As I understand it, the Data class is an internal abstraction, not intended to be exposed to userland like this.

```
ObjectSpace.each_object(Class).select{|c| c < Data} => [Encoding::Converter, NameError::message]
```

Associated revisions

Revision 684bdf61 - 11/29/2017 08:23 AM - nobu (Nobuyoshi Nakada)
object.c: deprecate Data
  - object.c (InitVM_Object): Data is deprecated now. [Feature #3072]

Revision 60930 - 11/29/2017 08:23 AM - nobu (Nobuyoshi Nakada)
object.c: deprecate Data
  - object.c (InitVM_Object): Data is deprecated now. [Feature #3072]

Revision 60930 - 11/29/2017 08:23 AM - nobu (Nobuyoshi Nakada)
object.c: deprecate Data
  - object.c (InitVM_Object): Data is deprecated now. [Feature #3072]

Revision 60930 - 11/29/2017 08:23 AM - nobu (Nobuyoshi Nakada)
object.c: deprecate Data
  - object.c (InitVM_Object): Data is deprecated now. [Feature #3072]

History

#1 - 04/01/2010 04:16 PM - nobu (Nobuyoshi Nakada)
begin
Hi,

At Thu, 1 Apr 2010 08:09:14 +0900, Run Paint Run Run wrote in [ruby-core:29186]:

As I understand it, the Data class is an internal abstraction, not intended to be exposed to userland like this.

```
ObjectSpace.each_object(Class).select{|c| c < Data} => [Encoding::Converter, NameError::message]
```

Sorry, I'm not sure you think what should not be exposed.
Could you elaborate?
The Data class as a superclass. Why does it participate in the class hierarchy? Does it have some semantic relevance I'm unaware of?

A class is a datatype. If one class and another have real identical data structures in common, there should be some relationships between them. Note what I'm talking about is about their data structures; not their behaviours. Ruby has ducky types so class inheritance depends exclusively to their ways of storing data. Data class is there because two T_DATA based classes should have some relationships in between.

Another reason why it is visible to you, is that MRI tend not to hide its internals to chat you. If you have any good reason to hide the class from your eyes, there can be chances for us to do so. But we have not met such reasons.

Yet the two aforementioned classes are, from a user's perspective, as different as can be. What is the relationship between the two that the superclass is maintaining?

The struct RData.

OK, I admit its name is poor. I admit there are many extensions that use Data_Wrap_Struct without inheriting Data class. Things are not going as planned at this point. But the plan itself is what I explained.

And did I mention there are possibilities for changing the situation when you can persuade us?

What is the relationship between the two that the superclass is maintaining?

The struct RData.

OK, I admit its name is poor. I admit there are many extensions that use Data_Wrap_Struct without inheriting Data class. Things are not going as planned at this point. But the plan itself is what I explained.

And did I mention there are possibilities for changing the situation when you can persuade us?
class. Things are not going as planned at this point. But
the plan itself is what I explained.

And did I mention there are possibilities for changing the
situation when you can persuade us?

Regarding the implementation, T_OBJECT and T_DATA methods are
guaranteed not to make any assumption on the internal structure
of their objects. So you can use T_OBJECT for Data_Wrap_Struct
instead. A possible plan would be removal Data class and
replacement by rb_cObject, but I can't estimate how it will
affect widely.

--
Nobu Nakada
=end

#8 - 04/21/2010 12:37 AM - mame (Yusuke Endoh)
- Assignee set to matz (Yukihiro Matsumoto)
- Target version set to 3.0

=begin
Hi,

At least, this is not a bug.
It is not harmful to be able to access Data class, is it?
In addition, we cannot change it soon for the compatibility
reason.

I move this ticket to Feature tracker with changing target.
The name of "Data" is, indeed, very poor.
And it is also arguable why Data is needed.
Let's discuss and refine it towards 2.0.

--
Yusuke Endoh mame@tsg.ne.jp
=end

#9 - 06/02/2010 11:37 AM - runpaint (Run Paint Run Run)

=begin

What is the relationship between the two that the superclass is maintaining?
The struct RData.

Which is utterly irrelevant to the user. ;-) A common superclass should imply common behaviour, not reveal a dubious artifact of an implementation.
FWIW, Yusuke's proposal is agreeable to me, as it is evident that a solution requires serious thought.
=
=end

#10 - 09/14/2010 04:03 PM - shyouhei (Shyouhei Urabe)
- Status changed from Open to Assigned

=begin
= end

#11 - 10/18/2011 09:16 AM - naruse (Yui NARUSE)
- Project changed from Ruby master to 14
- Category deleted (core)
- Target version deleted (3.0)

#12 - 10/23/2011 05:21 PM - naruse (Yui NARUSE)
- Project changed from 14 to Ruby master
#13 - 11/20/2012 08:50 PM - mame (Yusuke Endoh)
- Description updated
- Target version set to 2.6

#14 - 10/25/2017 02:09 PM - mame (Yusuke Endoh)
Matz, I think there are no longer reason to keep Data class. Like Fixnum/Bignum unification, how about deleting it?

#15 - 11/29/2017 08:07 AM - matz (Yukihiro Matsumoto)
Agreed. How about making Data alias to Object.
It may be removed in the future (3.0?).

Matz.

#16 - 11/29/2017 08:23 AM - nobu (Nobuyoshi Nakada)
- Status changed from Assigned to Closed

Applied in changeset trunk/r60930.

---

object.c: deprecate Data

- object.c (InitVM_Object): Data is deprecated now. [Feature #3072]