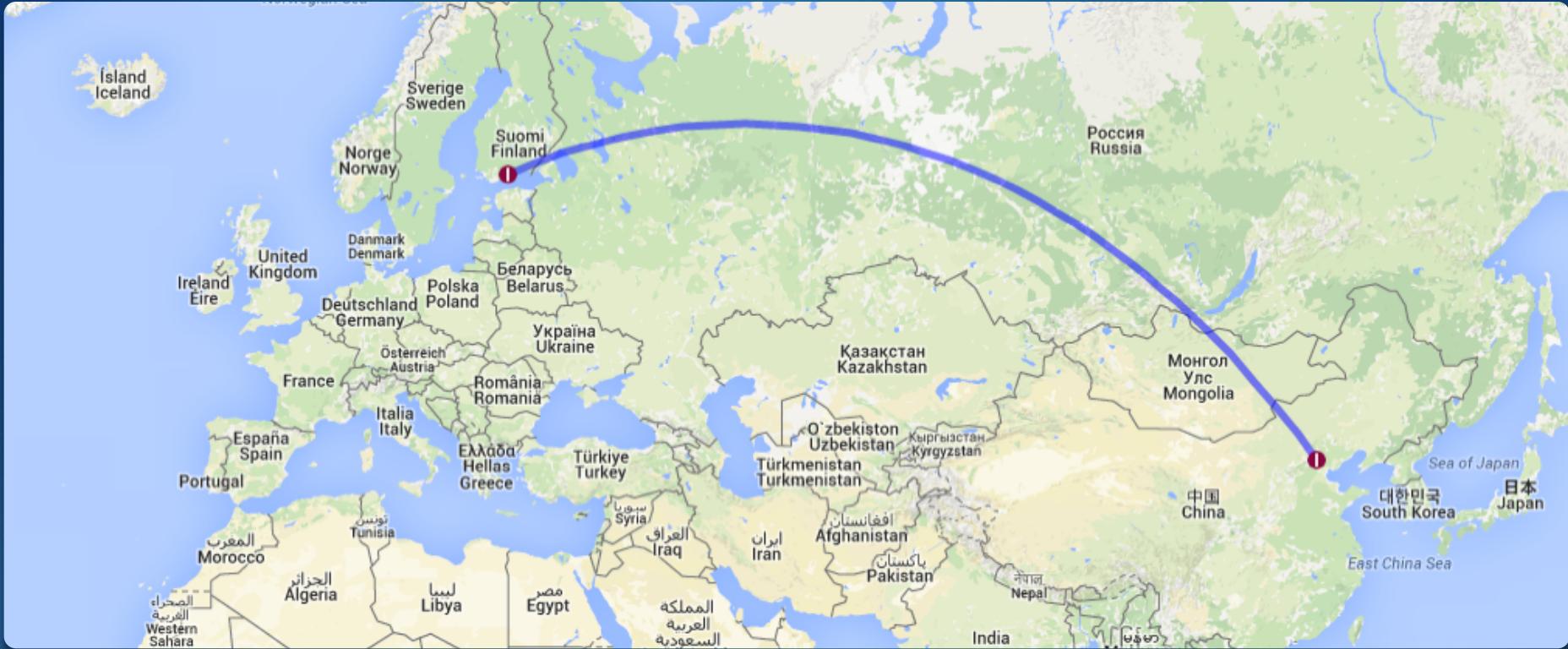


# Developing OpenResty Framework

USING DECOUPLED LIBRARIES

by Aapo Talvensaari @ OpenResty Con 2015



## Where Did I Get Here?

About 6.500 Kilometers – Almost Nothing Compared to The Length of The Great Wall of China



# Who Am I?

PROFESSIONALLY

**WEB PROGRAMMER SINCE 90'S**

ColdFusion ➔ ASP ➔ PHP ➔ Java ➔ ASP.NET ➔ OpenResty

**SYSTEM ADMINISTRATOR**

Linux and Windows Platforms, and Cloud

**JOB**

IT Manager at Aalto Capital Oy

Founder of Talvensaari Solutions Oy

## What to Expect?

In addition to English with Finnish accent, I will show you some of my libraries and how they work together.

