Ruby trunk - Feature #15074

Create 'official' C API documentation on ruby-doc.org

09/05/2018 05:13 AM - v0dro (Sameer Deshmukh)

Status:	Open	
Priority:	Normal	
Assignee:		
Target version:		

Description

Currently working with Ruby C API is basically reading a bunch of random blogs and coming up with a solution based on inputs from various sources (some of which can be out of date).

The only relevant resource for C API docs with practical examples for each API is currently is this website: https://silverhammermba.github.io/emberb/

However the author is sometimes unresponsive and the documentation on some APIs is missing.

The extension.rdoc file in the ruby repo is not complete either.
There is almost no mention of various critical functions for things like string encoding and hashes.

Therefore, I propose we maintain a full C API documentation divided into meaningful sections and host this complete and 'official' documentation on ruby-doc.org website such that it is directly picked up from the Ruby repo (the same way the current docs for the Ruby core library are). We should at least aim to make it as complete and comprehensive as the Python C API docs: https://docs.python.org/3/c-api/index.html

The other benefit of maintaining it this way is that users can easily see changes across Ruby versions. Having good documentation for C APIs is critical for creating fast and efficient scientific computing libraries for Ruby.

The C API docs can subdivided into the following sections. Please feel free to add to the list:

- Introduction
  - Include files
  - Interfacing extensions with Ruby gems using mkmf.
- Garbage Collection
- Basic Ruby functionality
  - Creating classes, constants, instance variables etc.
  - Method calls and constant lookup.
- Global Interpreter Lock
- Interfacing C structs with Ruby
- Interfaces to Ruby core classes
  - String
  - String encodings
  - Hash
  - Float
  - Integer
  - Symbol

History

#1 - 09/05/2018 11:00 AM - shevegen (Robert A. Heiler)

I think this would be nice to have. Matz often said that nobody minds if ruby becomes faster :) - and I think nobody will mind if the documentation becomes better (qualitatively; but also somewhat more documentation in general).

The link you gave is quite good (the one to github); it's not ideal, in my opinion, when external documentation is better than the official one.

Perhaps part of the problem is that the changes to the documentation in

regards to the official documentation come in small steps, such as improving this or that example, adding this or that explanation. We need a little bit of a comprehensive "global view" over the documentation. I understand that you refer mostly to the C API level, but I think we can include this in general. Perhaps we could add some sort of wiki, so that contribution could be made simpler, before these changes can be synced by a core committer to svn/git. The reason I mention wiki is because it lowers the contribution barrier; and it may be much easier for core committers to commit change when these are mentioned on the wiki. This could also help simplify external contributions, e. g. imagine if the author of the above link would add a disclaimer or otherwise approve that (some of) his changes could be integrated into the official documentation, to help improve it. Then we could integrate what is useful into the wiki, before it may eventually be committed "into" ruby itself; a bit like with wikipedia, and the wikipedia discussion-pages of main articles.

It may also be helpful to explore adding a fourth section here at the bugtracker, solely for documentation-related issues (rather than bug, feature or misc). But anyway, I do not want to digress too much from your thread, so in general I agree to your statements in regards to improving the C API documentation in the long run.

#2 - 09/09/2018 02:31 AM - v0dro (Sameer Deshmukh)

Alright. So as a first step can you please tell me how it would be possible to create this page for the C API? Or maybe start with the wiki?

I can send a PR with some preliminary docs once you approve a course of action.

#3 - 09/10/2018 07:23 AM - hsbt (Hiroshi SHIBATA)

At first, ruby-doc.org is not our official web resource. It's a third party resource.

But ruby-doc.org maybe uses documentation generated by rdoc with ruby source code. So, You can add its documentation under the doc directory.

#4 - 09/11/2018 12:38 AM - v0dro (Sameer Deshmukh)

Alright. I think its a matter of adding the extension.rdoc file to the ruby-doc.org sources and then push changes to extension.rdoc in the core ruby repo.

#5 - 09/11/2018 11:01 PM - v0dro (Sameer Deshmukh)

hsbt (Hiroshi SHIBATA) what do you think about writing the docs within C API files like ruby.h, intern.h and extension.h itself so that ruby-doc.org can run doxygen on them to generate the relevant documents? It will keep the documentation up-to-date as well.

#6 - 09/12/2018 09:12 PM - normalperson (Eric Wong)

sameer.deshmukh93@gmail.com wrote:

> hsbt (Hiroshi SHIBATA) what do you think about writing the docs within C API
> files like ruby.h, intern.h and extension.h itself so
> that ruby-doc.org can run doxygen on them to generate the
> relevant documents? It will keep the documentation up-to-date
> as well.

(not hsbt, here) I think that's a better idea since files we import from ccan/* already do that.

What's more important than API documentation (which is too narrow in scope IMHO) is:

a) overall view of core internal data structures
b) maintaining consistency in internal data structures

Understanding APIs is much easier once data structures are understood and consistent.

For example, I just committed my fix for [Bug #15050] in r64703, r64705 and r64706. The cause of the bug was having multiple sources of truth as to what constitutes a "root fiber".

Multiple sources of truth is almost always a sign of bad data structure design as it is difficult-to-maintain for any application or database, leading to bugs. So the goal is to

design and document data structures well, and use that as a foundation for good code.

In my not so humble opinion, trying to understand with code/APIs before data structures is a total waste of time.