Ruby master - Feature #15921

R-assign (rightward-assignment) operator

06/14/2019 03:08 AM - nobu (Nobuyoshi Nakada)

Status:	Open	
Priority:	Normal	
Assignee:	matz (Yukihiro Matsumoto)	
Target version:		

Description

From https://bugs.ruby-lang.org/issues/15799#change-78465, proposal of the rightward-assignment operator by =>.

```
$ ./ruby -v -e '(1..).lazy.map {|x| x*2} => x' -e 'p x.first(10)'
ruby 2.7.0dev (2019-06-12T06:32:32Z feature/rassgn-assoc c928f06b79) [x86_64-darwin18]
last_commit=Rightward-assign by ASSOC
[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
```

https://github.com/nobu/ruby/tree/feature/rassgn-assoc

Related issues:

Related to Ruby master - Feature #15799: pipeline operator	**Closed**

History

#1 - 06/14/2019 03:08 AM - nobu (Nobuyoshi Nakada)

- Related to Feature #15799: pipeline operator added

#2 - 06/14/2019 10:49 AM - Hanmac (Hans Mackowiak)

where does the rightward assign works and where it is blocked? y => x might be treated as Hash Parameter

like m y => x is this m(y) => x or still m({y => x})

#3 - 06/14/2019 12:27 PM - nobu (Nobuyoshi Nakada)

This has lower precedence, so the latter.

#4 - 06/16/2019 12:55 AM - ioquatix (Samuel Williams)

There are two areas where I think this is a great addition:

```
x = if foo
    bar
else
    baz
end

if foo
    bar
else
    baz
end => x
```

I prefer the latter, because it avoids messing with the indentation/readability of the if expression.

Additionally, sometimes I find using irb I have made very large expression. In terminal, going to start of line isn't always obvious/easy. So, I wish to save expression, usually I just press enter and then write x = _ to save last result. But when I go back in history to execute statement again, I must make same "hack". So, I wish I can just write:

```
very long query to get list of users => users
```

That way I don't need to think so hard or go back to start of statement. It might also be nice in middle of expressions, e.g.

```
Users.where(active: true) => active_users.where(type: "admin") => admin_users
```

I don't know if such usage is possible or anticipated, I just wanted to show some ideas - for long expressions sometimes I want to check the middle of the expression.

#5 - 06/16/2019 12:55 AM - ioquatix (Samuel Williams)

If it's not clear, previous statement is evaluated like:

```
(Users.where(active: true) => active_users).where(type: "admin") => admin_users
```

#6 - 06/17/2019 12:56 PM - nobu (Nobuyoshi Nakada)

ioquatix (Samuel Williams) wrote:

> If it's not clear, previous statement is evaluated like:
>
> ```
> (Users.where(active: true) => active_users).where(type: "admin") => admin_users
> ```

It can't be higher precedence than ., or it will conflict with other syntaxes too much.
Rather it should be interpreted like as:

```
admin_users = (active_users.where(type: "admin") = Users.where(active: true))
```

Though it is a syntax error at the parenthesis after where currently.

#7 - 07/29/2019 08:26 AM - ko1 (Koichi Sasada)

- Assignee set to matz (Yukihiro Matsumoto)

#8 - 07/29/2019 08:31 AM - nobu (Nobuyoshi Nakada)

nobu (Nobuyoshi Nakada) wrote:

> ioquatix (Samuel Williams) wrote:
>
> > If it's not clear, previous statement is evaluated like:
> >
> > ```
> > (Users.where(active: true) => active_users).where(type: "admin") => admin_users
> > ```
>
> It can't be higher precedence than ., or it will conflict with other syntaxes too much.

You may be able to use |> here.

```
Users.where(active: true) => active_users |> where(type: "admin") => admin_users
```