Ruby master - Feature #11630

possibility to serialize Proc or Lambda

10/28/2015 12:34 PM - lionel_perrin (Lionel PERRIN)

Status:	Open	
Priority:	Normal	
Assignee:		
Target version:		

Description

It would create a reliable alternative to gems like https://github.com/ngty/sourcify and thus makes much easier to implement the ruby-spark gem (https://github.com/ondra-m/ruby-spark).

I assume that it implies the Proc API to include methods like: Proc#get_external_variables.

To keep a seamless behavior of a deserialized Proc, it is important to add also a method like: Proc#has_side_effect?. This method would be return true if the Proc modifies at least one of its parameters or one of the external variables. (I have the feeling that this development might be a duplicate for issues/6806)

One extra feature would be to have these two methods available for Method as well: Method#get_external_variables (returning the instance variables) and Method#has_side_effect?

History

#1 - 10/29/2015 11:32 AM - lionel_perrin (Lionel PERRIN)

The python API around lambda and function objects makes possible to implement serialization of lambda, for instance with https://github.com/apache/spark/blob/master/python/pyspark/cloudpickle.py. I'm looking for a ruby equivalent feature.

In addition to Proc#has_side_effect? and Proc#get_external_variables, one may of course require Proc#source_code.

#2 - 10/29/2015 12:44 PM - Hanmac (Hans Mackowiak)

about external variables, how do you see @vars?

for sample

```
a = 4
proc { a = 5 }.call
p a
```

does change a local variable and cant be serialized because of that and the binding

```
@a = 4
proc { @a = 5 }.call
p @a
```

is that true for this one too because of it? or is that not the problem?

what about using instance_eval?

```
class A; attr_accessor :x; end
a = A.new
a.x = 4
pr = proc { @x = 5 }
a.instance_eval(&pr)

p a.x #=> 5
```

what if its not doing it over binding but over parameters?

```
class A; attr_accessor :x; end
a = A.new
a.x = 4
proc {|b| b.x = 5 }.call(a)

p a.x #=> 5
```

shouldn't that be allowed to serialized because it doesnt need a binding?

#3 - 11/23/2015 01:34 PM - lionel_perrin (Lionel PERRIN)

In my mind, there is no perfect behavior when it comes to serializing closures with bind variables. This being said, it should not prevent us from serializing pure functions.

Since the behavior of a serialized proc with closure can easily be subject to discussions, I suggest that we focus more on the Proc API. My guess is that if we could provide a suitable API, one could implement different kind of serialization. Especially, it should be possible to implement serialization when Proc neither has external variables, nor side effect.

Hans, as you mentioned, it is worth discussing the behavior of Proc#get_external_variables. I don't know the in-depth details of Proc when it comes to binding I understand that they are subtle differences between the two first cases you describe. May be in case 1: Proc#get_external_variables => [a], case 2: Proc#get_external_variables => [binding] ? Does it make sense ?