Table of Contents

int64

This package provides operations for working with 64-bit signed integers (Int64) in MoonBit.

Basic Operations

Int64 values can be created from regular 32-bit integers using from_int. The package also provides constants for the maximum and minimum values representable by Int64.

```
1
2    test "basic operations" {
3      let i : Int64 = -12345L
4
5      inspect(@int64 from_int(-12345) == i, content="true")
6
7
8      inspect(@int64 max_value, content="9223372036854775807")
9      inspect(@int64 min_value, content="-9223372036854775808")
10
11
12      inspect(@int64 abs(i), content="12345")
13    }
```

Binary Representation

The package provides functions to convert Int64 values to their binary representation in both big-endian and little-endian byte order:

```
1
2   test "binary conversion" {
3     let x = 258L
4     let be_bytes = x to_be_bytes()
5     let le_bytes = x to_le_bytes()
6
7
8     inspect(
9       be_bytes to_string(),
10      content=(
11        #|b"\x00\x00\x00\x00\x00\x00\x01\x02"
12      ),
13    )
14    inspect(
15      le_bytes to_string(),
16      content=(
17        #|b"\x02\x01\x00\x00\x00\x00\x00\x00"
18      ),
19    )
20
21
22    let len = be_bytes length()
23    inspect(len, content="8")
24  }
```

Method-Style Usage

All operations are also available as methods on Int64 values:

```
1
2   test "method style" {
3     let x = -42L
4
5
6     inspect(x abs(), content="42")
7
8
9     inspect(
10      x to_be_bytes(),
11      content=(
12        #|b"\xff\xff\xff\xff\xff\xff\xff\xd6"
13      ),
14    )
15  }
```

Note that Int64 implements the Hash trait, allowing it to be used as keys in hash maps and members of hash sets.