

Introduktion til Lua

Thomas B. Rasmussen

FLUG - Fyns Linux User Group

25. Oktober, 2012



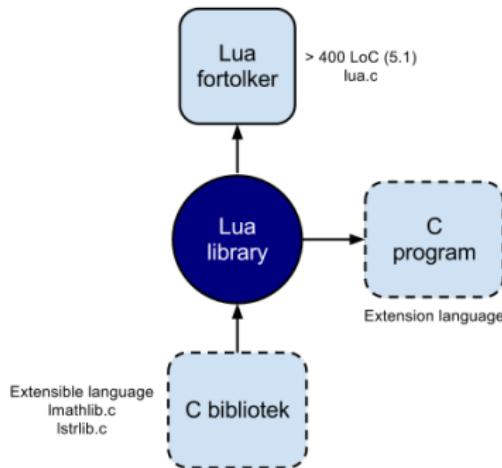
Hvad er Lua?

- Lua er et kraftfuldt, letvægts, embeddable scripting sprog
- Udgivet under MIT licens
- Lua er udviklet i standart C (for portabilitet)
- Populært som embedded sprog i fx. C og C++ programmer
- Udviklet i Brasilien
- Lua er portugisisk for måne

Lua releases

- 5.2 – 16. december 2011
- 5.1 – 21. februar 2006
- 5.0 – 11. april 2003
- 4.0 – 6. november 2000
- 3.0 – 1. juli 1997
- 2.1 – 7. februar 1995
- 1.0 – 28. juli 1993

Lua oversigt



Lua 5.2 source code <http://www.lua.org/source/5.2/>

Hvem og hvor benyttes Lua?

- Adobe Lightroom
- World of Warcraft
- Lighttpd
- Wikipedia – Lua som template sprog
- Redis 2.6.0+
- Xavante – webserver skrevet i Lua
- PL/Lua – loadable language for PostgreSQL
- Samsung – Samsung Game Framework (Lua og SDL)
- Ginga – brasiliansk Digital TV Middleware System



- Installer igennem de fleste pakkesystemer
`sudo apt-get install lua`
- Hent *.tar.gz fra <http://www.lua.org/download.html>
- Lua for Windows batteries included
<http://code.google.com/p/luaforwindows/>



- Lua script

```
$lua hello-world.lua  
$hello world
```

- Lua interaktiv

```
$lua  
Lua 5.1.4 Copyright (C) 1994-2008 Lua.org,  
PUC-Rio  
> print "Hello world"  
Hello world  
>^C
```



Lua keywords

and	break	do	else
elseif	end	false	for
function	goto ¹	if	in
local	nil	not	or
repeat	return	then	true
until	while		

<http://www.lua.org/manual/5.2/manual.html#3.1>

¹Først med i Lua fra 5.2

- Nil – svarende til NULL i mange andre sprog
- Boolean – true, false
- Number – heltal, kommatal...
- String
- Table
- Function
- Userdata & Threads

Typer

Table

- Tabeller implementere associative arrays
- Det er værd at bemærke at Lua er 1 indekseret...
- Tabeller er mere end bare arrays...

Funktioner i Lua er first-class values, dvs

- de kan gemmes i variabler
- sendes med som argumenter til funktioner
- og returneres fra funktioner



Expressions

Aritmetiske operatorer

- + – addition
- - – substraktion
- * – multiplikation
- / – division
- ^ – eksponential
- % – modulo²
- Lua har ikke operatorerne ++, +=, *=, -= etc.

²Først fra Lua 5.1

Expressions

Relationelle operatorer

- `<-` mindre end
- `>-` større end
- `<=` – mindre end lig med
- `>=` – større end lig med
- `==` – lig med
- `~=` – forskellig fra

Expression

Logiske operatorer

- and
- or
- not
- Lua har ikke bitwise operatorer som & og |

Statements

Assignment

- = – assignment

Statements

Lokale variabler

- Variabler er som standart globale
- Lokale variabler skal erklæres som local³
- Udover at man kan spare sig selv for en del fejl kan der være en performance gevindst ved at erklære variabler lokale

³Som Javascripts var

Statement

Kontrol strukturer

- if then else
- while
- repeat
- for
- break og return

Functions

- Lua funktioner understøtter multiple return values
- Variabelt antal argumenter ...
- Named arguments
- Closures

Minder om threads

- egen stak
- egne lokale variabler
- egen instruktions pointer
- deler globale variabler

Og så alligevel ikke

- er ikke concurrent, der kører aldrig mere end en af gangen
- non-preemptive multithreading. Kan ikke stoppes udefra når de kører, suspenderer selv
- forsimpler da synkronisering derved ikke er et problem



Coroutines

Tilstande

- suspended (pause, sleep)
- running
- dead
- normal (coroutine resumer anden coroutine)



Coroutines

Funktioner

Modulet coroutine indeholder de følgende funktioner

- `create(...)`
- `status(...)`
- `resume(...)`
- `yield(...)`



Metatables & Metamethods

Metatables og metametoder kan benyttes til at

- udvide Lua's indbyggede funktionalitet og semantik
- ændre eksisterende metoder og operatorer
- setmetatable()
- getmetatable()



Metatables & Metamethods

Aritmetiske metametoder

- __add – +
- __sub – -
- __mul – *
- __div – /
- __unm – -
- __mod – %
- __pow – ^
- __concat – ..

