Ruby trunk - Feature #15287

New TracePoint events to support loading features

11/07/2018 05:19 AM - ko1 (Koichi Sasada)

Status:	Closed
Priority:	Normal
Assignee:	ko1 (Koichi Sasada)
Target version:	2.6

Description

Abstract

I propose the following new TracePoint events:

- loaded (invoked after require/load)
- method_added (invoked after method definition)

Background

Sometimes we need to hook loading iseq. For example, checking loading files and so on.
Also we want to know what kind of methods are defined.

For both purpose, we can use some hook methods such as Module#method_added and so on.
However, defining methods we can override this features.
So that if we have two tools/libraries using this feature, they can be conflicted.

Proposal

Introduce new TracePoint events:

- loaded (invoked after require/load)
- method_added (invoked after method definition)

Also the following methods can be added:

- Active only loaded event:
  - TracePoint#loaded_feature returns feature name.
  - TracePoint#loaded_iseq returns RubyVM::InstructionSequence object (MRI only, internal feature)

Optional proposal

Add class_added alias name for class event.

Associated revisions

Revision 5ac990e8 - 12/06/2018 01:42 PM - ko1 (Koichi Sasada)

script_compiled TracePoint event [Feature #15287]

- vm_trace.c: add script_compiled event. This event invoked after script compiling and before evaluating compiled script. Also the following methods are added:

TracePoint#compiled_instruction_sequence method to get compiled
RubyVM::InstructionSequence instance.

TracePoint#compiled_eval_script method to get compiled script (String)
by *eval methods (return nil if compiling by file).

- vm_trace.c (tracepoint_attr_raised_exception):

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66249 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66249 - 12/06/2018 01:42 PM - ko1 (Koichi Sasada)

script_compiled TracePoint event [Feature #15287]

- vm_trace.c: add script_compiled event. This event invoked after script compiling and before evaluating compiled script. Also the following methods are added:

TracePoint#compiled_instruction_sequence method to get compiled
RubyVM::InstructionSequence instance.

TracePoint#compiled_eval_script method to get compiled script (String)
by *eval methods (return nil if compiling by file).

- vm_trace.c (tracepoint_attr_raised_exception):

Revision 66249 - 12/06/2018 01:42 PM - ko1 (Koichi Sasada)

script_compiled TracePoint event [Feature #15287]

- vm_trace.c: add script_compiled event. This event invoked after script compiling and before evaluating compiled script. Also the following methods are added:

TracePoint#compiled_instruction_sequence method to get compiled
RubyVM::InstructionSequence instance.

TracePoint#compiled_eval_script method to get compiled script (String)
by *eval methods (return nil if compiling by file).

- vm_trace.c (tracepoint_attr_raised_exception):

Revision 7510eef7 - 12/12/2018 03:45 PM - ko1 (Koichi Sasada)

remove compiled_ prefix. [Feature #15287]

- vm_trace.c: remove compiled_ prefix from the following methods:
  - compiled_eval_script
  - compiled_instruction_sequence [Feature #15287]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@66364 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 66364 - 12/12/2018 03:45 PM - ko1 (Koichi Sasada)

remove compiled_ prefix. [Feature #15287]

- vm_trace.c: remove compiled_ prefix from the following methods:
  - compiled_eval_script
  - compiled_instruction_sequence [Feature #15287]

Revision 66364 - 12/12/2018 03:45 PM - ko1 (Koichi Sasada)

remove compiled_ prefix. [Feature #15287]

- vm_trace.c: remove compiled_ prefix from the following methods:
  - compiled_eval_script
  - compiled_instruction_sequence [Feature #15287]

History

#1 - 11/07/2018 08:15 AM - shevegen (Robert A. Heiler)

I love introspection, so .. \o/

#2 - 11/22/2018 08:02 AM - ko1 (Koichi Sasada)

I got Matz's approval except naming "loaded" because the name "loaded" can be after require/load. This proposal is just after compiling (iseq genration) and just before running loading code.

Other possibility:

- iseq_generated (iseq is internal name)
- script_compiled

We need good name...

#3 - 12/05/2018 07:24 AM - ko1 (Koichi Sasada)

Matz agreed about naming of "script_compiled".

If others especially English natives have ideas, please tell me.
If no, I'll commit "script_compiled" event some days later.

