# Ruby master - Feature #17170

## Numeric.zero, Numeric.one

09/15/2020 07:07 PM - foonlyboy (Eike Dierks)

| | | |
|---|---|---|
| **Status:** | Feedback | |
| **Priority:** | Normal | |
| **Assignee:** | | |
| **Target version:** | | |

**Description**

Hi at the ruby team,

I'd like to suggest to enhance Numeric to provide two new class methods, which shall be: zero and one.

- Integer.zero shall be equal to Integer(0)
- Float.zero shall be equal to Float(0)
- BigDecimal.zero shall be equal to BigDecimal(0)
- Complex.zero shall be equal to Complex(0)

Likewise for one, you get the idea.

Numeric already provides #zero?, so Numeric.zero.zero? shall always be true.

I expect this to make code more explicit. And it would save a pair of braces.
(Don't laugh--This really got me here.)

---

Maybe you already considered that for 3.0, which would be a late addition.

Ruby shines in teaching mathematics. You know, we have zero and one there.

I use ruby in the financial realm. For me, it's important to write BigDecimal.zero.

---

I expect that the new API should not break existing code. We could try it with Rails first. Let's ask them.

It would be nice to have that in [Rails] 3.

~eike

### History

#### #1 - 09/16/2020 12:11 AM - shyouhei (Shyouhei Urabe)

*- Status changed from Open to Feedback*

```
% rbenv exec irb
irb(main):001:0> require 'bigdecimal'
=> true
irb(main):002:0> require 'bigdecimal/util'
=> true
irb(main):003:0> require 'complex'
=> false
irb(main):004:0> 0.to_i
=> 0
irb(main):005:0> 0.to_f
=> 0.0
irb(main):006:0> 0.to_r
=> (0/1)
irb(main):007:0> 0.to_c
=> (0+0i)
irb(main):008:0> 0.to_d
=> 0.0
irb(main):009:0> 0.to_d.class
=> BigDecimal
irb(main):010:0>
```

Do these methods work for you?  They need no braces.

#### #2 - 09/16/2020 01:36 AM - sawa (Tsuyoshi Sawada)

*- Description updated*