Ruby master - Bug #13872

Duplicate assignment no longer silences "assigned but unused variable" warning

09/05/2017 05:17 PM - segiddins (Samuel Giddins)

Status:	Closed		
Priority:	Normal		
Assignee:			
Target version:			
ruby -v:	ruby 2.5.0dev (2017-09-04 trunk 59742) [x86_64-darwin16]	**Backport:**	2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN

Description

On ruby 2.4.1:

ruby -W -e 'def a; var = var = "foo"; end'

prints no warning

On ruby 2.5.0dev (2017-09-04 trunk 59742) [x86_64-darwin16]

ruby -W -e 'def a; var = var = "foo"; end'

prints warning -e:1: warning: assigned but unused variable - var

This feature was useful when the local binding would be passed to another scope (like to ERB), and those variables were only accessed via the binding

Associated revisions

Revision f644d3ef - 10/22/2017 01:37 AM - nobu (Nobuyoshi Nakada)

parse.y: workaround for warnings

- parse.y (mark_lvar_used): enable workaround to suppress unused local variables. [ruby-core:82656] [Bug #13872]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@60339 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 60339 - 10/22/2017 01:37 AM - nobu (Nobuyoshi Nakada)

parse.y: workaround for warnings

- parse.y (mark_lvar_used): enable workaround to suppress unused local variables. [ruby-core:82656] [Bug #13872]

Revision 60339 - 10/22/2017 01:37 AM - nobu (Nobuyoshi Nakada)

parse.y: workaround for warnings

- parse.y (mark_lvar_used): enable workaround to suppress unused local variables. [ruby-core:82656] [Bug #13872]

Revision 60339 - 10/22/2017 01:37 AM - nobu (Nobuyoshi Nakada)

parse.y: workaround for warnings

- parse.y (mark_lvar_used): enable workaround to suppress unused local variables. [ruby-core:82656] [Bug #13872]

History

#1 - 09/06/2017 03:43 PM - ted (Ted Johansson)

Previously, foo = foo = "bar" was used as a workaround. Now that the workaround is "fixed", I see incoming PRs changing the workaround to foo = "bar"; foo = foo.

It would be nice if we had a way to handle these false positives directly, without having to resort to tricks. :-)

segiddins (Samuel Giddins) wrote:

> On ruby 2.4.1:
>
> ruby -W -e 'def a; var = var = "foo"; end'

prints no warning

On ruby 2.5.0dev (2017-09-04 trunk 59742) [x86_64-darwin16]

ruby -W -e 'def a; var = var = "foo"; end'

prints warning -e:1: warning: assigned but unused variable - var

This feature was useful when the local binding would be passed to another scope (like to ERB), and those variables were only accessed via the binding

#2 - 09/06/2017 06:29 PM - shevegen (Robert A. Heiler)

Well, the first could be solved if there is some way to also propagate (local) variables towards ERB
or other bindings. Like to have different binding "levels" available. Copy environments and stuff. :D
(Actually ... one could probably store stuff in ENV and have them then propagated? But admittedly that
is also another work around towards anyone just wanting to use local variables instead).

But I think that, to the topic at hand, there may also be cases where people may not have wanted
to do such assignments; could happen due to some quick copy/paste or so. So for these cases, I
think that the 2.5.x behaviour is better, though it also does not seem to distinguish between
same_name_variable = same_name_variable = 'something else' - the "did you mean" gem could also
ask a question such as "did you really mean to give the variable the same name again"? :-)

The "workaround" was probably never intended to be used or be meaningful; at the least I can
not think that matz would think that the above was ... well, revealing the intent.
(self-assignment of variables? By the way, I did not know this workaround), but I agree that
some people may have liked that workaround. Perhaps ruby core may consider if you could add
some alternative proposal without the double assignment?

I mean, if the major use case is in regards to bindings and local variables then perhaps some
other way to use/propagate variables like that could convince matz / the core team?

#3 - 10/16/2017 02:23 AM - matsuda (Akira Matsuda)

This is not a bug. The warning you see is a result of an improvement of "unused variable" detection at r59585.
Now it simply checks whether each variable is actually "used" or not. Please also take a look at #13809 to see what was solved here.
var = var = "foo" warns now because var = var = "foo" doesn't really "use" the variable var.

Please update your code to actually "use" all the assigned local variables, in this case, to var = "foo"; var = var.

#4 - 10/16/2017 04:08 AM - normalperson (Eric Wong)

ronnie@dio.jp wrote:

> Please upgrade your code to actually "use" all the assigned
> local variables, in this case, to var = "foo"; var = var.

The problem is with binding:

```
require 'erb'
foo = :bar
puts ERB.new('<%= foo %>').result(binding)
```

The above uses foo, but binding isn't handled by the parser.
Maybe the parser should not warn if the word binding'
appears in scope? And check for local variables named
binding'...

But yes, this is annoying with ERB when I want my code
to be warning-clean.

#5 - 10/17/2017 12:16 AM - matsuda (Akira Matsuda)

Yes, in my understanding that is exactly the case we use this idiom for, usually when the contents of ERB is user-given and so it's uncertain if the local variable 'foo' will be used there or not.

```
 require 'erb'
-foo = :bar
+foo = :bar; foo = foo
 puts ERB.new('<%= foo %>').result(binding)
```

#6 - 10/17/2017 09:30 AM - nobu (Nobuyoshi Nakada)

There could be:

1. add assignments in user code
2. enable duplicate assignment as a special case
3. add another way to suppress warnings, e.g. magic comment
4. etc

#7 - 10/22/2017 01:37 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Closed

Applied in changeset trunk|r60339.

---

parse.y: workaround for warnings

- parse.y (mark_lvar_used): enable workaround to suppress unused local variables. [ruby-core:82656] [Bug #13872]