Metatables & Metamethods

Relationelle metametoder

- __lt – <
- __le – <=
- __eq – =

Der findes ikke tilsvarende metametoder for ~=, > og >= da Lua oversætter vha. af de ovenstående og not operatoren.



Metatables & Metamethods

Library metametoder

- __tostring – tostring()
- __metatable – setmetatable()
- __len – #⁴

⁴Først fra Lua 5.2

Metatables & Metamethods

Table-Access metametoder

- `__index` – når et ikke eksisterende felt læses fra tabel rammes denne metode
- `__newindex` – ved skrivning til ikke eksisterende felt rammes denne metode
- `rawget(tbl, k)` – tilgår felt uden om `__index`
- `rawset(tbl, k, v)` – opdaterer felt uden om `__newindex`



- `_G` – er en tabel der indeholder globale værdier

Der er ændret en del i 5.2. Bl.a. er funktionerne `setfenv()` og `getfenv()` er deprecated

Moduler & Packages

Moduler indlæses med require

```
require "module"  
module.foo()
```

Man kan omdøbe moduler og deres funktioner så man slipper for at skrive så meget

```
local m = require "module"  
m.foo()  
local f = m.foo  
f()
```

require kigger efter module.lua i \$LUA_PATH



Moduler & Packages

Kompilerede moduler

I Lua er det muligt at prækompilere sin kode til bytecode

```
$luac -o module.out module.lua
```

- hurtigere load af programmet (samme afviklings tid)
- mulighed for binær distribution
- fjerne bytecode compileren (mindre memory footprint)



Det er muligt at få listet bytecode i det binære modul

```
$luac -l module.out
```

Derudover kan man benytte luac til at checke for syntax fejl

```
$luac -p module.lua
```

- `$LUA_PATH`⁵ – system variabel der fortæller Lua hvor den skal lede efter moduler
- `package.path` – Lua variabel der inderholder `$LUA_PATH`
- `$LUA_CPATH` – fortæller Lua hvor den skal lede efter C moduler
- `package.cpath` – Lua variabel der inderholder `$LUA_CPATH`

```
$cat ~/.bashrc | grep 'LUA_PATH'  
export LUA_PATH='~/lua/LIBS/?..lua;  
/usr/local/share/lua/5.1/?..lua;;'
```

⁵Samme som Java `$CLASSPATH`

Selv om Lua ikke har class begrebet er det muligt at implementere de OOP begreber vi kender fra andre sprog

- Objekter
- Konstruktører
- Nedarvning
- Polymorphi
- Private/Public

Til OOP udnytter vi Lua's tabel datatype og at funktioner er first-class values

Weak tables

- Lua har automatisk memory management og garbage collection (GC)
- Selv den smarteste GC kan ikke altid vide hvad der er garbage, derfor weak tables
- Det er kun objekter der GC
- collectgarbage() kører en GC

Weak tables

Tabels kan have forskellige modes

Sættes ved at ændre værdien for feltet `__mode` for tabellens metatable

- Strong – default, betyder der ikke GC hvis tabellen er accessible
- Weak key – weak reference på keys
- Weak value – weak reference på values
- Weak – weak reference af både key og values

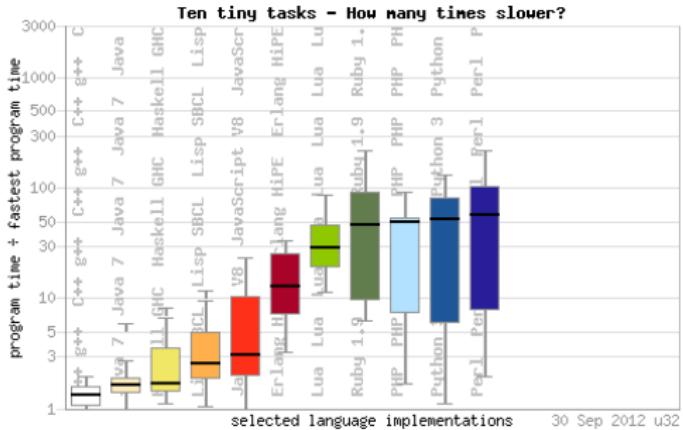
```
mt.__mode = "k"      - "k", "v" eller "kv"
```

- basic – Lua basis modul: assert(), print(), tostring() etc.
- coroutine – operationer ifbm. coroutiner
- package – funktionalitet til at loade packages/biblioter
- math – forskellige matematiske funktioner
- table – funktioner til operationer på tabeller
- string – streng funktioner
- io – I/O funktioner fx læse eller skrive til/fra filer
- os – Operativsystem funktioner
- debug – til debugging og profiling
- bit32⁶ – indeholder funktioner til bitwise manipulation

