IRB Reboot: Modernize Implementation and Features

ITOYANAGI Sakura

RubyKaigi 2018

Powered by Rabbit 2.2.1 and COZMIXNG
1st day's morning, it was cloudy.
Greeting

2nd day's morning,
it was rainy.
But today...
Greeting

It's a beautiful day outside.
Greeting

Birds are singing, flowers are blooming...
Greeting

On days like these…
IRB Reboot: Modernize Implementation and Features
Let me introduce myself

name

ITOYANAGI Sakura

GitHub

aycabta

maintainer

RDoc
Community: Asakusa.rb

Asakusa.rb every Ruby Tuesday
Company:
Space Pirates, LLC.

Space Pirates, LLC.
Hobby: Climbing

I planned to climb Mt. Zao, it's the highest mountain in Miyagi. The highest mountain is the nearest place to space. It fits for Space Pirates.
Hobby: Climbing

But it's so far from this venue, so I went to **gorge** near here.
Hobby: Climbing

The **gorge** means narrow river between escarpments.
Hobby: Climbing

The Tohoku University official web site provides digging points map for fossils.
Hobby: Climbing

http://www.museum.tohoku.ac.jp/exhibition_info/mini/fossi/locality/locmap.html
A dozen pink points are fossils digging points.
Hobby: Climbing

This venue is hemmed in by many fossils digging points.
Hobby: Climbing

I went to some digging points.
Hobby: Climbing

Waterfall
Hobby: Climbing

Waterfall
Hobby: Climbing

Gorge
Hobby: Climbing

Gorge
Hobby: Climbing

I burst through from 4m to 10m many waterfalls into several kilometers gorge.
I bivouacked in the gorge with a bonfire for cooking rice and miso soup.
Hobby: Climbing

When I was cooking rice and miso soup early morning by bonfire I was given notice "Today, we have Asakusa.rb" so I escaped the gorge quickly and went to Asakusa.rb by bullet train and joined it and went back to Sendai by midnight highway bus,
Hobby: Climbing

in a 24 hours period.
And joined pre-party of RubyKaigi.
Hobby: Climbing

It was the hardest experience of this RubyKaigi.
Hobby: Climbing

In gorge, I didn't find a fossil of the aimed whale, but found so many shell beds (dense shell fossils) and leaf's fossils.
Hobby: Climbing

Shell...?
Today's topic

IRB Reboot: Modernize Implementation and Features
Recent years keiju-san's

In the several past RubyKaigis, keiju-san who is godfather of Ruby and the author of IRB talked about old Ruby.
Recent years keiju-san's

He said

"The first language design of Ruby was like shell."

in "Ruby Archaeology" at RubyKaigi 2013.
Recent years keiju-san's

After that, keiju-san carries on talking about Ruby and shell.
Recent years keiju-san's

2014: Reish, an unix shell for rubyist.

2015: Usage and implementation of Reish which is an Unix shell for Rubyist

2017: Irb 20th anniversary memorial session: Reish and Irb2
Recent years keiju-san's

Next session of Hagi(#rubykaigiC) is keiju-san's one, don't miss it.
Today's topic

Let's back up a minute.
Today's topic

I sent 2 patches to IRB.

#14683 IRB with Ripper

#14787 Show documents when completion
I talked about RDoc with Ripper,
"Ruby Parser
In IRB 20th Anniversary…
Now Let Time Resume"
at RubyKaigi 2017.
Ruby syntax is very complex.
Ruby's parser is spaghetti.

- Lexical analyzer is tightly coupled with parser
- `parse.y` has over 11,000 lines
- The overwhelming weight of syntax to come and the uncertainty of `lex_state`
Ruby's syntax is very dirty.
Ruby's syntax is very dirty.
Ruby's syntax is very complex.
#14683 IRB with Ripper

...How use RDoc use Ripper

Ruby's syntax 😏abrades😏 parser developer.
But, the abradable syntax for parser developer is gentle for Ruby users by matz.

It's great point of Ruby.
So RDoc had very many bugs in parsing Ruby code.
I fixed so many bugs of RDoc, and replaced it. Fixed RDoc with Ripper version.
Ripper is one of Ruby's standard libraries of lexical analysis by parse.y.
I think that Ripper is best way for parsing Ruby code, for following latest Ruby syntax.
But IRB implement pure Ruby parser. It's hard to support Ruby's new syntax.
I thought that Ripper makes IRB's source code parsing better.
I discussed Ruby's REPL with matz, and matz said "I implemented mruby's REPL(mirb), learning from IRB's history".
REPL needs when code block will end(close) because REPL evaluates it at the timing.
The mirb uses token's lex_state parser->lstate syntax error messages parser->error_buffer[0].message
(The parser is a struct mrb_parser_state)
In CRuby, token's lex_state Ripper syntax error messages RubyVM:::InstructionSequence
I ported mirb's implementation to IRB.
IRB has some prompt features, PROMPT_N, PROMPT_S, and %NNi.
#14683 IRB with Ripper

- **PROMPT_N**
  - Prompt when the code line is continued

- **PROMPT_S**
  - Prompt when the code block is in literal

- **%NNi**
  - Nesting level of the code block
#14683 IRB with Ripper

PROMPT_N
- Ripper

PROMPT_S
- Ripper

%NNi
- Ripper
In CRuby, I could resolve the parameters of prompt by Ripper.
The PROMPT_N is a part of "when the code block is ended" logic.
#14683 IRB with Ripper