I'm reconsidering "method_added" event because there are several other points we change the class/module methods. We already has several hooks provided by method hook:

```
    rb_define_private_method(rb_cClass, "inherited", rb_obj_dummy, 1);
    rb_define_private_method(rb_cModule, "included", rb_obj_dummy, 1);
    rb_define_private_method(rb_cModule, "extended", rb_obj_dummy, 1);
    rb_define_private_method(rb_cModule, "prepended", rb_obj_dummy, 1);
    rb_define_private_method(rb_cModule, "method_added", rb_obj_dummy, 1);
    rb_define_private_method(rb_cModule, "method_removed", rb_obj_dummy, 1);
    rb_define_private_method(rb_cModule, "method_undefined", rb_obj_dummy, 1);
```

Maybe other events we want to introduce. But (this is internal reason) we don't have enough bits to represent them.

There are several ideas:

- (1) add all events above (change internal)
- (2) add one event (extended, for example) and add new method to recognize which kind of extension Ruby did (for example, tp.extension_type #=> :undef)

Maybe we have no time to conclude this spec so I think it is difficult to introduce this feature (method_added) in Ruby 2.6.

#4 - 12/06/2018 01:42 PM - ko1 (Koichi Sasada)

- Status changed from Open to Closed

Applied in changeset trunk|r66249.

---

script_compiled TracePoint event [Feature #15287]

- vm_trace.c: add script_compiled event. This event invoked after script compiling and before evaluating compiled script. Also the following methods are added:

TracePoint#compiled_instruction_sequence method to get compiled
RubyVM::InstructionSequence instance.

TracePoint#compiled_eval_script method to get compiled script (String)
by *eval methods (return nil if compiling by file).

- vm_trace.c (tracepoint_attr_raised_exception):

#5 - 12/10/2018 07:36 AM - ko1 (Koichi Sasada)

English naming question!

As I mentioned in commit log, I introduced the following two methods:

- TracePoint#compiled_instruction_sequence method to get compiled RubyVM::InstructionSequence instance.
- TracePoint#compiled_eval_script method to get compiled script (String) by *eval methods (return nil if compiling by file).

Matz agreed with TracePoint#compiled_instruction_sequence but has question about TracePoint#compiled_eval_script.

Both use prefix compiled_ but we use two meanings: iseq is the result of compilation and eval_script is source string. The result and the source.

Is it natural or do we have better name?

#6 - 12/10/2018 01:38 PM - Eregon (Benoit Daloze)

"compiled" for the source code seems unnatural.
How about #source_code or #source or #eval_script instead?

instruction_sequence already implies "compiled" so I think the prefix is not needed there too.

#7 - 12/10/2018 07:48 PM - tenderlovemaking (Aaron Patterson)

I agree with @eregon. The user has to subscribe to the "compiled_script" event, so the fact that the script has been compiled is already known.

For example:

```
tp = TracePoint.new(:script_compiled) do |e|
  p e.compiled_eval_script
```

```
end

tp.enable

eval <<-eocode
def omglolwut
  p :hello
end
eocode
```

When the tracepoint executes, we already know it's for a "compiled_script". So "compiled_eval_script" seems redundant. Just "eval_script", or "eval_source" seems good enough.

#8 - 12/10/2018 09:11 PM - ko1 (Koichi Sasada)

Eregon (Benoit Daloze) wrote:

> How about #source_code or #source or #eval_script instead?

I named eval_script because we can not get source code other than *eval methods (not by require and load).

> instruction_sequence already implies "compiled" so I think the prefix is not needed there too

We can get ISeq only on compiled_script event. #instruction_sequence seems too general for me, and for example, I'm afraid that users can misused at call/return event (method's ISeq).
This is why I named raised_exception, not exception (on raise event). Not value but returned_value (on return event).

But instruction_sequence and eval_script (evaled_script?) can be enough long to detect specific methods for script_compiled.

#9 - 12/12/2018 05:52 AM - ko1 (Koichi Sasada)

Matz decided to remove compiled_ prefix.
I'll commit it soon.

#10 - 12/27/2018 10:10 PM - Eregon (Benoit Daloze)

I wonder, what purpose do you envision for instruction_sequence?
It is of course MRI-specific (as documented), so I'm not sure how one would use it on other Ruby implementations.

> I named eval_script because we can not get source code other than *eval methods (not by require and load).

That would be a very useful feature though, and could maybe be created lazily when the method is called.
It seems somewhat similar to https://github.com/jruby/jruby/issues/5206#issue-328885424, which also wants to be able to modify the source on the fly.

#11 - 02/06/2019 11:26 PM - Eregon (Benoit Daloze)

Eregon (Benoit Daloze) wrote:

> I wonder, what purpose do you envision for instruction_sequence?
> It is of course MRI-specific (as documented), so I'm not sure how one would use it on other Ruby implementations.

Actually, I think it is suboptimal to have MRI-specific methods returning MRI-specific types (RubyVM::...) in a portable class like TracePoint.
It means TracePoint cannot be supported completely on other Ruby implementations, because e.g., some method like TracePoint#instruction_sequence is MRI specific.