NAME

Types::Common::Numeric − drop−in replacement for MooseX::Types::Common::Numeric

STATUS

This module is covered by the Type-Tiny stability policy.

DESCRIPTION

A drop-in replacement for MooseX::Types::Common::Numeric.

Types

The following types are similar to those described in MooseX::Types::Common::Numeric.

```
PositiveNum
PositiveOrZeroNum
PositiveInt
PositiveOrZeroInt
NegativeNum
NegativeOrZeroNum
NegativeInt
NegativeOrZeroInt
SingleDigit
```
    SingleDigit interestingly accepts the numbers −9 to −1; not just 0 to 9.

This module also defines an extra pair of type constraints not found in MooseX::Types::Common::Numeric.

```
IntRange[`min, `max]
```
    Type constraint for an integer between min and max. For example:

```
    IntRange[1, 10]
```

    The maximum can be omitted.

```
    IntRange[10]    # at least 10
```

    The minimum and maximum are inclusive.

```
NumRange[`min, `max]
```
    Type constraint for a number between min and max. For example:

```
    NumRange[0.1, 10.0]
```

    As with IntRange, the maximum can be omitted, and the minimum and maximum are inclusive.

    Exclusive ranges can be useful for non-integer values, so additional parameters can be given to make
    the minimum and maximum exclusive.

```
    NumRange[0.1, 10.0, 0, 0]  # both inclusive
    NumRange[0.1, 10.0, 0, 1]  # exclusive maximum, so 10.0 is invalid
    NumRange[0.1, 10.0, 1, 0]  # exclusive minimum, so 0.1 is invalid
    NumRange[0.1, 10.0, 1, 1]  # both exclusive
```

    Making one of the limits exclusive means that a < or > operator will be used instead of the usual <= or
    >= operators.

BUGS

Please report any bugs to <http://rt.cpan.org/Dist/Display.html?Queue=Type−Tiny>.

SEE ALSO

Types::Standard, Types::Common::String.

MooseX::Types::Common, MooseX::Types::Common::Numeric, MooseX::Types::Common::String.

AUTHOR

Toby Inkster <tobyink@cpan.org>.

COPYRIGHT AND LICENCE

DISCLAIMER OF WARRANTIES