

Ruby trunk - Bug #6292

Incorrect GC cleanup order with C extension classes

04/14/2012 06:16 AM - owensmk (Mike Owens)

Status: Rejected	
Priority: Normal	
Assignee:	
Target version: 1.9.3	
ruby -v: ruby 1.9.3p179 (2012-04-11 revision 35295) [x86_64-linux]	Backport: 2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN
Description	
<p>I have two classes as C extensions. One (the child) holds a reference to the other (parent). The child references parent via <code>rb_ivar_set()</code>. Further, child also registers mark function to mark parent. Despite this, parent is still GC'd before child on process exit. Thus, when dchild's destructor is called (next), its parent ref points to invalid memory, causing potential segfault.</p> <p>Working example code is here:</p> <p>https://github.com/linterra/ruby-c-ext-gc</p> <p>This also occurs if I set child to nil and then call <code>GC.start()</code>.</p> <p>Behavior is same on 1.9.1 through <code>ruby_1_9_3@35295</code>.</p> <p>My understanding of proper behavior is that child's reference to parent (and/or it's mark function) should keep parent destructor from being called first. Neither of these seem to work.</p>	

History

#1 - 04/14/2012 09:16 AM - matz (Yukihiko Matsumoto)

- Status changed from Open to Rejected

The GC sweeper does not keep the reference dependency. If it does, the GC will be far slower, or will consume too much memory. I am sorry but I have to reject this bug report. It's spec.

Every deallocator should not reference other objects.

Matz.

#2 - 04/14/2012 12:42 PM - owensmk (Mike Owens)

I understand the concern. At least I'm clear on the behavior.

matz (Yukihiko Matsumoto) wrote:

The GC sweeper does not keep the reference dependency. If it does, the GC will be far slower, or will consume too much memory. I am sorry but I have to reject this bug report. It's spec.

Every deallocator should not reference other objects.

Matz.