Table of Contents

MoonBit Float Package Documentation

This package provides operations on 32-bit floating-point numbers (Float). It includes basic arithmetic, trigonometric functions, exponential and logarithmic functions, as well as utility functions for rounding and conversion.

Special Values

The package defines several special floating-point values:

```

test "special float values" {

  inspect(@float.infinity, content="Infinity")
  inspect(@float.neg_infinity, content="-Infinity")


  inspect(@float.not_a_number, content="NaN")


  inspect(@float.max_value, content="3.4028234663852886e+38")
  inspect(@float.min_value, content="-3.4028234663852886e+38")
  inspect(@float.min_positive, content="1.1754943508222875e-38")
}


test "checking special values" {

  inspect(@float.infinity.is_inf(), content="true")
  inspect(@float.neg_infinity.is_neg_inf(), content="true")
  inspect(@float.infinity.is_pos_inf(), content="true")
  inspect(@float.not_a_number.is_nan(), content="true")
}
```

Rounding Functions

The package provides various ways to round floating-point numbers:

```

test "rounding functions" {

  inspect(@float.ceil(3.2), content="4")
  inspect(@float.ceil(-3.2), content="-3")


  inspect(@float.floor(3.2), content="3")
  inspect(@float.floor(-3.2), content="-4")


  inspect(@float.round(3.7), content="4")
  inspect(@float.round(3.2), content="3")


  inspect(@float.trunc(3.7), content="3")
  inspect(@float.trunc(-3.7), content="-3")
}
```

Utility Functions

Other useful operations on floats:

```

test "utility functions" {

  inspect(@float.abs(-3.14), content="3.140000104904175")


  inspect(3.14.to_int(), content="3")


  inspect(@float.default(), content="0")
}
```

Byte Representation

Functions to convert floats to their byte representation:

```

test "byte representation" {
  let x : Float = 3.14

  let be_bytes = x.to_be_bytes()

  let le_bytes = x.to_le_bytes()
  inspect(be_bytes.length(), content="4")
  inspect(le_bytes.length(), content="4")
}
```

Method Style

All functions can also be called in method style:

```

test "method style calls" {
  let x : Float = 3.14
  inspect(x.floor(), content="3")
  inspect(x.ceil(), content="4")
  inspect(x.round(), content="3")
  let y : Float = 2.0
  inspect(y.pow(3.0), content="8")
}
```