Ruby master - Bug #10929
NilClass#to_proc and & don’t mix?
03/02/2015 11:43 PM - trans (Thomas Sawyer)

<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>matz (Yukihiro Matsumoto)</td>
</tr>
<tr>
<td>Target version:</td>
<td></td>
</tr>
<tr>
<td>Backport:</td>
<td>2.0.0: UNKNOWN, 2.1: UNKNOWN, 2.2: UNKNOWN</td>
</tr>
</tbody>
</table>

### Description

This is sort of like a "who would ever care" kind of bug. Nonetheless technically it seems like it is one. So I thought it best that I report it just the same.

```ruby
class NilClass
  def to_proc
    Proc.new{|*x| nil}
  end
end

def f(&b)
  b.call(1)
end
f(&nil)
=> NoMethodError: undefined method `call' for nil:NilClass
```

(Maybe it was fixed already. Filing out this issue reminded me I need to update my version of Ruby.)

### History

**#1 - 03/06/2015 01:00 PM - hanachin (Seiei Miyagi)**

- File block_from_nil.patch added

I wrote a patch for this issue.

```diff
diff --git bootstrap/test_block.rb bootstrap/test_block.rb
index cdc5960..4347fdd 100644
--- bootstrap/test_block.rb
+++ bootstrap/test_block.rb
@@ -611,3 +611,17 @@
   assert_equal 'ok', %q{
     end
   }
+  class NilClass
+    def to_proc
+      Proc.new{|x| nil}
+    end
+  end
+
+  def foo(&blk)
+    blk.call('ok')
+  end
+  def f(&nil)
+    => NoMethodError: undefined method `call' for nil:NilClass
```

(diff --git bootstrap/test_block.rb bootstrap/test_block.rb
index cdc5960..4347fdd 100644
--- bootstrap/test_block.rb
+++ bootstrap/test_block.rb
@@ -611,3 +611,17 @@
   assert_equal 'ok', %q{
     end
   }
+  class NilClass
+    def to_proc
+      Proc.new{|x| x}
+    end
+  end
+  def foo(&blk)
+    blk.call('ok')
+  end
+  def f(&nil)
+    => NoMethodError: undefined method `call' for nil:NilClass
```

09/02/2022
if (proc != Qnil) {
  if (proc != Qnil || rb_respond_to(Qnil, idTo_proc)) {
    if (!rb_obj_is_proc(proc)) {
      VALUE b;
  }

#2 - 03/08/2015 10:43 PM - ko1 (Koichi Sasada)
- Assignee set to matz (Yukihiro Matsumoto)

  I'm not sure we support it or not.
  For me, it is easy to understand that passing nil is special case, passing no block.

#3 - 12/16/2019 03:15 PM - mame (Yusuke Endoh)
- Status changed from Open to Feedback

  The suggested change is harmful. It will break delegation: def foo(&blk); bar(&blk); end; foo().
  Ruby allows redefinition of Integer#/ which is incredibly harmful, so it is possible to allow NilClass#to_proc. But do you really want to allow that?

#4 - 01/14/2020 02:31 AM - ko1 (Koichi Sasada)
- Status changed from Feedback to Rejected

  it can break delegation methods.

Files

block_from_nil.patch  771 Bytes  03/06/2015  hanachin (Seiei Miyagi)