New method needed to set and get the current recursion limit

<table>
<thead>
<tr>
<th>Status</th>
<th>Rejected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee</td>
<td>ko1 (Koichi Sasada)</td>
</tr>
<tr>
<td>Target version</td>
<td>2.0.0</td>
</tr>
</tbody>
</table>

```
=begin
Hi, I would like to propose the addition of two methods for setting and getting the recursion limit:
recursion_limit(limit)
recursion_limit => returns an Integer value

At this time, I have been experimenting with recursion to compute the nth fibonacci number using ruby 1.9.1 ( i.e ruby 1.9.1p0 (2009-01-30 revision 21907) ). These tests are being ran on a Apple PowerMac G5 2.5 Ghz Quad 4 GB RAM and a MacBook Pro 2.8 GHz Intel Core 2 Duo 4 GB RAM. Also, I was able to successfully produce a result using an iteration which I'm attaching to verify the correctness of the other two algorithms:

action: ./fibonacci-recursive.rb 10000
result:

```

```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
```
653268083615937374790483865268230409824630564318873545443695598274916060620098984183938664527313000888
302692356736131351159792974378544137521305205043770160226475831890652789085515436615958297829882975106
312005754287834532134915256753022783629441540266584402572114343496118002391208270465089293622388354615057765
832291252546039391152820929525285934436420904245428929403910706233889910858410651831738604374707379085526317
6432573399371287193758746897499236058370657428301616374089691784263768242128352581128205163709280893320
999057097290064367426202389783114700504749984592503663356930388383192338678305613643535189213327973292081
337332642652398976392272340786292817795358057099369104917547080893184105146323382174656373212482263830
92103297701648054726243823748624114503093812206564914032751086643394517512116526545631331113140424368548
50106765843493532836956534280717687753283482343455573667197313927462736291082106792807847180352931117167
7892465908993863549327894523777647446192240337686740040213303432974969202832814593341882681768389930720
363349756231171031012919516979460763273758925350772552375947388435404067715555779056450443016640194625
8097221672975861502696844146952034614932291105970676243268515992834709891128470674086200856871350162603120
719031720860940812983215810772820763531866246117282455372085326350757956430072517744315051539609005168
330223941632226408852848523431580515154946224434882493980390505070483482449327453732624657755879089187190836
6205809095947343150052402532709749699531877072437682590741993963226584147498193609285223945039707165431564
2132815768990508578318340491743455627052022356484644951961124603263139709506938264870661132645076650746115
1267752374826219864250307112984411826226610571635150696200986170494525047913781151541399415506712562711
97133252763631939606902895650288268608362241082050562430701974976171121233066073310059947366875

History

#1 - 02/27/2009 12:33 AM - daz (Dave B)

=begin
See if this post helps:

http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-talk/89931

=end

#2 - 02/27/2009 03:32 AM - jredville (Jim Deville)

=begin

-----Original Message-----
From: Dave B [mailto:redmine@ruby-lang.org]
Sent: Thursday, February 26, 2009 7:33 AM
To: ruby-core@ruby-lang.org
Subject: [ruby-core:22545] [Feature #1218]

Issue #1218 has been updated by Dave B.

See if this post helps:

http://blade.nagaokaut.ac.jp/cgi-bin/scat.rb/ruby/ruby-talk/89931

=end

#3 - 02/28/2009 09:27 PM - Anonymous

=begin

Hi Dave, thanks for responding to the post. However, this doesn't resolve the issue that I'm seeing after increasing the stack size.

Should be some class methods for obtaining and setting the recursion limit.

Issue #1218 has been updated by Dave B.

=end

#4 - 03/01/2009 11:23 PM - austin (Austin Ziegler)

=begin

On Sat, Feb 28, 2009 at 7:26 AM, Conrad Taylor redmine@ruby-lang.org wrote:

Issue #1218 has been updated by Conrad Taylor.

=end
Hi Dave, thanks for responding to the post. However, this does resolve the issue that I'm seeing after increasing the stack size. There should be some class methods for obtaining and setting the recursion limit.

You can't change the recursion limit from within the program on all operating systems.

-austin

Austin Ziegler * halostatue@gmail.com * http://www.halostatue.ca/
* austin@halostatue.ca * http://www.halostatue.ca/feed/
* austin@ziegler.ca

#5 - 03/02/2009 05:42 PM - brent (Brent Roman)

setrlimit(RLIMIT_STACK, size_in_bytes)
should do what you want under a POSIX compliant OS.

See the manpage for setrlimit and getrlimit

This could easily be made accessible to Ruby code via a ‘C’ extension.

I have found that 1.8 Ruby is very inefficient in its use of stack-space and made some patches that improve this.

With ulimit -s 8192, the attached little script on gives up after:

1.6.8-mbari8B: 14374 levels
1.8.6-p287: 7198 levels
1.8.7-p72: 7197 levels
1.8.7-p72-mbari8B: 14750 levels
1.8.6-p287-mbari8B: 14750 levels
1.9.0: 7704 levels (but, see note below)

Ruby 1.9 seems completely unaffected by stack size configured via setrlimit (via ulimit -s from the bash shell)

Perhaps 1.9 is allocating the stack on the heap.
If this is the case, you will need some support in the 1.9 core for changing the size of this allocation.

