Allow questionmarks in variable names

07/08/2019 09:18 AM - aquaj (Jérémie Bonal)

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

Description

Hi,

I thought such an issue would've already been discussed but no number of searches allowed me to find a similar request. Feel free to close if I missed a previous refusal.

From time to time, especially when trying to clear up complex conditional logic, I find myself wishing I could add ? to variable names, since I got used to it while naming methods.

For example, currently:

```ruby
if (node? && terminal?) || (halting && (value == halting))
  # ...
end
```

becomes

```ruby
last_node = self.node? && self.terminal?
halt_on_node = halting && (value == halting)
if last_node || halt_on_node
  # ...
end
```

halt_on_node is clear enough, but last_node feels like it would contain a node, instead of expressing its actual purpose ("is the node the last one?").

Right now a developer would have two options as I see them:

1 - extract the conditional to a method def last_node? which can be a bit much if it's the only place this code is called.
2 - rename the variable something like is_last_node, which feels a bit silly since we're in ruby and used to seeing ?s for predicates.

Trying to assign to a questionmarked variable (a? = true) raises a SyntaxError. IMHO, it would make for more coherent design to allow it, just like we do in method names.

I was afraid that variable? would be already parsed as beginning a ternary expression (variable?1:3) but this isn't parsed either, the only thing it's used for is for method calls (a?5 <=> a?7(5)), so this change wouldn't disrupt any current behavior, the expression would just be looked up like any other call instead of only looking up methods.

The only thing I can see with this is that it might raise the issue of allowing ?s in variable names too, which I'm not sure makes a lot of sense (unlike ? which denotes "booleanness", a trait shared by variables and methods alike, I can't see how a variable would be "dangerous").

Related issues:

- Related to Ruby master - Feature #12046: Allow attr_reader :foo? to define in... Rejected
- Related to Ruby master - Feature #5781: Query attributes (attribute methods e... Assigned

History

#1 - 07/08/2019 10:13 AM - zverok (Victor Shepelev)

If this proposal will not cause some serious conflicts, it seems to be also useful for those cases:

```ruby
class Task
  def initialize(name, ready)
    @name, @ready = name, ready
  end

  attr_reader :name
end
```

09/01/2022
def ready?
  @ready
end

Could (ideally) became

class Task
def initialize(name, ready?)
  @name, @ready? = name, ready?
end

  attr_reader :name, :ready?
end

Generally, it will be a step towards identifiers uniformity (because "extract local variable to method" and vice versa is pretty usual refactorings, and for boolean methods foo? is a common and linter-enforced convention, but for boolean locals it is impossible to use).

#2 - 07/08/2019 05:08 PM - shevegen (Robert A. Heiler)

I thought such an issue would've already been discussed but no number of searches allowed me to find a similar request.

I think this has been discussed before - I can not point to another issue request for this, though, so perhaps I misremember. I also don't remember why "?" are not allowed as a trailing part for variables.

IMHO, it would make for more coherent design to allow it, just like we do in method names.

I do not see why this would make the design more "coherent", per se.

I understand the flexibility-idea, but I don't see why this would make the design more "coherent" or "consistent" as such.

  I was afraid that variable? would be already parsed as beginning a ternary expression (variable?1:3) but this isn't parsed either, the only thing it's used for is for method calls (a?5 <=> a?(5)), so this change wouldn't disrupt any current behavior

I think this is a backwards-incompatible change so introduction would require more thought.

  The only thing I can see with this is that it might raise the issue of allowing 's in variable names too, which I'm not sure makes a lot of sense (unlike ' which denotes "booleanness", a trait shared by variables and methods alike, I can't see how a variable would be "dangerous").

I do not see why adding "?" to variable names as such would require of one to also add "!" as well. I often don't fully understand why this is said, either. "?" and "!" do not have the same, and not even opposing use cases either. Their use cases and rationales are different.

As far as for method names, I like "?", but I dislike "!" quite a bit. I make use of "?" a lot but I avoid "!" as much as possible. People use/abuse the latter, in my opinion.

I have seen code like for optparser that goes about:

def foobar
  # do stuff here
end.parse!(ARGV)

I don't like that style of code layout. Then again, what people prefer is a lot up to an individual preference.

zverok wrote:

  Could (ideally) became

class Task
def initialize(name, ready?)
    @name, @ready? = name, ready?
end

attr_reader :name, :ready?
end

I dislike such a "new" style too. :-) By the way, I copy/pasted your example, and you mis-spelled "initialize".

Good thing the did-you-mean gem can warn about this ... I have had similar typos too, usually when writing "initialize". ;-) Generally, it will be a step towards identifiers uniformity

You can of course ask what style is the "best" one; or that ruby should allow any arbitrary syntax, for variable names, methods and so forth.

