Ruby master - Feature #16035
Allow non-finalizable objects such as Integer, static Symbol etc in ObjectSpace::WeakMap
08/01/2019 07:02 PM - byroot (Jean Boussier)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>nobu (Nobuyoshi Nakada)</td>
</tr>
<tr>
<td>Target version:</td>
<td></td>
</tr>
</tbody>
</table>

**Description**
This goes one step farther than what @nobu (Nobuyoshi Nakada) did in [https://bugs.ruby-lang.org/issues/13498](https://bugs.ruby-lang.org/issues/13498)
With this patch, special objects such as static symbols, integers, etc can be used as either key or values inside WeakMap. They simply don't have a finalizer defined on them.

This is useful if you need to deduplicate value objects, e.g. some minimal use case:

```ruby
class Money
  REGISTRY = ObjectSpace::WeakMap.new
  private_constant :REGISTRY

  def self.new(amount)
    REGISTRY[amount] ||= super.freeze
  end

  def initialize(amount)
    @amount = amount
  end
end

if Money.new(42).eql?(Money.new(42))
  puts "Same instance"
else
  puts "Different instances"
end
```

This is a very simple example, but more complex examples can create use a dynamically created symbol as deduplication key, etc.

It also removes one weirdness introduced in the mentioned patch:

```ruby
wmap = ObjectSpace::WeakMap.new
wmap["foo".to_sym] = Object.new # works fine with dynamic symbols
wmap[:bar] = Object.new # cannot define finalizer for Symbol (ArgumentError)
```

**Proposed patch:** [https://github.com/ruby/ruby/pull/2313](https://github.com/ruby/ruby/pull/2313)

**Associated revisions**

**Revision a4a19b11 - 08/29/2019 11:40 AM - byroot (Jean Boussier)**
Allow non-finalizable objects in ObjectSpace::WeakMap
[feature #16035]
This goes one step farther than what nobu did in [feature #13498]
With this patch, special objects such as static symbols, integers, etc can be used as either key or values inside WeakMap. They simply don't have a finalizer defined on them.
This is useful if you need to deduplicate value objects

**Revision 79117d4a - 08/29/2019 01:06 PM - nobu (Nobuyoshi Nakada)**
NEWS: [Feature #16035] [ci skip]
Use identhash as WeakMap

As ObjectSpace::WeakMap allows FLONUM as a key, needs the special deal for its hash.  [Feature #16035]

[DOC] Remove outdated note from WeakRef#initialize

The note

Raises an ArgumentError if the given +orig+ is immutable, such as Symbol, Integer, or Float.

has not been true since #2313 (GH-2313, Feature #16035) when @casperisfine enabled storing non-finalizable objects in the underlying ObjectSpace::WeakMap.

On Ruby 2.7+, WeakRef.new(1) + 1 works fine and the result is the expected 2.

History

#1 - 08/29/2019 06:31 AM - ko1 (Koichi Sasada)
42 never be collected. Is it intentional?

#2 - 08/29/2019 06:31 AM - ko1 (Koichi Sasada)
- Status changed from Open to Feedback

#3 - 08/29/2019 06:41 AM - ko1 (Koichi Sasada)
- Status changed from Feedback to Assigned
- Assignee set to nobu (Nobuyoshi Nakada)

sorry, value is collectable (my misunderstand).

#4 - 08/29/2019 06:44 AM - matz (Yukihiro Matsumoto)
I think this is the required behavior for WeakMap to implement a cache for example. Accepted.

Matz.

#5 - 08/29/2019 11:41 AM - byroot (Jean Boussier)
- Status changed from Assigned to Closed

Applied in changeset gilia4a19b114ba94b8f28d5a91ae05d595a516006d5.

Allow non-finalizable objects in ObjectSpace::WeakMap

[feature #16035]

This goes one step farther than what nobu did in [feature #13498]

With this patch, special objects such as static symbols, integers, etc can be used as either key or values inside WeakMap. They simply don't have a finalizer defined on them.

This is useful if you need to deduplicate value objects

#6 - 11/18/2021 12:18 PM - headius (Charles Nutter)
I am behind the times here, but it is worth noting that implementations which cannot guarantee idempotency of fixnum and flonum-ranged Integers and Floats will have trouble implementing the spirit of this change. On JRuby, it is not possible to treat two fixnums created separately as the same object, since the WeakMap implementation needs to compare by object identity and JRuby represents all fixnums and flonums as full objects with their own identities.

#7 - 11/18/2021 12:21 PM - byroot (Jean Boussier)
It's fine, even on MRI you'll get that with BigNum:

```ruby
>> (2**68).equal?(2**68)
=> false
```

#8 - 11/18/2021 02:04 PM - headius (Charles Nutter)

@byroot (Jean Boussier) True, and this has indeed always been a known behavior difference on JRuby that users just deal with (don't use bare integers as keys, mostly). I really just want to point out that this difference extends to key identity in WeakMap and other identity maps.

#9 - 11/19/2021 08:35 PM - Eregon (Benoit Daloze)

We implemented this correctly in TruffleRuby.
It's a little bit tricky as then indeed some wrapper with custom equals() is needed for primitives:
https://github.com/oracle/truffleruby/commit/e839f3f5887bddd04b855a286e9f74443fbcccf53
That means the exact same semantics as equal? on CRuby.