Ruby developers meeting at RubyKaigi2018

Ruby Committers vs the World

2018/06/01
Attendees: many

Next Meeting

- https://bugs.ruby-lang.org/issues/14769
  - 2018/06/21 (Thu) 14:00-18:00 (JST)
  - At Cookpad Tokyo office
- FYI: Previous meetings: https://bugs.ruby-lang.org/projects/ruby/wiki

Ruby 2.6.0 preview 2 was released

Notable features (since 2.5)

- MJIT
- AST module (What is it? Is there a link?)
- RubyGems 3
- Endless Range: `(1..)`

Notable incompatibility (since 2.5)

- $SAFE is now process global

AST

- Will MRI Ruby get a live Abstract Syntax Tree like Crystal or TruffleRuby? (by Brandon Weaver)

Language tooling support

- Are there any plans to provide APIs from Ruby interpreter for language tooling? Things like auto completion, jump to definition, `NoMethodErrors`, etc. (by Sunny Juneja)

Refinements

- Are there any plans to deprecate/remove refinements in Ruby 3? (Vladimir)

autoload

- Ruby's autoload feature is deprecated (bug #5653) but very useful to load programs faster and keep memory usage down - will there be an alternative with the same functionality in the future? (Thomas)
- a_matsuda: Rails would be broken if autoload is removed. Matz, you must use once
- ko1: no problem if require'ing all files is enough fast

[Feature #14594] Rethink `yield_self`'s name => `then' (usa)

- unak: matz, why do you think that "then" is so good
- ko1: then is a keyword, is the same as "then" of Promise
- unak: I don't like it
- a_matsuda: it's not a verb
- matz: people hated ->, but time solved the problem
- unak: enquete

[Feature #13581] Syntax sugar for method reference (ko1)

- matz: .: is not bad, but there is no good reason to use symbol in this case
- matz: \ is not good, and I don't like other proposals
- ko1: We can exploit SyntaxError
- ktou: can we add any character to the last (not first) of the method name?
- naruse: use escape sequence

result = `cmd` # = %x`cmd`
m =
m = obj.foo
m = obj.foo
m = obj.**foo**

m = obj.foo

m = obj➡️foo
m = obj👉foo
m = %m{ obj.foo }
m = obj.$foo
m = obj
m = obj.`foo`
m = obj.~~foo~~ (do not use this feature :D)
m = objλfoo or obj.λfoo (eregon)

[Feature #14799] Startless range (mame)

- mrkn: In numpy, we can write reverse array as `ary[::-1]`, so I want to write `ary[(..)%-1]` in Ruby.

    - Isn't ary.reverse much clearer?
    - mrkn: reverse is not usable for multi-dimensional array: `ary[:, ::-1, :]`
    - "NumPy DSL"
- User.where(id: (..10)) <- User.where('id <= ?', 10)
- User.where(id: (...10)) <- User.where('id < ?', 10)

[Feature #6394] Support `SO_ORIGINAL_DST` socket option (akr)
[Feature#14696] add optname `SO_ORIGINAL_DST`

- akr: why doesn't glibc provide the constant? kosaki, answer
- kosaki: you can define it by yourself in your application
- akr: indeed it's safe, but unhelpful
- (I cannot follow the discussion)

"`while`" with "`else`" keyword like Python (ko1)

```
while cond
  body
else
  when_condition_is_over
  # skipped if break'ed
end
```

similar to

```
begin
  # body where exceptions are rescued
rescue
  # If exception
else
  # Only if no exception, no exceptions rescued here
end
```

```
begin
  raise
rescue

else

end
if cond
  raise
resuce
```

```
else
  ??
end

def foo

else

end
(warns: -:4: else without rescue is useless)

% all-ruby -e 'def foo
else
end'
...
ruby-1.4.6            /tmp/rbonG6Uo:2: syntax error
                  #<Process::Status: pid 17002 exit 1>
ruby-1.6.0
...
ruby-1.8.7-p374
ruby-1.9.0-0          -e:3: warning: else without rescue is
useless
...
ruby-2.6.0-preview1   -e:3: warning: else without rescue is
useless
ruby-2.6.0-preview2   -e:2: else without rescue is useless
                  #<Process::Status: pid 17323 exit 1>
```