

Ruby - Bug #13152

Numeric parsing differences between ruby <-> crystal

01/23/2017 07:26 PM - jzakiya (Jabari Zakiya)

Status:	Rejected	
Priority:	Normal	
Assignee:		
Target version:		
ruby -v:		Backport: 2.2: UNKNOWN, 2.3: UNKNOWN, 2.4: UNKNOWN
Description I noticed this difference between ruby and crystal when converting a gem. <pre>puts -2**4 -> -16 (ruby) 16 (crystal) puts (-2)**4 -> 16 (both)</pre> ruby parses <code>-2**4</code> as <code>-(2**4)</code>, while crystal does <code>(-2)**4</code>, which is more intuitive. This creates need to be careful converting negative number usage from ruby <-> crystal. (I haven't investigated differences with other languages.) Using parentheses to explicitly create intended outcomes can overcome this. However, on the heels of the discussion/decision to not change the default rounding behavior of numerics in 2.4.0 would it also be worth it to change this parsing behavior to make this more natural and intuitive, as in crystal?		
Related issues:		
Has duplicate Ruby - Bug #16677: Negative integer powered (**) to a float num...		Closed
Has duplicate Ruby - Bug #18188: <code>-1 ** 0</code> is 1 not -1		Rejected

History

#1 - 01/23/2017 07:29 PM - jzakiya (Jabari Zakiya)

I noticed this difference between ruby and crystal when converting a gem.

```
puts -2**4 -> -16 (ruby) || 16 (crystal)
puts (-2)**4 -> 16 (both)
```

ruby parses `-2**4` as `-(2**4)`, while crystal does `(-2)**4`, which is more intuitive. This creates need to be careful converting negative number usage from ruby <-> crystal. (I haven't investigated differences with other languages.) Using parentheses to explicitly create intended outcomes can overcome this.

However, on the heels of the discussion/decision to not change the default rounding behavior of numerics in 2.4.0 would it also be worth it to change this parsing behavior to make this more natural and intuitive, as in crystal?

#2 - 01/23/2017 11:49 PM - duerst (Martin Dürst)

- Status changed from Open to Feedback

A few questions:

- What is the result if this expression is written as a formula (i.e. with the exponent as a superscript)? Wouldn't this be the more 'intuitive' behavior?
- What would the backwards compatibility issues be for Ruby when changing this? (I guess there might be quite a few subtle bugs.)
- What do other languages (in particular languages more well known than Crystal) do?
- Wouldn't it be quite easy to automate this part of the conversion?
- Why do I think that the sentence "However, on the heels of the discussion/decision to not change the default rounding behavior of numerics in 2.4.0 would it also be worth it to change this parsing behavior to make this more natural and intuitive, as in crystal?" doesn't make sense? On the heels of not making a backwards incompatible change, would it really be worth to make another backwards incompatible change?

#3 - 01/24/2017 05:17 AM - nobu (Nobuyoshi Nakada)

- Description updated

- Status changed from Feedback to Rejected

It is intentional.

That unary minus is not considered a part of an integer literal.

#4 - 01/24/2017 05:26 AM - shyouhei (Shyouhei Urabe)

Martin Dürst wrote:

- What do other languages (in particular languages more well known than Crystal) do?

Here we go:

```
zsh % /bin/sh
sh-3.2$ echo $((-2 ** 4))
16
sh-3.2$
```

```
zsh % perl -e 'warn(eval("-2 ** 4"))';
-16 at -e line 1.
```

```
zsh % php -a
Interactive shell
```

```
php > $a = -2 ** 4; echo $a;
-16
php >
```

```
zsh % python3
Python 3.6.0 (default, Dec 24 2016, 00:01:50)
[GCC 4.2.1 Compatible Apple LLVM 8.0.0 (clang-800.0.42.1)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> -2 ** 4
-16
>>>
```

```
zsh % /Users/urabe.shyouhei/Applications/Julia-0.5.app/Contents/Resources/julia/bin/julia
```

```

 _ _(_) _ | A fresh approach to technical computing
(_) | ( ) ( ) | Documentation: http://docs.julialang.org
 _ _ _ | | _ _ _ | Type "?help" for help.
| | | | | | | / _ \ | |
| | | | | | | ( | | | Version 0.5.0 (2016-09-19 18:14 UTC)
_/_ | \_ ' _ | _ | \_ ' _ | Official http://julialang.org/ release
|_|/ | | | | | | | | x86_64-apple-darwin13.4.0
```

```
julia> -2 ** 4
ERROR: syntax: use "^" instead of "**"
```

```
julia> -2 ^ 4
-16
```

```
julia>
```

```
zsh % ghci
GHCi, version 8.0.2: http://www.haskell.org/ghc/ :? for help
Prelude> -2 ** 4
-16.0
Prelude>
```

```
zsh % ocaml
OCaml version 4.04.0
```

```
# -2 ** 4 ;;
Error: This expression has type int but an expression was expected of type
      float
# -2. ** 4. ;;
- : float = 16.
```

#

I also checked other languages like C#, Swift, Go, Rust and turned out they don't have this kind of operator.

