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:

```
1
2   test "special float values" {
3
4     inspect(@float infinity, content="Infinity")
5     inspect(@float neg_infinity, content="-Infinity")
6
7
8     inspect(@float not_a_number, content="NaN")
9
10
11    inspect(@float max_value, content="3.4028234663852886e+38")
12    inspect(@float min_value, content="-3.4028234663852886e+38")
13    inspect(@float min_positive, content="1.1754943508222875e-38")
14  }
15
16
17  test "checking special values" {
18
19    inspect(@float infinity is_inf(), content="true")
20    inspect(@float neg_infinity is_neg_inf(), content="true")
21    inspect(@float infinity is_pos_inf(), content="true")
22    inspect(@float not_a_number is_nan(), content="true")
23  }
```

Rounding Functions

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

```
1
2   test "rounding functions" {
3
4     inspect(@float ceil(3.2), content="4")
5     inspect(@float ceil(-3.2), content="-3")
6
7
8     inspect(@float floor(3.2), content="3")
9     inspect(@float floor(-3.2), content="-4")
10
11
12    inspect(@float round(3.7), content="4")
13    inspect(@float round(3.2), content="3")
14
15
16    inspect(@float trunc(3.7), content="3")
17    inspect(@float trunc(-3.7), content="-3")
18  }
```

Utility Functions

Other useful operations on floats:

```
1
2   test "utility functions" {
3
4     inspect(@float abs(-3.14), content="3.140000104904175")
5
6
7     inspect(3.14 to_int(), content="3")
8
9
10    inspect(@float default(), content="0")
11  }
```

Byte Representation

Functions to convert floats to their byte representation:

```
1
2   test "byte representation" {
3     let x : Float = 3.14
4
5     let be_bytes = x to_be_bytes()
6
7     let le_bytes = x to_le_bytes()
8     inspect(be_bytes length(), content="4")
9     inspect(le_bytes length(), content="4")
10  }
```

Method Style

All functions can also be called in method style:

```
1
2   test "method style calls" {
3     let x : Float = 3.14
4     inspect(x floor(), content="3")
5     inspect(x ceil(), content="4")
6     inspect(x round(), content="3")
7     let y : Float = 2.0
8     inspect(y pow(3.0), content="8")
9   }
```