Table of Contents

uint

This package provides functionalities for handling 32-bit unsigned integers in M
oonBit. To this end, it includes methods for converting between UInt and other n
umber formats, as well as utilities for byte representation.

Basic Properties

uint provides constants for UInt's value range and default value:

```
1
2   test "uint basics" {
3
4     inspect(@uint.default(), content="0")
5
6
7     inspect(@uint.max_value, content="4294967295")
8     inspect(@uint.min_value, content="0")
9   }
```

Byte Representation

UInt can be converted to bytes in both big-endian and little-endian formats:

```
1
2   test "uint byte conversion" {
3     let num = 258U
4
5
6     let be_bytes = num.to_be_bytes()
7     inspect(
8       be_bytes,
9       content=(
10        #|b"\x00\x00\x01\x02"
11      ),
12    )
13
14
15    let le_bytes = num.to_le_bytes()
16    inspect(
17      le_bytes,
18      content=(
19        #|b"\x02\x01\x00\x00"
20      ),
21    )
22  }
```

Converting to Other Number Types

UInt can be converted to Int64 when you need to work with signed 64-bit integers
:

```
1
2   test "uint type conversion" {
3     let num = 42U
4     inspect(num.to_int64(), content="42")
5     let large_num = 4294967295U
6     inspect(large_num.to_int64(), content="4294967295")
7   }
```

These conversion functions are also available as methods:

```
1
2   test "uint methods" {
3     let num = 1000U
4
5
6     inspect(num.to_int64(), content="1000")
7     inspect(
8       num.to_be_bytes(),
9       content=(
10        #|b"\x00\x00\x03\xe8"
11      ),
12    )
13    inspect(
14      num.to_le_bytes(),
15      content=(
16        #|b"\xe8\x03\x00\x00"
17      ),
18    )
19  }
```