At the time of writing there are many different Ruby implementations ranging from the common ones such as MRI, Jruby and Rubinius to slightly less common ones such as mruby and Topaz.

A recurring problem in all these implementations is that there's no standard API for profiling resource usage. For example, for MRI there's ruby-prof but in order to get garbage collection and memory usage related information you have to patch MRI. As far as I'm aware of Jruby is the only implementation that at this point offers proper tools for profiling your Ruby application as it piggybacks on top of the JVM and all the awesome profiling tools that come with it.

What I'd hereby like to propose is an API for all Ruby implementations that allows developers to profile at least the following:

- Garbage collection information
- Memory usage
- Object allocations

These items are discussed in detail below.

### Topics

#### Garbage Collection

In the past one would have to use REE or install a bunch of patches in order to get GC information such as when it would run, how many objects it processed, etc. What I'd like to see here is the following:

- When the GC starts/stops
- How many objects are freed and how many are left intact
- The time it takes for the GC to run

If applicable there could be more added, this is just what I can think of from the top of my head.

#### Memory Usage

Next to the GC this is a particular important addition. Similar to GC information one would have to install MRI patches to get memory usage information or worse, use something along the lines of the following:

```ruby
# Returns memory usage of the entire process in KB
def memory_usage
  return `ps -o rss= #{Process.pid}`.strip.to_f
end
```

The above code is actually fairly common but also inaccurate and adds a pretty big overhead as a subshell has to be started for every call. It also wouldn't work on systems where ps is not available or would otherwise work differently.

Ideally one should be able to get the memory usage of individual memory calls similar to what you can now achieve using ruby-prof + a set of patches. This
would greatly reduce the amount of time and effort required to hunt down memory
leaks or otherwise memory intensive operations when it's not entirely clear why
something is not performing as well as it should.

Object Allocations

This one goes a bit hand in hand with the memory usage topic. Ideally one
should be able to get the amount of allocated objects per method call in a bit
more accurate way than ObjectSpace currently provides. For example, it would be
nice if one could see that method X allocated 2500 String objects, 2 Array
Objects and 1 Foo::Bar object.

API Design

Code wise I'm not entirely sure how this would look, which is part of why I'm
opening this feature request. I'm thinking out loud here but I was thinking of
something along the lines of the following:

```ruby
require 'profiler/memory'
require 'profiler/allocations'
require 'profiler/gc'

# The same would be used for memory and allocation profiling but using
# different constants (e.g. MemoryProfiler and AllocationProfiler)
GC.enable_profiling

# Do some serious Ruby work here

# Here `result` would contain the profiling results per method call made
# since profiling was enabled.
result = GC.disable_profiling
```

The core idea however is that every Ruby implementation would offer the same
public API (though the internals may differ). This way one could write a tool
that for example displays a graph of the memory usage over time that would work
in all the Ruby implementations. Currently this is not something that's
possible due to the vastly different APIs (or total lack thereof).

Note that this request isn't a "I want this and I want it now" request but more
the start of a discussion about such an API. The amount of time and effort
required to get this to work on the various implementations could easily take
months if not a few years (heavily depending on the amount of people
available). Especially if you consider that some implementations might
currently not even make the required information publically available.

So to cut a long story short: I hereby open the discussion on a common API for
profiling resource usage in the various Ruby implementations.

p.s. Although I'd also be interested in seeing execution time of methods and
such I'm not entirely sure how I'd envision such an API. As such I've left it
out of this request but of course everybody is welcome to discuss/include that
as well.

History

#1 - 07/10/2013 07:53 PM - yorickpeterse (Yorick Peterse)
For those that are curious, this is the patch for ruby-prof that I'm talking
about: https://gist.github.com/YorickPeterse/5944545

#2 - 07/11/2013 06:53 AM - drbrain (Eric Hodel)
In what ways is GC::Profiler insufficient?

#3 - 07/11/2013 05:53 PM - yorickpeterse (Yorick Peterse)
In what ways is GC::Profiler insufficient?

GC::Profiler only provides fairly limited details on garbage collection. It also doesn't provide any information as to what methods actually triggered garbage collection and the associated resources. Instead it provides a listing based on indexes (garbage collection runs I presume) that looks like the following:

<table>
<thead>
<tr>
<th>Index</th>
<th>Invoke Time(sec)</th>
<th>Use Size(byte)</th>
<th>Total Size(byte)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.073</td>
<td>181920</td>
<td>703480</td>
</tr>
<tr>
<td>2</td>
<td>0.140</td>
<td>182000</td>
<td>703480</td>
</tr>
<tr>
<td>3</td>
<td>0.207</td>
<td>182000</td>
<td>703480</td>
</tr>
<tr>
<td>4</td>
<td>0.263</td>
<td>182000</td>
<td>703480</td>
</tr>
</tbody>
</table>

Another issue is that there's no standard on the output here. MRI uses the above format but Rubinius and Jruby each have their own format. This in turn would make it increasingly hard to write something that parses this output, especially considering the recent increase of Ruby implementations and the potential differences in the above output.

Yorick