Ruby trunk - Feature #14579

Hash value omission

03/06/2018 01:50 PM - shugo (Shugo Maeda)

Status:	Rejected
Priority:	Normal
Assignee:	
Target version:	

Description

How about to allow value omission in Hash literals:

```
x = 1
y = 2
h = {x:, y:}
p h #=> {:x=>1, :y=>2}
```

And in keyword arguments:

```
def login(username: ENV["USER"], password:)
  p(username:, password:)
end

login(password: "xxx") #=> {:username=>"shugo", :password=>"xxx"}
```

Related issues:

Related to Ruby trunk - Feature #11105: ES6-like hash literals	**Rejected**

History

#1 - 03/06/2018 01:50 PM - shugo (Shugo Maeda)

- Related to Feature #11105: ES6-like hash literals added

#2 - 03/06/2018 01:51 PM - shugo (Shugo Maeda)

- File hash_value_omission.diff added

#3 - 03/08/2018 11:04 AM - Eregon (Benoit Daloze)

I find this syntax very confusing.

The two occurrences of password: above means two very different things (required kwarg and auto-hash creation).

For

```
x = 1
y = 2
h = {x:, y:}
```

it looks to me like the values are missing.
I'd prefer a syntax which is different than "key syntax without value", and refers to the variable name used for the value more clearly, like:

```
x = 1
y = 2
h = {x, y}
```

That would also work for the second case like so:

```
def login(username: ENV["USER"], password:)
  p({username, password})
end
```

#4 - 03/08/2018 11:09 AM - Eregon (Benoit Daloze)

Should this also work for non-Symbols keys like:

```
x = 1
y = 2
```

```
h = { "x" => , "y" => }
```

#5 - 03/08/2018 11:30 AM - phluid61 (Matthew Kerwin)

Eregon (Benoit Daloze) wrote:

> I'd prefer a syntax which is different than "key syntax without value", and refers to the variable name used for the value more clearly, like:
>
> ```
> x = 1
> y = 2
> h = {x, y}
> ```

Please no, this is too close to perl's weird handling of lists/hashes. To me it reads like you're trying to write:

```
h = {1=>2}
```

#6 - 03/09/2018 02:55 AM - shevegen (Robert A. Heiler)

I agree with Matthew.

I understand the suggestion trying to make the syntax even more succinct
(less to type) but I think it's one step too much. Ruby already has a
quite condensed syntax.

I think this syntax here, asked by Benoit, is also problematic:

```
h = { "x" => , "y" => }
```

Has a slight "visual" problem, at the least to me. I would expect =>
to "point" to something on the right hand side, which the normal
syntax in hashes, in ruby, requires (unless you use the foo: :bar
syntax notation).

The:

```
h = {x:, y:}
```

to my brain it's indeed a bit confusing because I would normally
expect something on the right side of "foo: ".

#7 - 03/09/2018 08:59 AM - shugo (Shugo Maeda)

Eregon (Benoit Daloze) wrote:

> I'd prefer a syntax which is different than "key syntax without value", and refers to the variable name used for the value more clearly, like:
>
> ```
> x = 1
> y = 2
> h = {x, y}
> ```

I proposed the above syntax in #11105, but it was rejected, and this proposal is alternative.

#8 - 03/15/2018 06:34 AM - matz (Yukihiro Matsumoto)

I prefer this syntax to #11105, but this introduces a Ruby-specific syntax different from ES6 syntax.
Besides that, I don't like both anyway because they are not intuitive (for me).

Matz.

#9 - 03/15/2018 07:49 AM - shugo (Shugo Maeda)

- Status changed from Open to Rejected

matz (Yukihiro Matsumoto) wrote:

> I prefer this syntax to #11105, but this introduces a Ruby-specific syntax different from ES6 syntax.
> Besides that, I don't like both anyway because they are not intuitive (for me).

So I withdraw this proposal.

Files