NAME

bigrat - Transparent BigNumber/BigRational support for Perl

SYNOPSIS

```
use bigrat;


$x = 2 + 4.5,"\n";   # BigFloat 6.5
print 1/3 + 1/4,"\n";   # produces 7/12
```

DESCRIPTION

All operators (inlcuding basic math operations) are overloaded. Integer and floating-point constants are created as proper BigInts or BigFloats, respectively.

Other than *bignum*, this module upgrades to Math::BigRat, meaning that instead of 2.5 you will get 2+1/2 as output.

Modules Used

`bigrat` is just a thin wrapper around various modules of the Math::BigInt family. Think of it as the head of the family, who runs the shop, and orders the others to do the work.

The following modules are currently used by bignum:

```
Math::BigInt::Lite      (for speed, and only if it is loadable)
Math::BigInt
Math::BigFloat
Math::BigRat
```

Math Library

Math with the numbers is done (by default) by a module called Math::BigInt::Calc. This is equivalent to saying:

```
use bigrat lib => 'Calc';
```

You can change this by using:

```
use bigrat lib => 'BitVect';
```

The following would first try to find Math::BigInt::Foo, then Math::BigInt::Bar, and when this also fails, revert to Math::BigInt::Calc:

```
use bigrat lib => 'Foo,Math::BigInt::Bar';
```

Please see respective module documentation for further details.

Sign

The sign is either '+', '-', 'NaN', '+inf' or '-inf'.

A sign of 'NaN' is used to represent the result when input arguments are not numbers or as a result of 0/0. '+inf' and '-inf' represent plus respectively minus infinity. You will get '+inf' when dividing a positive number by 0, and '-inf' when dividing any negative number by 0.

Methods

Since all numbers are not objects, you can use all functions that are part of the BigInt or BigFloat API. It is wise to use only the bxxx() notation, and not the fxxx() notation, though. This makes you independed on the fact that the underlying object might morph into a different class than BigFloat.

Cavaet

But a warning is in order. When using the following to make a copy of a number, only a shallow copy will be made.

```
$x = 9; $y = $x;
$x = $y = 7;
```

If you want to make a real copy, use the following:

```
$y = $x->copy();
```

Using the copy or the original with overloaded math is okay, e.g. the following work:

```
$x = 9; $y = $x;
print $x + 1, " ", $y,"\n";        # prints 10 9
```

but calling any method that modifies the number directly will result in **both** the original and the copy beeing destroyed:

```
$x = 9; $y = $x;
print $x->badd(1), " ", $y,"\n";        # prints 10 10


$x = 9; $y = $x;
print $x->binc(1), " ", $y,"\n";        # prints 10 10


$x = 9; $y = $x;
print $x->bmul(2), " ", $y,"\n";        # prints 18 18
```

Using methods that do not modify, but testthe contents works:

```
$x = 9; $y = $x;
$z = 9 if $x->is_zero();                # works fine
```

See the documentation about the copy constructor and = in overload, as well as the documentation in BigInt for further details.

Options

bignum recognizes some options that can be passed while loading it via use. The options can (currently) be either a single letter form, or the long form. The following options exist:

a or accuracy

This sets the accuracy for all math operations. The argument must be greater than or equal to zero. See Math::BigInt's bround() function for details.

```
perl -Mbigrat=a,50 -le 'print sqrt(20)'
```

p or precision

This sets the precision for all math operations. The argument can be any integer. Negative values mean a fixed number of digits after the dot, while a positive value rounds to this digit left from the dot. 0 or 1 mean round to integer. See Math::BigInt's bfround() function for details.

```
perl -Mbigrat=p,-50 -le 'print sqrt(20)'
```

t or trace

This enables a trace mode and is primarily for debugging bignum or Math::BigInt/Math::BigFloat.

l or lib

Load a different math lib, see *MATH LIBRARY*.

```
perl -Mbigrat=l,GMP -e 'print 2 ** 512'
```

Currently there is no way to specify more than one library on the command line. This will be hopefully fixed soon ;)

v or version

This prints out the name and version of all modules used and then exits.

```
perl -Mbigrat=v
```

EXAMPLES

```
perl -Mbigrat -le 'print sqrt(33)'
perl -Mbigrat -le 'print 2*255'
perl -Mbigrat -le 'print 4.5+2*255'
perl -Mbigrat -le 'print 3/7 + 5/7 + 8/3'
perl -Mbigrat -le 'print 12->is_odd()';
```

LICENSE

This program is free software; you may redistribute it and/or modify it under the same terms as Perl itself.

SEE ALSO

Especially *bignum*.

Math::BigFloat, *Math::BigInt*, *Math::BigRat* and *Math::Big* as well as *Math::BigInt::BitVect*, *Math::BigInt::Pari* and *Math::BigInt::GMP*.

AUTHORS

(C) by Tels *http://bloodgate.com/* in early 2002 - 2005.