Table of Contents

uint16

The moonbitlang/core/uint16 package provides functionality for working with 16-bit unsigned integers. This package includes constants, operators, and conversions for UInt16 values.

Constants

The package defines the minimum and maximum values for UInt16:

```

test "UInt16 constants" {

  inspect(@uint16.min_value, content="0")


  inspect(@uint16.max_value, content="65535")
}
```

Arithmetic Operations

UInt16 supports standard arithmetic operations:

```

test "UInt16 arithmetic" {
  let a : UInt16 = 100
  let b : UInt16 = 50


  inspect(a + b, content="150")


  inspect(a - b, content="50")


  inspect(a * b, content="5000")


  inspect(a / b, content="2")


  inspect(@uint16.max_value + 1, content="0")
  inspect(@uint16.min_value - 1, content="65535")
}
```

Bitwise Operations

UInt16 supports various bitwise operations:

```
test "UInt16 bitwise operations" {
  let a : UInt16 = 0b1010
  let b : UInt16 = 0b1100


  inspect(a & b, content="8")


  inspect(a | b, content="14")


  inspect(a ^ b, content="6")


  inspect(a << 1, content="20")
  inspect(a << 2, content="40")


  inspect(a >> 1, content="5")
  inspect(b >> 2, content="3")
}
```

Comparison and Equality

UInt16 supports comparison and equality operations:

```
test "UInt16 comparison and equality" {
  let a : UInt16 = 100
  let b : UInt16 = 50
  let c : UInt16 = 100


  inspect(a == c, content="true")
  inspect(a != b, content="true")


  inspect(a > b, content="true")
  inspect(b < a, content="true")
  inspect(a >= c, content="true")
  inspect(c <= a, content="true")
}
```

Default Value and Hashing

UInt16 implements the Default trait:

```

test "UInt16 default value" {

  let a : UInt16 = 0
  inspect(a, content="0")


  let value : UInt16 = 42
  inspect(value.hash(), content="42")
}
```

Type Conversions

UInt16 works with various conversions to and from other types:

```

test "UInt16 conversions" {

  inspect((42).to_uint16(), content="42")


  let value : UInt16 = 100
  inspect(value.to_int(), content="100")


  inspect((-1).to_uint16(), content="65535")
  inspect((65536).to_uint16(), content="0")
  inspect((65537).to_uint16(), content="1")


  inspect(b'A'.to_uint16(), content="65")
  inspect(b'\xFF'.to_uint16(), content="255")
}
```