Inside "splitted sentence", IRB uses PROMPT_N prompt.
Like:

```
method(a, b, c)
```
The PROMPT_S is implemented by checking corresponding open and close tokens of literals.
Literal tokens:

- "
- '
- percent literals
  - %q { and }
  - %w { and }
- blah blah blah
- here-document
Example:

"This is multiline string"
Example:

```ruby
%w{
  array
  of
  strings
}
```
Inside String or other literal, IRB uses PROMPT_S prompt.
The %NNi is implemented by count corresponding name space open and close tokens.
Increase nesting level when takes open token token

- if, unless, while, until, rescue
  - skip post-fix version (it doesn't need end)
- def, do, case, for, begin, class, module
- [, {, (
Decrease down nesting level when takes open token

end

], }, )
class C
  def m
    if true
      1 if true
    end
  end
end

# nesting level is 0
# increase nesting level to 1
# increase nesting level to 2
# increase nesting level to 3
# skip (post-fix if)
# decrease nesting level to 2
# decrease nesting level to 1
# decrease nesting level to 0

# <=== evaluation!
Actual example by default:

```
↓ %NNi (nesting level)
irb(main):001:0> def foo(a,
irb(main):002:2*         b)  # PROMPT_N
irb(main):003:1>     <<-EOM
irb(main):004:1" Hello,
irb(main):005:1" World!
irb(main):006:1" EOM
irb(main):007:1> end
=> :foo
irb(main):008:0>
```
#14683 IRB with Ripper

https://bugs.ruby-lang.org/issues/14683
This removes

- lib/irb/slex.rb (283 lines)
- lib/irb/ruby-token.rb (268 lines)
This simplifies lib/irb/ruby-lex.rb (1181 to 287 lines)
Ruby parser of IRB was shrunk from total 1732 lines to 287 lines.
The simple implementation is best, because Ruby syntax is complex.

The simple implementation is easy to support and keep the gentleness of Ruby.
#14787 Show documents when completion

This is second patch for IRB.
#14787 Show documents when completion

I talked about this Q&A time at RubyKaigi 2017.
First, RDoc's RI binary files are installed to Ruby's directory.
CRuby:

$ tar xzvf ruby-2.5.1.tar.gz
$ cd ruby-2.5.1
$ autoconf
$ ./configure
$ make
$ make install # <=== RDoc runs inside
#14787 Show documents when completion

CRuby:

```
$ rbenv install 2.5.1 # ===> RDoc runs inside
```
RubyGems:

$ gem install rails # ===> RDoc runs inside
#14787 Show documents when completion

But many users set:

$ gem install rails --no-document
#14787 Show documents when completion

Many blogs recommend:

```
$ cat ~/.gemrc
install: --no-document
update: --no-document
```
#14787 Show documents when completion

Unfortunately many users don't need documents data, but I understand it.
#14787 Show documents when completion

Because it's just for RI(r i command).
#14787 Show documents when completion

Usage of class:

```
$ ri 'String'
```
#14787 Show documents when completion

Usage of instance method:

```ruby
$ ri 'String#gsub'
```
Show documents when completion

Usage of class method:

```
$ ri 'String.new'
```
#14787 Show documents when completion

Bothersome.
#14787 Show documents when completion

I wrote on the ticket:

"RDoc installs all documents to Ruby's directory by default."

[cited from `#14787']
Many users never use it because it's just for RI("ri" command).

[cited from `#14787' ]
I think that it is a reason of that many users don't attach importance to documentation.

[cited from `#14787']
#14787 Show documents when completion

I want to improve the importance of RDoc's data.
shevegen (Robert A. Heiler) replied to the ticket:

"I also do not use "ri" on the commandline."

[cited from `#14787 shevegen']
I would not know why, because I myself simply do not use local look-up ways for documentation normally.

[cited from `#14787 shevegen']
It's the same opinion of me.
#14787 Show documents when completion

And the continuation of shevegen's comment:
Show documents when completion

"I really "just google"."

[cited from `#14787 shevegen']
And using the browser is about 100x more convenient for me as well.

[cited from `#14787 shevegen']
It's the exact same opinion of me.
#14787 Show documents when completion

Perfect.
I'm actually sad.
#14787 Show documents when completion

I want to improve the importance of RDoc's data(2).
#14787 Show documents when completion

IRB(with Readline) completes namespace such as classes, modules, methods and so on when it caught TAB key.
In the patch of this ticket, I use RDoc as a library.
When you press TAB key one more just after that namespace is exact matched, RI document is shown.
#14787 Show documents when completion

Demonstration
This is just an aside, I want Ruby's documentation design.
#14787 Show documents when completion

In the ticket, I talked about language documentation design.
Perl has "perldoc" feature and users easily access documents of modules by "perldoc" command.

[cited from `#14787']
Python has "docstring" feature and users can access it on REPL.

[cited from `#14787']
Those are each language's design of importance.

[cited from `#14787'"
Users use the language on the documentation design, so library developers write documents on the documentation design.

[cited from `#14787']
Ruby doesn't have documentation design like Perl and Python.

[cited from `#14787']
Ruby just has RDoc, IRB, and any other supports, but these are just fragmented features, these are not a documentation design.

[cited from `#14787']
I want to improve Ruby's documentation design.
Documentation design

This ticket is a slice of my documentation design.
I want to improve the gentleness of Ruby's documentation design.
I think that the gentleness for users is very important in Ruby.
Documentation design

Please remove --no-document for improvement documentation at Ruby 2.6 or later.
Thank you for your attention

Please write documents!