

BitOps

Sisältää perus binäärioperaatioita, ja muistipuskureihin liittyviä funktioita. Jotkin näistä funktioista ovat päällekkäisiä luaJIT:n **bit** -kirjaston kanssa, eikä niiden välillä ole suurta toiminnallista eroa.

Huomaa, että monista operaatioista on olemassa eri versioista (esim. not16 ja not32) joiden erona on, että operaatioiden aikana lukuja käsitellään funktion nimessä mainitulla tarkkuudella – binääri-operaatioiden tapauksessa muunnos tehdään aina unsigned muotoon (ei etumerkkiä).

BitOps.not16 (v)

BitOps.not32 (v)

v Arvo jolle halutaan tehdä binäärinen NOT operaatio.

Suorittaa annetun luvun biteille loogisen NOT operaation, eli invertoi ne.

Esimerkki:

-- Tulos on 0xFF

local r = BitOps.not16 (0xFF00)

BitOps.and16 (v1, v2)

BitOps.and32 (v1, v2)

v1 Binäärisen JA -operaation syöteluku.

v2 Binäärisen JA -operaation syöteluku.

Suorittaa annettujen lukujen välillä AND operaation.

Esimerkki:

-- Tulos on 0xFF

local r = BitOps.and16 (0xFFFF, 0xFF)

BitOps.or16 (v1, v2)

BitOps.or32 (v1, v2)

v1 Binäärisen OR operaation syöte.

v2 Binäärisen OR operaation syöte.

Suorittaa annettujen lukujen välillä OR operaation.

Esimerkki:

```
-- Tulos on 0xFFFF
```

```
local r = BitOps.and16 (0xFF00, 0xFF)
```

BitOps.xor16 (v1, v2)

BitOps.xor32 (v1, v2)

v1 Binäärisen XOR operaation syöte.

v2 Binäärisen XOR operaation syöte.

Suorittaa annettujen lukujen välillä n.s. poissulkevan OR operaation (eng. exclusive or), jota usein merkitään XOR lyhenteellä.

Esimerkki:

```
-- Tulos on 0xFF
```

```
local r = BitOps.and16 (0xFFFF, 0xFF00)
```

BitOps.shr16 (v, n)

BitOps.shr32 (v, n)

BitOps.shl16 (v, n)

BitOps.shl32 (v, n)

v Siirto operaation kohdeluku.

n Siirrettävien bittien määrä.

Siirtää luvun v bittejä n kappaletta joko oikealle (shr) tai vasemmalle (shl). Vasemmalle siirto kasvattaa luvun arvoa, ja oikealle siirto pienentää sitä.

Esimerkki:

```
-- Siirtää bittejä puolikkaan tavun verran,
```

```
-- joka vastaa yhtä heksadesimaali numeroa
```

```
-- Tulos on 0xFF00
```

```
local r = BitOps.shl16 (0xFF0, 4)
```

BitOps.createBuffer (n)

n Luotavan puskurin koko tavuina.

Luo uuden raakadatapuskurin.

Esimerkki:

```
-- Luo puskurin jonka koko on 32 tavua.
```

```
local buf = BitOps.createBuffer (32)
```

BitOps.fillBuffer (b, v, i, n)

b Puskuri jolle operaatio tehdään (luotu createBuffer kutsulla)

v Arvo jota puskuuriin kirjoitetaan (huom! Arvo muutetaan BYTE tyyppiseksi, lukualue 0x0 .. 0xFF)

i Indeksistä josta täyttö aloitetaan (oltava ≥ 0)

n Kirjoitettavien tavujen määrä (puskurin indeksistä *i* lähtien).

Kutsu kirjoittaa BitOps.createBuffer -kutsulla luotuun muistipuskuriin tiettyä argumenttina annettua tavua, ja sitä voidaan käyttää esimerkiksi nollaamaan puskurin muistipaikat.

Huomaa että indeksointi seuraa lua käytäntöä, eli puskurin ensimmäinen tavu on indeksissä 1!

Esimerkki:

```
-- Luo puskurin ja nollaa sen kaikki tavut
```

```
local buf = BitOps.createBuffer (32)
```

```
BitOps.fillBuffer (buf, 0x0, 0, 32)
```

BitOps.deleteBuffer (b)

b Puskuri jolle operaatio tehdään (luotu createBuffer kutsulla)

Tuhoaa muistipuskurin ja vapauttaa sille varatun muistin.

Esimerkki:

```
-- Luo puskurin ja sitten tuhoaa sen
```

```
local buf = BitOps.createBuffer (2048)
if buf then
    BitOps.deleteBuffer (buf)
end
```

BitOps.getBufferLen (b)

b Puskuri jonka koko halutaan tietää

Palauttaa argumenttina annetun muistipuskurin pituuden – eli sille varatun muistin määrän tavuina.

