Using LuaJIT in mid-load web-projects

Alexander Gladysh
CTO, LogicEditor

Российские
интернет-
технологии

Российские интернет-технологии

On Lua and LuaJIT

Lua:

- powerful,
- fast,
- lightweight,
- extensible,
- embeddable

scripting programming language.

Language

On Lua and LuaJIT

- Origins.
- Popularity growth.
- Where is Lua used?
- LuaRocks.

Popular Dialects

On Lua and LuaJIT

- Lua 5.1 vs. Lua 5.2,
- LuaJIT 2.0,
- Metalua.

LuaJIT 2.0

On Lua and LuaJIT

- JIT, FFI, performance.
- Limitations on x86_64.
- LuaJIT vs. Lua 5.2.

Why Lua?

On Lua and LuaJIT

Historically: We're coming from computer games industry, where Lua "rules the world".

Pragmatically:

- Works fast!
- Pleasant to code!
- Easy to learn!

Where to get programmers?

On Lua and LuaJIT

Reeducation.

Main problems learning Lua

On Lua and LuaJIT

- Mindless tinkering with the language.
  - NIH-syndrome. Lua is to easy to tinker with.
  - Diverging from the mainstream. Costs and benefits.
- Language idiosyncrasies:
  - Global variables by default.
  - Arrays are indexed from 1.
  - Size of array with nil element is not defined.
  - Everything that is not nil or false — true (0 too).

Most important!

On Lua and LuaJIT

When you code in Lua — code in Lua!

Lua / LuaJIT place in your stack

On Lua and LuaJIT

First and foremost:

- User-configurable business-logic.
- Code that would otherwise be written in C/C++/OCaml.

Frameworks to build webservices with Lua

Our stack

Some of the popular ones:

- Kepler/WSAPI
- OpenResty
- Luvit

We have a "bicycle", built on WSAPI.

What web-problems are we solving with Lua?

Our stack

- Browser and social games.
- Ad networks.
- Other web-services, mobile games etc.

Hardware

Our stack

- Linode
- Hetzner EX6

OS

Our stack

- Xen XCP on top of Ubuntu Sever.
- domU on Ubuntu Server:
  - HTTP frontends (nginx).
  - HTTP backends (32-bit).
  - Workers (32-bit).
  - DB (Redis, MySQL).
  - Aux (Bind, nginx-based config-server, deployment, monitoring etc.).

Backends

Our stack

- nginx
- spawn-fcgi + multiwatch
- LuaJIT 2.0
- FCGI/WSAPI
- Application code

Workers

Our stack

- runit
- LuaJIT 2.0
- Application code

IPC

Our stack

- ØMQ
- Tasks:
  - Replacement for broken signals.
  - In-process cache reset.
  - (Workers get tasks via Redis.)

System tuning

Our stack

- OS
  - bit.ly/kernel-magic (for frontends and backends)
- Redis
  - I/O Scheduler: noop on guests, deadline on host
- nginx
  - worker_rlimit_nofile

"DevOps"

Our stack

- Deployment
- High Availability
- Monitoring

Main libraries

Our stack

- lua-nucleo, lua-aplicado (better — Penlight, telescope)
- slnunicode
- luatexts, luajson
- luasocket, luaposix (better — ljsyscall)
- WSAPI
- lua-zmq
- lua-hiredis (better — ljffi-hiredis)
- luasql-mysql

DSL and code generation

Our stack

- HTTP request handlers.
  - Code (partially statically validated).
  - Docs.
  - (Planned) Smoke-tests.
- SQL schema.
  - "ORM" wrapper code.
  - DB schema patches.
  - Auto-backoffice.
  - Docs.
- bit.ly/lua-dsl-talk
- Common parts of a project are generated from text templates.

Performance

Our stack

- About 160M synthetic hits per day *per EX6-class server* in ad networks.
- About 8K simultaneous active users *per EX6-class server* in online games.

What did we encounter?

Pitfalls

In main:

- Couple "mysterious" problems, due to bugs in early LuaJIT2 betas (all fixed by now).
- Problems, caused by two versions of the same LuaRocks package installed in the system.
- Exploding Redis.
- Lousy Hetzner HDD reliability.

Diagnostics, debugging and monitoring

Pitfalls

- Partial static code validation.
- Runtime validation.
- Autotests.
- GC tuning and monitoring.
- Monitoring for request times, memory usage etc.
- Debugging with logs.

Main unsolved problems

Pitfalls

- Long polling / Comet.
- OS signal handling.
- More efficient CPU usage with HTTP handlers.
- LuaRocks:
  - Can install two versions of the same rock in the system.
  - Can't upgrade a package.

Next generation stack

- Unblocking API with coroutines, no callback. Get inspired by, or adapt OpenResty.
- Complete transition to LuaJIT FFI.
- Consider dropping LuaRocks.
- Drop FCGI, move to epoll and lua-http-parser.
- Simplify the architecture as much as possible. Drop the configuration server. More code generation!
- New DSL design.

Want to know more?

Official Site	lua.org, luajit.org
Wiki	lua-users.org/wiki, wiki.luajit.org
Mailing Lists	lua.org/lua-l.html, luajit.org/list.html
StackOverflow	stackoverflow.com/questions/tagged/Lua
IRC	#lua at irc.freenode.net

Questions?

@agladysh

ag@logiceditor.com

meetup.com/Lua-in-Moscow