- brent

Usaku NAKAMURA wrote:

Issue #1218 has been updated by Conrad Taylor.

Hi Dave, thanks for responding to the post. However, this does resolve the issue that I'm seeing after increasing the stack size. There should be some class methods for obtaining and setting the recursion limit.

- Conrad

http://redmine.ruby-lang.org/issues/show/1218

http://redmine.ruby-lang.org

--

View this message in context:
Sent from the ruby-core mailing list archive at Nabble.com.

#6 - 03/02/2009 06:01 PM - nobu (Nobuyoshi Nakada)
Hi,

At Mon, 2 Mar 2009 17:41:03 +0900, Brent Roman wrote in [ruby-core:22619]:

Ruby 1.9 seems completely unaffected by stack size configured via setrlimit (via ulimit -s from the bash shell)

Perhaps 1.9 is allocating the stack on the heap.

1.9 uses the default of pthread.

--
Nobu Nakada

#7 - 03/02/2009 10:08 PM - Anonymous

austin (Austin Ziegler) Ziegler - I was able to change these limits within Python using FreeBSD, Linux, and Mac OS X operating systems. The first two are production machines and the last is a development machine. Otherwise, the Python examples would have crapped out as did the Ruby recursive examples. Thus, I'm sure that it's possible on most modern platforms to change these limits within the Ruby language.

#8 - 03/03/2009 01:20 AM - brent (Brent Roman)

I don't do much pthreads programming.
The call you want seems to be:

int pthread_attr_setstacksize(pthread_attr_t *attr, size_t stacksize)
Set the stack size attribute in a thread attributes object.
See:
(The prospective 'C' extension would have to get at the relevant pthread_attr_t)

Here's the little test script I forgot to attach yesterday:

$a=0
def x
    $a += 1
    x
end
begin
    x
    rescue SystemStackError
        puts "After $#{$a} levels"
        puts $!
end

Conrad Taylor-3 wrote:

Issue #1218 has been updated by Conrad Taylor.

austin (Austin Ziegler) Ziegler - I was able to change these limits within Python using FreeBSD, Linux, and Mac OS X operating systems. The first two are production machines and the last is a development machine. Otherwise, the Python examples would have crapped out as did the Ruby recursive examples. Thus, I'm sure that it's possible on most modern platforms to change these limits within the Ruby language.

http://redmine.ruby-lang.org/issues/show/1218
#9 - 03/04/2009 01:44 AM - austin (Austin Ziegler)

On Mon, Mar 2, 2009 at 8:07 AM, Conrad Taylor redmine@ruby-lang.org wrote:

Issue #1218 has been updated by Conrad Taylor.

austin (Austin Ziegler) Ziegler - I was able to change these limits within Python using FreeBSD, Linux, and Mac OS X operating systems. The first two are production machines and the last is a development machine. Otherwise, the Python examples would have cramped out as did the Ruby recursive examples. Thus, I'm sure that it's possible on most modern platforms to change these limits within the Ruby language.

Did you try this on Windows?

I'm not sure it's possible on most modern platforms (and despite what people think and my own personal prejudices, Windows is a modern platform that Rubyists can't afford to ignore).

-austin

Austin Ziegler * halostatue@gmail.com * http://www.halostatue.ca/
  * austin@halostatue.ca * http://www.halostatue.ca/feed/
  * austin@zieglers.ca

#10 - 03/05/2009 07:12 PM - Anonymous

Hi, I do not have access to a Windows machine because I do all my development using Unix environments.

-Conrad

#11 - 09/14/2010 04:37 PM - shyouhei (Shyouhei Urabe)

- Status changed from Open to Assigned

#12 - 02/13/2012 11:38 PM - mame (Yusuke Endoh)

- Status changed from Assigned to Rejected

I'm rejecting this feature ticket because no progress has been made for a long time. See [ruby-core:42391].

I think it is impossible to implement recursion_limit exactly because we cannot know how much stack each C function consumes.

I expected Python's recursion_limit to fail conservatively (i.e., it will raise an exception for big limit even when it is actually possible). But it was worse than I expected. Python caused SEGV when big limit is specified:

import sys
sys.setrecursionlimit(1000000)
def foo():  
  foo()  
  foo()  #=> SEGV

I think that this is not a correct way, at least, in Ruby.

--
Yusuke Endoh mame@tsg.ne.jp

<table>
<thead>
<tr>
<th>Files</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>fibonacci-iterative.rb</td>
<td>373</td>
<td>02/27/2009</td>
<td>Anonymous</td>
</tr>
<tr>
<td>fibonacci-recursive.rb</td>
<td>408</td>
<td>02/27/2009</td>
<td>Anonymous</td>
</tr>
<tr>
<td>fibonacci-tail-recursive.rb</td>
<td>478</td>
<td>02/27/2009</td>
<td>Anonymous</td>
</tr>
</tbody>
</table>