Ruby - Bug #6067

Conditional assignment of a nested constant raises a SyntaxError

02/23/2012 12:02 PM - brixen (Brian Shirai)

Status:	Closed		
Priority:	Normal		
Assignee:	matz (Yukihiro Matsumoto)		
Target version:			
ruby -v:	ruby 1.9.3p125 (2012-02-16 revision 34643) [x86_64-darwin10.8.0]	**Backport:**	

Description

I can conditionally assign a simple constant:

$ ruby -v -e 'p X ||= 1'
ruby 1.9.3p125 (2012-02-16 revision 34643) [x86_64-darwin10.8.0]
1

However, conditional assignment of a nested constant raises a SyntaxError:

$ ruby -v -e 'module A; end; p A::X ||= 1'
ruby 1.9.3p125 (2012-02-16 revision 34643) [x86_64-darwin10.8.0]
-e:1: constant re-assignment

But I can assign the constant unconditionally, of course:

$ ruby -v -e 'module A; end; p A::X = 1'
ruby 1.9.3p125 (2012-02-16 revision 34643) [x86_64-darwin10.8.0]
1

Is this a bug?

Thanks,
Brian

Related issues:

Is duplicate of Ruby - Bug #5449: nested constant opasign not working	**Closed**	**10/15/2011**

History

#1 - 03/06/2012 12:09 AM - steakknife (Barry Allard)

One would think it would evaluate similar to:

(A::X = 1 if ! defined? A::X or A::X.nil? ; A::X)

For comparison, these also fail:

```
ruby -e 'module A; end; p A::X = 2; p A::X ||= 1'
ruby -e 'module B module Inner end end; B::Inner::X ||= 4; p B::Inner::X'
```

Also, these correctly succeed:

```
ruby -e 'module F X ||= 2 end; p F::X'
ruby -e 'module G class Foo; X ||= 3 end end; p G::Foo::X'
ruby -e 'module H module Inner end end; H::Inner::X = 4; p H::Inner::X'
```

#2 - 03/11/2012 05:40 PM - ko1 (Koichi Sasada)

- Assignee set to matz (Yukihiro Matsumoto)

#3 - 03/18/2012 06:46 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

#4 - 07/14/2012 04:15 PM - nahi (Hiroshi Nakamura)

Closing as a duplicate, since #5449 was confirmed as a bug by Matz. #5449 will fix it.

#5 - 07/14/2012 04:16 PM - nahi (Hiroshi Nakamura)

- Status changed from Assigned to Closed