Personally I much prefer the current behaviour and am slightly against this proposal - but I am not totally against it either as I understand the rationale (for more flexibility that is; not the part about "coherent" or consistency or something abstract like that). If I would have to vote pro/con, though, I'd be more in favour of con.

One drawback of this proposal, if implemented, will be that people would no longer know whether:

foo?

Is a variable; or a method call. Right now they know that it must be a method call. Again, we can see pros/cons either way, but we need to mention cons too.

I am pretty sure that this would be a backwards-incompatible change, though, so this probably has as a target point ruby 4.0 or so ... :P

Ultimately you only have to convince matz about it, but although I can not pinpoint to older issue requests, I think this has come up before; perhaps on the mailing list or something like that.

#3 - 07/09/2019 12:44 AM - mame (Yusuke Endoh)

I couldn't find an existing proposal for a local variable that ends with ?. There are many proposals for "attr_reader :foo?" (#5781#10720#11167 #12046), but all the tickets have been rejected. (#5781 is still open, but I guess it is just forgotten.)

In #5781, matz is explicitly against an instance variable that ends with ?:

    I don't want to allow instance variable names ending "?", just because ? is for predicates, not for variables.

So, the point is that we should allow only local variables but not instance variables. I'm personally a bit negative.

#4 - 07/09/2019 06:53 AM - noniq (Stefan Daschek)

I personally often write code like this:

```ruby
class Foobar
    attr_reader :completed

    def completed?
        @completed
    end
end
```

This feels unnecessary complex to me. In #12046 it was discussed to allow attr_reader :foo? which would internally access @foo?. This change has been rejected by Matz:

    as a rule, attr_reader x creates an instance variable @x, but we cannot have @x?.

I'm personally not very interested in having / using instance variables ending in ?, but if allowing such variable names would result in a feature like #12046 becoming possible, I'm very much in favour of this proposal.

mame wrote:
So, the point is that we should allow only local variables but not instance variables. I'm personally a bit negative.

As described above, for me personally only the instance variable case would be interesting. I don't think that I would ever feel the need to use local variables ending in a question mark.

#5 - 07/09/2019 11:18 PM - aquaj (Jérémie Bonal)

shevegen (Robert A. Heiler) wrote:

I do not see why this would make the design more "coherent", per se.

I guess "design coherence" is a very subjective notion ^^"  
What I meant by that is that in most cases in Ruby, whether an identifier is a local variable or a method call is transparent.  
I've also often heard (especially when talking about why methods are a bit clunky to pass around in Ruby) that it was a conscious design choice that did matter to matz.  
Because of that it feels "natural" (coherent? ;)) to me that the naming rules should be more uniform between local variables and methods.

shevegen (Robert A. Heiler) wrote:

I do not see why adding "?" to variable names as such would require of one to also add "!" as well. I often don't fully understand why this is said, either. "?" and "!" do not have the same, and not even opposing use cases either. Their use cases and rationales are different.

I do think this discussion is a bit out of the scope of the issue, but I wanted to bring it up because I was writing the proposal with this idea of making naming rules more uniform.  
If local variable names end up allowing trailing ?s, I think trailing Is would be the last remaining difference between local variable names and method names. I figured it might be brought up while discussing the issue so I went ahead and mentioned it but I realize now it might just have made the proposal a bit more confusing.

shevegen (Robert A. Heiler) wrote:

I think this is a backwards-incompatible change so introduction would require more thought.

I don't see any behaviour this could break in existing code.  
What are for now method lookups would turn into variable-or-method lookups, and since it's currently impossible to assign to a ?-ended variable name, it would always default to the method lookup (thus keeping the original behavior intact).  
Am I missing something here ?

shevegen (Robert A. Heiler) wrote:

One drawback of this proposal, if implemented, will be that people would no longer know whether:

foo?

Is a variable; or a method call. Right now they know that it must be a method call. Again, we can see pros/cons either way, but we need to mention cons too.

As can be surely gathered from the beginning of this reply: Where you see a drawback, I see a feature here, and it's actually what's motivating this proposal. Once again, everything is a matter of opinion I guess ;)

--

On the matter of allowing it in instance variable names too, I'm personally against it. An instance variable is clearly different from a method call at first glance. I understand the appeal given the code examples that were shown, but this feels like a mean to an end (attr_reader :foo?) and I'm not sure it'd be the best way forward.

#6 - 07/30/2019 03:57 AM - ko1 (Koichi Sasada)

- Assignee set to matz (Yukihiro Matsumoto)

#7 - 08/28/2019 12:52 AM - mrkn (Kenta Murata)

- Related to Feature #12046: Allow attr_reader .foo? to define instance method foo? for accessing @foo added

#8 - 08/28/2019 12:52 AM - mrkn (Kenta Murata)

- Related to Feature #5781: Query attributes (attribute methods ending in "?" mark) added