So from my experiment I cannot say if it's Ruby or Cristal who's wrong. I think it's a design choice.

#5 - 01/24/2017 09:17 AM - yancya (Shinta Koyanagi)

Martin Dürst wrote:

- What do other languages (in particular languages more well known than Cristal) do?

I tried it by nodejs v7.4.0.

```
console.log(process.version);
//=> v7.4.0

console.log(-2 ** 4);
//console.log(-2 ** 4);
//           ^^
//SyntaxError: Unexpected token **

console.log((-2) ** 4); //=> 16
console.log(-(2 ** 4)); //=> -16
```

#6 - 01/24/2017 09:23 AM - repeatedly (Masahiro Nakagawa)

Martin Dürst wrote:

- What do other languages (in particular languages more well known than Cristal) do?

Here is clang result:

```
import std.stdio;

void main()
{
    writeln(-2 ^^ 4);    // -16
    writeln((-2) ^^ 4); // 16
}
```

#7 - 01/24/2017 10:05 AM - kaoru (Kaoru TAKAHASHI)

Martin Dürst wrote:

- What do other languages (in particular languages more well known than Cristal) do?

here is METAFONT result.

```
This is METAFONT, Version 2.7182818 (TeX Live 2016/Debian) (preloaded base=mf)
**\relax

*show -2**4;
>> 16
*show (-2)**4;
>> 16
```

#8 - 01/24/2017 01:15 PM - znz (Kazuhiro NISHIYAMA)

https://www.google.co.jp/search?q=-2**4

```
-(2 ** 4) =
-16
```

#9 - 01/24/2017 01:32 PM - ka (k a)

Martin Dürst wrote:

- What do other languages (in particular languages more well known than Cristal) do?

Lua result:

```
$ lua
Lua 5.3.3 Copyright (C) 1994-2016 Lua.org, PUC-Rio
> -2 ^ 4
-16.0
> - 2 ^ 4
-16.0
> (-2) ^ 4
16.0
```

#10 - 01/24/2017 01:43 PM - yasu (Yasuyuki Hirata)

FORTRAN

```
PROGRAM POW
WRITE (*,*) -2**4
STOP
END
```

=> -16

#11 - 01/25/2017 12:54 AM - ytomino (yuta tomino)

Ada

```
with Ada.Integer_Text_IO;
procedure pow is
begin
Ada.Integer_Text_IO.Put (-2 ** 4);
end;
```

-16

#12 - 01/25/2017 01:10 AM - ytomino (yuta tomino)

By the way, about "/" and "%" operators, in Ruby:

```
irb(main):001:0> -10%3 # (-10) % 3 == 2
=> 2
irb(main):002:0> 0-10%3 # 0 - (10 % 3) == -1
=> -1
```

in Ada:

```
with Ada.Integer_Text_IO;
procedure modu is
begin
Ada.Integer_Text_IO.Put (-10 mod 3); -- -(10 % 3) == -1
Ada.Integer_Text_IO.Put (0 - 10 mod 3); -- 0 - (10 % 3) == -1
Ada.Integer_Text_IO.Put ((-10) mod 3); -- (-10) % 3 == 2
end;
```

-1 -1 2

Which is more intuitive?

#13 - 01/25/2017 01:10 AM - metanest (Makoto Kishimoto)

```
$ nawk --version
awk version 20121220 (FreeBSD)
$ nawk 'END { printf("%d\n", -2 ** 4) }' < /dev/null
-16
```

#14 - 02/21/2017 07:11 PM - stomar (Marcus Stollsteimer)

The comparison with other programming languages is kind of interesting, but the main argument IMO for the implemented behavior hasn't been mentioned:

We are talking about mathematical operations here, and in **mathematics**, the expression $-a^b$ is equivalent to $-(a^b)$, because the exponentiation has a higher precedence. So Ruby's behavior *is* more "natural and intuitive" IMO (while I would tend to consider Crystal's behavior wrong, or confusing at best).

#15 - 03/13/2020 02:19 AM - shyouhei (Shyouhei Urabe)

- Has duplicate Bug #16677: Negative integer powered (**) to a float number results in a complex added

#16 - 03/13/2020 04:34 AM - sawa (Tsuyoshi Sawada)

wolframalpha.com

$-2^{**4} =$
 -16

$-2^4 =$
 -16

#17 - 09/24/2021 01:01 AM - shyouhei (Shyouhei Urabe)

- Has duplicate Bug #18188: -1^{**0} is 1 not -1 added