Esimerkki:

```
-- Luo puskurin ja näyttä sen koon
```

```
local buf = BitOps.createBuffer (2048)
if buf then
    print ("Buffer is " .. BitOps.getBufferLen(buf) .. " bytes long")
end
```

BitOps.setBufferLen (b)

b Puskuri jonka pituus halutaan asettaa

Muuttaa jo luodun muistipuskuri kokoa. Operaatio tehdään varaamalla puskurille uusi muistialue ja kopiaimalla vanha sisältö uuteen – jos uusi puskur on pienempi kuin vanha, kopioidaan vain se osa joka uuteen puskuriiin mahtuu.

Esimerkki:

```
-- Luo puskurin ja muuttaa sen koon pienemmäksi  
local buf = BitOps.createBuffer (2048)  
BitOps.setBufferLen (buf, 32)
```

BitOps.getBytes (b, i)

BitOps.setByte (b, i, v)

BitOps.getWord (b, i)

BitOps.setWord (b, i, v)

BitOps.getDWord (b, i)

BitOps.setDWord (b, i, v)

BitOps.getString (b, i)

BitOps.setString (b, i, v)

BitOps.getFloat32 (b, i)

BitOps.setFloat32 (b, i, v)

b Puskuri jolle operaatio tehdään

i Luettavan tai arvon alkukohdan indeksi (tavuja)

v Kirjoitettava arvo

Näillä funktioilla voidaan kirjoittaa ja lukea muistipuskurista yksittäisiä tavuja, sanoja, kaksoisanoja, merkkijonoja tai IEEE koodattuja liukulukuja.

Huomaa että indeksointi seuraa lua käytäntöä, eli puskurin ensimmäinen tavu on indeksissä 1!

Esimerkki:

```
-- Luo puskurin  
local buf = BitOps.createBuffer (2048)  
BitOps.setString(buf, 1, "Handling buffer")  
local byte = BitOps.getBytes(buf, 1) -- Lukee merkin 'H' ascii koodin 0x48
```

BitOps.asBitArray (v, [n])

v Luku joka halutaan muuttaa

n Vastauksen sisältämä TAVU määrä (ei pakollinen)

Muuttaa annetun luvun lua taulukoksi, joka sisältää riveillä annetun luvun bitit.

Argumentilla *n* voidaan määrätä montako tavua vastauksessa on (1 tavu vastaa 8 bittiä, eli 8 riviä).

Esimerkki:

```
-- Luo puskurin ja muuttaa sen koon pienemmäksi
```

```
local arr = BitOps.asBitArray(0xF0, 1)
```

>> tulos:

```
arr = {1, 1, 1, 1, 0, 0, 0, 0}
```

BitOps.arrayAsDWord (t)

t Luvuksi muunnettava taulukko

Käänteinen operaatio BitOps.asBitArray () -kutsulle. Muuntaa annetun bitti taulukon takaisin luvuksi.

Esimerkki:

```
-- v saa arvon 0xF0
```

```
local v = BitOps.arrayAsDWord ({1, 1, 1, 1, 0, 0, 0, 0})
```

BitOps.fillArray (v, n, [i])

v Arvo jota taulukon soluihin kirjoitetaan

n Kuinka monta kertaa arvo kirjoitetaan taulukkoon

i Indeksi josta kirjoittaminen aloitetaan (ei pakollinen)

Luo uuden lua -taulukon ja täyttää sen annetulla luvulla.

Esimerkki:

```
-- luo taulukon jossa 512 riviä arvoina 1
```

```
local t = BitOps.fillArray( 1, 512 )
```

BitOps.convertTo (v, t)

v Arvo joka muunnetaan

t Muoto merkkijonona, johon luku halutaan muuttaa "int8", "uint8", "int16", "uint16", "int32", "uint32", "int64", "uint64", "float32".

Muuttaa luvun annettuun datatyyppiin. Tätä muunnosta tarvitaan melko harvoin, mutta joskus on välttämätöntä tulkita luku esimerkiksi uint16 tyyppissä.

Esimerkki:

```
local v = 0xFFFF
```

```
local int16 = BitOps.convertTo (v, "int16")  -- Tulos -32767
local uint16 = BitOps.convertTo (v, "uint16")  -- Tulos 65535
local overFlow = BitOps.convertTo ( (v+1), "uint16")  -- Tulo nolla
```

BitOps.binaryDWordAsFloat (dw)

BitOps.binaryFloatAsDWord (f)

dw Kaksoissana joka halutaan muuttaa float32 koodatuksi.

f Float32 luku josta halutaan saada raaka binääriesitys.

Näillä kutsuilla on mahdollista muuttaa esimerkiksi tiedostosta tai sarjaliikenneportista luettu binääriesitys takaisin liukuluvuksi, ja toisinpäin.

Esimerkki:

-- puskuriin luetussa datassa on liukuluku,

-- palautetaan se binääri esityksestä käytettävään muotoon

b = BitOps.getDWord (buf, 5)

f = BitOps.binaryDWordAsFloat (b)

Revision #1

Created 25 May 2022 10:36:09 by Severi Hiltunen

Updated 10 May 2023 10:25:34 by Severi Hiltunen