⁶Fra Lua 5.2



Hastighed

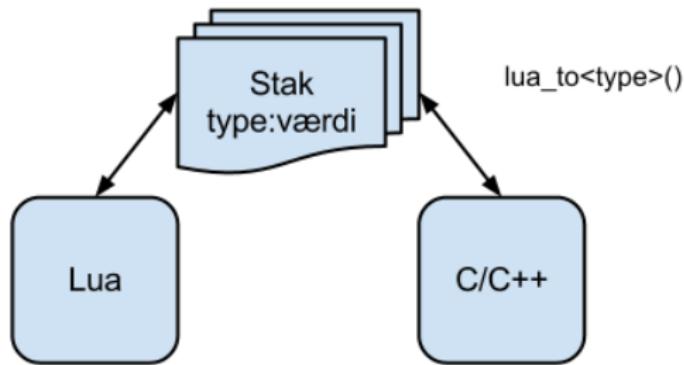


Kilde: <http://shootout.alioth.debian.org/u32/which-programming-languages-are-fastest.php>



- Det er muligt at embedde en Lua fortolker i C/C++ programmer
- Det er muligt at kalde Lua funktioner fra C/C++
- Det er muligt at kalde C/C++ funktioner fra Lua
- Udfordringer:
 - Dynamic & static typed
 - Automatic & manual memory management
- Løsning: Lua & C kommunikerer via en stak

Lua & C integration



All kommunikation går igennem stak'en: `lua_State*`

- Det er også muligt at integrere Lua med Java
 - Luajava – <http://www.keplerproject.org/luajava/>
 - JNLua – Java Native Lua
<http://code.google.com/p/jnlua/>
- Jeg har ingen praktisk erfaring med at kombinere Java og Lua...

LuaJit - just-in-time Compiler

- Installer via pakkesystem eller hent fra luajit.org
- State-of-the-art JIT compiler skrevet i assembler
- OS: Linux, Android, BSD, OSX, IOS, Windows
- CPU: X86, X64, ARM, PPC, MIPS
- Compilering er hurtig og giver markante hastighedsforbedringer

LuaJit - just-in-time Compiler

Jit compilet kode

```
$time lua hello-world.lua
```

```
real    0m0.002s  
user    0m0.000s  
sys     0m0.000s
```

```
$time luajit hello-world.lua
```

```
real    0m0.002s  
user    0m0.000s  
sys     0m0.000s
```



LuaJit - just-in-time Compiler

Jit compilet kode

```
$time lua primes.lua
78498
real    0m1.063s
user    0m1.060s
sys     0m0.000s
```

```
$time luajit primes.lua
78498
real    0m0.120s
user    0m0.116s
sys     0m0.000s
```

LuaRocks - pakkesystem

- Hent LuaRocks⁷ via pakkesystem
sudo apt-get install luarocks
- Eller hent tarball <http://luarocks.org/en/Download>

```
$ ./configure  
$make  
$sudo make install
```

⁷Ruby Gems, Perl CPAN, Haskell Cabal...



LuaRocks - pakkesystem

Indstaller pakker

- Find den relevante pakke

<http://luarocks.org/repositories/rocks/>

```
$luarocks help
```

```
$luarocks help install
```

```
$luarocks install luasocket
```

LuaRocks - pakkesystem

Eksempler på pakker

- LuaSocket – socket library
- LuaCurl – curl library til Lua
- LuaRedis – Redis library
- hige – mustache template modul
- abelhas – Particle swarm optimization (PSO)
- lpdf – A library for generating PDF documents (PDFlib)
- LuaPSQL – PostgreSQL client bindings for Lua
- lplib – Geometriske funktioner

Ressourcer

-  Programming in Lua
-  Lua Programming Gems
-  Lua Reference Manual



- Officiel hjemmeside <http://www.lua.org>
- Programming in Lua <http://www.lua.org/pil/>
- Side om Lua <http://lua-users.org/>
- Pakkesystem <http://luarocks.org>
- JiT compiler <http://luajit.org>
- Lua projekter <http://luaforge.net/>
- Ion windowmanager <http://tuomov.iki.fi/software/>
- Lua projekt side <http://www.keplerproject.org>
- Wikipedia [http://en.wikipedia.org/wiki/Lua_\(programming_language\)](http://en.wikipedia.org/wiki/Lua_(programming_language))
- Reimplementationer af Lua
<http://lua-users.org/wiki/LuaImplementations>



Links

(fortsat)

- Lua reference http://pgl.yoyo.org/luai/i/_
- Lua C reference <http://pgl.yoyo.org/luai/i/3.7+Functions+and+Types>
- World of Warcraft <http://wowprogramming.com>
- World of Warcraft <http://www.wowwiki.com/Lua>
- Lighttpd <http://redmine.lighttpd.net/projects/lighttpd/wiki/absolution>
- Redis <http://redis.io/commands/eval>
- Xavante webserver
<http://keplerproject.github.com/xavante/>
- PostgreSQL <http://pllua.projects.postgresql.org/>
- Ginga TV Middleware <http://www.ginga.org.br/>

