### Ruby master - Feature #6561

**?= operator**

06/09/2012 04:32 AM - slayer (Vlad Moskovets)

<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></td>
</tr>
<tr>
<td>Target version:</td>
<td>2.6</td>
</tr>
</tbody>
</table>

#### Description

I have feature proposition to add new operator ??= its like ||= but assigns only when variable is not defined
So it should be shortcut for

```ruby
@var ??= 1
eq

defined?(@var) ? @var : @var = 1
```

It should useful for caching nil'able or false'able items

#### Related issues:

Is duplicate of Ruby master - Feature #6023: Add "a ?? 2" support for meaning... Rejected 02/15/2012

#### History

**#1 - 06/09/2012 04:47 AM - rosenfeld (Rodrigo Rosenfeld Rosas)**

Related to #6023:

https://bugs.ruby-lang.org/issues/6023

---

**#2 - 06/13/2012 09:58 PM - rosenfeld (Rodrigo Rosenfeld Rosas)**

Would this be valid only for instance variables or any variable?

---

**#3 - 06/13/2012 11:17 PM - nobu (Nobuyoshi Nakada)**

rosenfeld (Rodrigo Rosenfeld Rosas) wrote:

> Would this be valid only for instance variables or any variable?

Only for instance variables, class variables, and global variables. "undefined local variable" makes no sense.

---

**#4 - 06/13/2012 11:36 PM - rosenfeld (Rodrigo Rosenfeld Rosas)**

exactly, I was just wanting to make sure. My other proposition for ??= would work for local variables as well, although it would always evaluate the expression if the current value is null while this proposition will only evaluate it once.

For example, suppose a method like this:

```ruby
def my_method something = nil, something_else = nil
  my_var = something || something_else
  my_var ??= begin
  ...
  end
end
```

Yeah, I know we could do "... || something_else || begin", but #6023 would allow false values to prevent the begin-end block execution.

The other approach would be something like:

```ruby
my_var = begin
  ...
end if my_var.nil?
```

Or
if my_var.nil?
my_var = begin
...
end
end

I'm not strictly defending #6023 because I don't remember needing something like this yet (that's why I couldn't think in a concrete example) as what motivated me to ask for such feature was to be able to cache "false" values in my instance variables, so both #6023 and #6561 would work for me.

I'm just stating what are the major differences between them...

#5 - 10/25/2012 07:37 PM - yhara (Yutaka HARA)
- Target version changed from 2.0.0 to 2.6

#6 - 10/26/2012 11:57 AM - nobu (Nobuyoshi Nakada)
- Status changed from Open to Rejected

I'm not against this feature concept, but '!=' conflicts with existing syntax. You have to propose different syntax.