Hello,

Please add "is" and "isnt" keywords, which behave like the "==" and "!=" operators respectively, to Ruby. These keywords come from the CoffeeScript language. They are useful to prevent accidental assignment (forgetting a "=" in "==") and they preserve reading order, unlike inverted "if 5 == x" expressions known as "Yoda conditionals".

Thanks for your consideration.

History

#1 - 12/02/2011 07:06 AM - drbrain (Eric Hodel)

```
=end
This would break compatibility with RubyGems:

$ egrep 'bis' lib/rubygems/package/tar_input.rb | grep -v '#'
s = new io, security_policy
yield is
is.close if is
is = zipped_stream entry
Gem::Package::TarReader.new is do |inner|
is.close if is
is = StringIO.new(dis)
is = StringIO.new(zis.inflate(entry.read))
=end
```

#2 - 12/02/2011 07:59 AM - wycats (Yehuda Katz)

I'm not in favor of this proposal, but I don't think it would break these cases, because they are not of the form foo is bar. It would, however, break this case:

```
def foo; end
def is(*); end
```

foo is 1

Yehuda Katz
(ph) 718.877.1325

On Thu, Dec 1, 2011 at 2:07 PM, Eric Hodel drbrain@segment7.net wrote:

Issue #5695 has been updated by Eric Hodel.

```
=end
This would break compatibility with RubyGems:

$ egrep 'bis' lib/rubygems/package/tar_input.rb | grep -v '#'
s = new io, security_policy
yield is
is.close if is
is = zipped_stream entry
Gem::Package::TarReader.new is do |inner|
is.close if is
is = StringIO.new(dis)
is = StringIO.new(zis.inflate(entry.read))
=end
```
Hello,

Please add "is" and "isnt" keywords, which behave like the "==" and "!=" operators respectively, to Ruby. These keywords come from the CoffeeScript language. They are useful to prevent accidental assignment (forgetting a "=" in "==") and they preserve reading order, unlike inverted "if 5 == x" expressions known as "Yoda conditionals".

Thanks for your consideration.

--
http://redmine.ruby-lang.org

#3 - 12/05/2011 08:23 PM - regularfry (Alex Young)
On 01/12/11 18:52, Suraj Kurapati wrote:

Issue #5695 has been reported by Suraj Kurapati.

Wouldn't it make more sense to have "is" use "==="? Then you could say, for instance, "if x is Numeric". "is" could then be syntax sugar for "case...when", and "isnt" could be implemented as the corresponding "else".

--
Alex

#4 - 12/07/2011 12:53 AM - trans (Thomas Sawyer)
Indeed, I have aliased #is to #include and aliased #is? as #=== before

class Collection
  is Enumerable
end

Collection.is? Enumerable

Which I find exceptionally readable.

#5 - 03/28/2012 01:03 AM - mame (Yusuke Endoh)
- Status changed from Open to Rejected
- Priority changed from Normal to 3
I strongly believe that this proposal has no hope.

--
Yusuke Endoh mame@tsg.ne.jp