Ruby master - Feature #6648

Provide a standard API for retrieving all command-line flags passed to Ruby

06/26/2012 07:56 AM - headius (Charles Nutter)

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

Description

=begin
Currently there are no standard mechanisms to get the flags passed to the currently running Ruby implementation. The available mechanisms are not ideal:

- Scanning globals and hoping they have not been tweaked to new settings
- Using external wrappers to launch Ruby
- ???

Inability to get the full set of command-line flags, including flags passed to the VM itself (and probably VM-specific) makes it impossible to launch subprocess Ruby instances with the same settings.

A real world example of this is "((%bundle exec%))" when called with a command line that sets various flags, a la ((%jruby -Xsome.vm.setting --1.9 -S bundle exec%)). None of these flags can propagate to the subprocess, so odd behaviors result. The only option is to put the flags into an env var (((|JRUBY_OPTS|)) or ((|RUBYOPT|))) but this breaks the flow of calling a simple command line.

JRuby provides mechanisms to get all its command line options, but they require calling Java APIs from Ruby's API set. Rubinius provides its own API for accessing comand-line options, but I do not know if it includes VM-level flags as well as standard Ruby flags.

I know there is a (({RubyVM})) namespace in the 2.0 line. If that namespace is intended to be general-purpose for VM-level features, it would be a good host for this API. Something like...

class << RubyVM
def vm_args; end # returns array of command line args *not* passed to the target script

```
def script; end # returns the script being executed...though this overlaps with $0
```

```
def script_args; end # returns args passed to the script...though this overlaps with ARGV, but that is perhaps warranted since ARGV can be modified (i.e. you probably want the original args)
```

end
=end

Related issues:

Is duplicate of Ruby master - Feature #4046: Saving C's **argv and cwd allows...	**Feedback**

History

#1 - 06/26/2012 07:56 AM - headius (Charles Nutter)

Oops, this should be a feature request.

#2 - 06/26/2012 08:26 AM - nobu (Nobuyoshi Nakada)

- Tracker changed from Bug to Feature

I'm positive to the feature, but RubyVM wouldn't be right place.
It is CRuby specific and unexpected to be in other implementations.

#3 - 06/26/2012 08:28 AM - nobu (Nobuyoshi Nakada)

- Description updated

#4 - 09/19/2012 04:09 PM - headius (Charles Nutter)

ARGV is a special class; perhaps ARGV could have the methods?

#5 - 09/20/2012 10:21 AM - headius (Charles Nutter)

I was mistaken...it is ARGF, not ARGV that is a special class. ARGV is a normal array.

Another option: ENV, which is a special Hash-like class. ENV.vm_args, ENV.script, and ENV.script_args aren't bad.

#6 - 10/15/2012 04:50 AM - headius (Charles Nutter)

Ping...we'd still like to have this to be able to build a unifying benchmark tool, which needs to be able to report the actual command-line arguments passed to the runtime. Current tricks are too ugly (parsing ps output, for example), and this would not be difficult to add.

I'm leaning toward ENV having a few special methods for this, but I'm open to other ideas.

#7 - 11/24/2012 08:50 AM - mame (Yusuke Endoh)

- Status changed from Open to Assigned

- Assignee set to matz (Yukihiro Matsumoto)

- Target version set to 2.6

I'm sorry that matz didn't noticed this.

--
Yusuke Endoh mame@tsg.ne.jp

#8 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)

- Target version deleted (2.6)