

- Has duplicate Feature #12906: do/end blocks work with ensure/rescue/else added

#11 - 12/23/2021 11:41 PM - hsbt (Hiroshi SHIBATA)

- Project changed from 14 to Ruby