Ruby master - Feature #16794

Rightward operators

04/16/2020 01:55 PM - Dan0042 (Daniel DeLorme)

Status:	Open
Priority:	Normal
Assignee:	
Target version:	

Description

While reading #15921 (r-assign) and #16670 (reverse pattern matching order) I felt a certain commonality to these and #15799 (pipeline) and I thought maybe there's currently a narrow window of opportunity to adopt an overarching and harmonious syntax for all of these "rightward" operations, rather than ad-hoc syntax for each.

I am **not** suggesting that all of the operators below should be adopted, or implemented at once, or exactly as-is; rather the idea is that the proposals linked above surrounding righward-ness could share elements in order to make them easy to distinguish from previous/existing syntax.

So here go the (wild) ideas:

=|>
rightward assignment: expr =|> var is equivalent to var = expr, also allow rescue =|> err for consistency.

~|>
rightward destructuring assignment / pattern match: expr ~|> pattern is equivalent to expr in pattern, just more natural when doing assignment without a pattern match (i.e. no possibiilty of NoMatchingPatternError). And maybe could be allowed as expression/condition?

|>
pipe to first argument of right-side method: expr |> foo(1) is equivalent to foo(expr,1)

|>>
pipe to last argument of right-side method: expr |>> foo(1) is equivalent to foo(1,expr)

**|>
pipe hash to keyword arguments of right-side method: expr **|> foo(1) is equivalent to foo(1,**expr), ok, I know, crazy idea :-)

Related issues:

Related to Ruby master - Feature #17353: Functional chaining operator	**Open**

History

#1 - 04/17/2020 02:57 AM - shevegen (Robert A. Heiler)

It's an interesting idea. I am not sure if I can easily adjust to the syntax, but this may be best to just ask matz directly (I think he suggested the pipeline operator, and streem kind of also taps into that whole handling of data idea too, so I just think it is best to ask matz what he thinks about it).

One slight concern is that there is a lot of syntax though, and some of the syntax is a bit hard to read/see - in particular that weird **|> - that is almost like an ASCII game, like the old animated moon-buggy if anyone remembers that "game". :-)

Syntax expression should be considered too, in my opinion (remember the old perl-inspired variables, which made it difficult to read some of the perl code and intent behind it, such as e. g. $& $: $. $- $^ ... ok I made a few up, but you get the idea).

#2 - 04/17/2020 02:13 PM - Dan0042 (Daniel DeLorme)

Actually instead of using |> as the common element between rightward operators, maybe just the pipe character would be enough.

|: (because |= is taken)
|~
|>
|>>

|**

#3 - 04/19/2020 07:28 AM - jackmaple (maple jack)

So,in fact,only one pipeline operator is needed to call the method in reverse.

```
puts "hello world"
#Pipeline call
"hello world" |> puts
```

#4 - 04/19/2020 09:01 AM - nobu (Nobuyoshi Nakada)

As for the pipeline operator, it could be more useful in combination with right assignment operators.

```
foo(...).bar(...) => x
|> zot(...) => y
```

is same as

```
y = (x = foo(...).bar(...))
  .zot(...)
```

#5 - 09/10/2020 09:34 PM - avit (Andrew Vit)

For symmetry, we also have <<- for heredocs "take this input for..."

It might make sense to think of ->> as "give this output to..."

#6 - 11/29/2020 10:26 PM - matz (Yukihiro Matsumoto)

- Related to Feature #17353: Functional chaining operator added