**Ruby master - Bug #13848**

**BigDecimal.new('200.') raises an exception**

08/29/2017 12:24 PM - ddooherty03 (Daniel Doherty)

<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>mrkn (Kenta Murata)</td>
</tr>
<tr>
<td>Target version</td>
<td>ruby 2.4.1p111 (2017-03-22 revision 58053) [x86_64-linux]</td>
</tr>
<tr>
<td>Backport:</td>
<td>2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN</td>
</tr>
</tbody>
</table>

**Description**

BigDecimal.new('200.') raises an exception:

ArgumentError: invalid value for BigDecimal(): "200."

It should accept this as a valid decimal number.

This requires fixing numbers read in from the wild.

**History**

**#1 - 08/29/2017 05:23 PM - shevegen (Robert A. Heiler)**

I believe you mean for BigDecimal.new('200.') to be equivalent to BigDecimal.new('200.0')?

I could see both ways be ok; there may be use cases where the error is better because of the user not being explicit enough; or the user does not want to have to type in the trailing '0'. So I can't say whether this is really a bug or not, it may be specifically wanted to be that way.

I guess if there are no trade offs though, then I'd agree with the proposal if you meant that it is the same to mean for a trailing '0' there - but I could also see for purity to require the '0' to be passed.

What about input such as:

```ruby
BigDecimal.new('0."
```

I mean it will be 0 anyway... it reads strangely to omit the the '0' to my eyes though ...

Latest docu can be found here, if anyone else is interested:

https://ruby-doc.org/stdlib/libdoc/bigdecimal/rdoc/BigDecimal.html

**#2 - 08/30/2017 11:21 AM - ddooherty03 (Daniel Doherty)**

Robert, yes, I think it should be the same as BigDecimal.new('200.0'). The problem arises, not when a programmer is typing the number—requiring a programmer to be explicit is OK by me—but when strings are processed that come from a user of the code.

```ruby
ss = '200.' # Read from a user-prepared file.
BigDecimal.new(ss) => ArgumentError ...
```

As it is, the programmer has to massage every string to recognize a reasonable, albeit a bit sloppy, string that is clearly meant as a valid number.

Also, it seems crazy to me that

```ruby
BigDecimal.new('200hello world') => 0.2e3
```

but BigDecimal('200.') raises an ArgumentError exception. Just sayin...

Regards,

**#3 - 09/07/2017 12:26 AM - kernigh (George Koehler)**

Float('200.0') also raises ArgumentError. In this way, BigDecimal.new seems consistent with Float. But it's strange that Float('200hello world') raises ArgumentError but BigDecimal.new('200hello world') succeeds.
#4 - 12/13/2017 03:26 PM - mrkn (Kenta Murata)
- Status changed from Open to Assigned
- Assignee set to mrkn (Kenta Murata)
- Target version set to 2.6

I'm working this bug at [https://github.com/ruby/bigdecimal/pull/87](https://github.com/ruby/bigdecimal/pull/87)

#5 - 12/25/2017 06:15 PM - naruse (Yui NARUSE)
- Target version deleted (2.6)

#6 - 12/07/2018 02:24 AM - mrkn (Kenta Murata)
- Status changed from Assigned to Closed

Fixed in r66222