Test frameworks and REPLs do not show deprecation warnings by default

Various people in #16345 said that:

The issue can be mitigated if all test frameworks enable all deprecation warnings.
The developer's practice can be supported by tools, such as test frameworks enable deprecation warnings automatically.

And this was used as a base to disable deprecation warnings by default in Ruby 2.7.2.

However, it seems no test frameworks or REPLs actually show deprecation warnings by default!
So Ruby developers will typically never see deprecation warnings for keyword arguments, and it will just break when they try Ruby 3.0.

I think only MSpec does Warning[:deprecated] = true, whether or not -w is passed, which seems the right thing to do.
Currently, RSpec enables Warning[:deprecated] = true only for rspec -w.
Same for test/unit 3.3.4 shipped with 2.7.2.
IRB in 2.7.2 does not show deprecated warnings.
Same for pry.
I think ruby-core needs to have a clear message here, like:

All test frameworks and REPLs should include this snippet and run it before running tests: Warning[:deprecated] = true if Warning.respond_to?(:[]=).
This is important so that developers see warnings in development, and that they see the warnings before updating to the next Ruby version.
Developers can choose to disable deprecation warnings explicitly if they want with Warning[:deprecated] = false.

And I think it would be good that ruby-core makes a PR or an issue to the main test frameworks/REPLs to show examples.

P.S.: if someone wants to disable all warnings with -W0 or $VERBOSE = nil, it will indeed disable them all, including deprecation warnings, so there is no need to check $VERBOSE for setting Warning[:deprecated] = true.

Related issues:
- Related to Ruby master - Feature #16345: Don't emit deprecation warnings by default. added
- Related to Ruby master - Feature #17000: 2.7.2 turns off deprecation warnings... added

History

#1 - 01/29/2021 12:57 PM - Eregon (Benoit Daloze)
- Related to Feature #16345: Don't emit deprecation warnings by default. added

#2 - 01/29/2021 12:57 PM - Eregon (Benoit Daloze)
- Related to Feature #17000: 2.7.2 turns off deprecation warnings... added

#3 - 01/29/2021 01:01 PM - Eregon (Benoit Daloze)
One worrying issue is even if we fix the main test frameworks and REPLs, it won't apply for those shipped in the stdlib in 2.7.2.
But maybe 2.7.3 could pick such changes, so at least users updating to latest 2.7.x would get deprecation warnings in IRB and in test/unit?

#4 - 01/29/2021 09:59 PM - kou (Kouhei Sutou)
test-unit 3.4.0 enables it by default.

09/29/2021
#5 - 02/10/2021 03:59 PM - jeremyevans0 (Jeremy Evans)
- Backport deleted (2.5: UNKNOWN, 2.6: UNKNOWN, 2.7: UNKNOWN, 3.0: UNKNOWN)
- Tracker changed from Bug to Misc

#6 - 02/16/2021 06:56 AM - aycabta (aycabta .)
IRB is often used by beginners for learning purposes, so I disagree.

#7 - 02/16/2021 12:19 PM - Eregon (Benoit Daloze)
kou (Kouhei Sutou) wrote in #note-4:

> test-unit 3.4.0 enables it by default.

Great to hear!

I wonder if it would make sense to include test-unit 3.4.0 in the next Ruby 2.7 release (e.g., 2.7.3)?
Do you know if test/unit users typically use the stdlib version or a newer version from the Gemfile?

I filed an issue for RSpec: [https://github.com/rspec/rspec-core/issues/2867](https://github.com/rspec/rspec-core/issues/2867)

#8 - 02/16/2021 12:26 PM - Eregon (Benoit Daloze)
aycabta (aycabta .) wrote in #note-6:

> IRB is often used by beginners for learning purposes, so I disagree.

I think a warning when using an experimental feature in IRB is a good thing. For example, if one uses Ractor and that later leads to a SEGV, as least there is hint, vs not showing anything, which the user will likely blame on IRB.

IRB is also used by experienced Rubyists to see if a given approach works well. In that case I think it is important to show deprecation warnings, as they might need to reconsider the approach based on that.

Maybe akr (Akira Tanaka) can comment too, since he suggested irb/pry should show deprecation warnings? (in [https://bugs.ruby-lang.org/issues/16345#note-31](https://bugs.ruby-lang.org/issues/16345#note-31))

#9 - 02/16/2021 12:31 PM - Eregon (Benoit Daloze)
I mixed experimental and deprecation warnings in my comment just above, sorry for that. Experimental warnings are already shown in IRB, as they should.

I think deprecation warnings should also be shown in IRB/REPLs in general, if a developer tries some code that is deprecated, they should know about it early, rather than having that code break on the next Ruby version.

#10 - 02/17/2021 08:06 AM - kou (Kouhei Sutou)

I wonder if it would make sense to include test-unit 3.4.0 in the next Ruby 2.7 release (e.g., 2.7.3)?

nagachika (Tomoyuki Chikanaga) What do you think about

> Do you know if test/unit users typically use the stdlib version or a newer version from the Gemfile?

Maybe most users specify gem "test-unit" in their Gemfile. If an user uses Gemfile, the user can't use test-unit without specifying gem "test-unit" in Gemfile. Maybe many users are using Gemfile.

#11 - 02/17/2021 08:47 AM - nagachika (Tomoyuki Chikanaga)

I wonder if it would make sense to include test-unit 3.4.0 in the next Ruby 2.7 release (e.g., 2.7.3)?

nagachika (Tomoyuki Chikanaga) What do you think about

In principle I bump up the version of bundle gems only for security fixes.
The users can specify test-unit in Gemfile if their projects use it, and they should do so. I don't think the bundled test-unit version is not critical in this context.

#12 - 02/17/2021 08:56 AM - nobu (Nobuyoshi Nakada)

kou (Kouhei Sutou) wrote in #note-10:

Maybe most users specify gem "test-unit" in their Gemfile. If an user uses Gemfile, the user can't use test-unit without specifying gem "test-unit" in Gemfile. Maybe many users are using Gemfile.

Off topic:
Bundler (rubygems?) ignores already installed gems (default/bundled/system/user) when Gemfile exists. Is it the intentional behavior?

#13 - 02/17/2021 09:06 AM - kou (Kouhei Sutou)

nobu (Nobuyoshi Nakada) wrote in #note-12:

Off topic:
Bundler (rubygems?) ignores already installed gems (default/bundled/system/user) when Gemfile exists. Is it the intentional behavior?

The ignored gems are not in the Gemfile?
If so, it's intentional. Bundler loads only listed gems.