Ruby master - Feature #7432
Explicit way to define local variable in scope
11/25/2012 04:32 PM - prijutme4ty (Ilya Vorontsov)

<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>3.0</td>
</tr>
</tbody>
</table>

**Description**
Imagine code such as this
```
def test_smth
result = nil
assert_nothing_raised { result = some_slow_calculation }
assert_equal expected_answer, result
end
```
Line `result = nil` means nothing more than our intention to simply bind variable `result` in scope of assertion block with outer local variable. In large methods it can be much heavier to find out this obscured intention.
It may be better to introduce a method (if it can be implemented) or a keyword which simply creates a local variable (probably with nil value just to mark its existence)

```
define_variable 'x' or (define_variable x if it's a keyword)
```

**History**

#1 - 11/25/2012 05:18 PM - charliesome (Charlie Somerville)
This should definitely not be a method, however I would welcome a 'local' keyword for this purpose.

#2 - 11/25/2012 06:17 PM - mame (Yusuke Endoh)
- Status changed from Open to Assigned
- Assignee set to matz (Yukihiro Matsumoto)
- Priority changed from Normal to 3
- Target version set to 3.0

Assigning to matz, but don't hold your breath; matz has rejected such a explicit variable declaration syntax many times.

--
Yusuke Endoh mame@tsg.ne.jp

#3 - 11/25/2012 08:02 PM - trans (Thomas Sawyer)
I have always been curious why there is no dynamic way to create local variables (other then eval).
Eg. `x = 10` might be dynamically written:
```
local :x, 10
```
Since we can create just about anything else dynamically, it seems like stark omission.

#4 - 11/25/2012 09:49 PM - matz (Yukihiro Matsumoto)
- Status changed from Assigned to Rejected

Since I am sick of 'var' and 'local' in other languages, I don't want to add explicit local variable declaration, that requires a new keyword. Introducing a new keyword may break existing programs.

trans (Thomas Sawyer) Your idea would eliminate many chances to optimize.

Matz.

#5 - 11/25/2012 11:11 PM - trans (Thomas Sawyer)
I see why then. Thanks.