Checkout also Leaf Corcoran's Excellent Lapis Framework:

<http://leafo.net/lapis/>

- ▶ Short History
- ▶ Basis for Building Web Applications
  - ▶ Routing
  - ▶ Templating
  - ▶ Validation and Filtering
  - ▶ Sessions
  - ▶ Style Sheets
- ▶ Libraries
- ▶ Kronometrix Analytics

# Why OpenResty?

- ▶ I wanted to expand skills, and learn something new
- ▶ I wanted to get a rid of gateway interfaces – less moving parts
- ▶ A real web server with scripting language support

(Basically a new Apache + mod\_php with a modern twist, e.g. Web Sockets support)

- ▶ I was already using Nginx
- ▶ Alternatives that I looked

(All the alternatives actually looked really nice, but I felt more comfortable with OpenResty)

- ▶ Node.js (I didn't enjoy JavaScript, and it wasn't using a real web server)
- ▶ Python / Ruby (too object oriented, usually behind a gateway interface)
- ▶ Go (not a scripting language, no particular web development focus)
- ▶ Simplicity, both in development and deployment, and performance

# OpenResty Stack



Nginx

by Igor Sysoev et al.

Web Server



OpenResty

by Yichun Zhang et al.

Web Application Server



Lua

by Roberto Ierusalimschy et al.

Programming Language

# Routing

## LUA-RESTY-ROUTE

- ▶ Nginx already does routing! Why oh why?
  - ▶ Makes web development enjoyable! Works as a glue.
  - ▶ Nginx configuration can be fairly static
  - ▶ Nice DSL for Specifying Route Handlers
- ▶ Middleware
- ▶ Before and After Filters
- ▶ Web Sockets Routing
- ▶ Pluggable Matchers
  - ▶ Simple - `ngx.re.match` + pre-defined `:string` and `:number` matchers
  - ▶ Regex - `ngx.re.match`
  - ▶ Match - `string.match`
  - ▶ Equals - plain `==`, no pattern or string matching

```
local r = require "resty.route"
local route = r.new()
-- Use a Middleware
-- (not needed in this example)
route:use "reqargs" {}
-- Add HTTP GET Route
route:get("/hello", function()
    ngx.print "世界你好"
end)
-- Dispatch the Request
route:dispatch()
```

# Templating

## LUA-RESTY-TEMPLATE

- ▶ One of the Top OpenResty Libraries in GitHub
- ▶ It's a Compiling Templating Engine
- ▶ Just Another Way to Write Lua
  - ▶ Almost all you can do in Lua, you can do in Template
  - ▶ Basically it is just plain string concatenation after all  
(white space and line-feed handling is fine tuned)
- ▶ Supports Lua / LuaJIT / OpenResty
- ▶ Handwritten Parser Using `string.find`
- ▶ Features
  - ▶ Layouts
  - ▶ Blocks
  - ▶ Include
  - ▶ Comments
  - ▶ Verbatim / Raw Blocks
  - ▶ Automatic Escaping
  - ▶ Precompilation
  - ▶ Caching
  - ▶ Preloading (WIP)

We've found 181 repository results

Sort: Most stars ▾

### openresty/lua-resty-redis

Perl ★ 437 ⚡ 116

Lua redis client driver for the ngx\_lua based on the cosocket API

Updated 15 days ago

### bungle/lua-resty-template

Lua ★ 170 ⚡ 42

Templating Engine (HTML) for Lua and OpenResty.

Updated 2 days ago

### openresty/lua-resty-mysql

Perl ★ 169 ⚡ 73

Nonblocking Lua MySQL driver library for ngx\_lua

Updated on Apr 23

### pint-sized/lua-resty-http

Perl ★ 166 ⚡ 68

Lua HTTP client cosocket driver for OpenResty / ngx\_lua.

Updated 19 days ago

### openresty/lua-resty-upload

Perl ★ 116 ⚡ 33

Streaming reader and parser for http file uploading based on ngx\_lua cosocket

Updated on Sep 11

```
<!DOCTYPE html>
<html lang="{{lang}}>
<head>
  <meta charset="{{charset}}>
  <title>{{title}}</title>
  {*blocks.styles*}
  {#
    Commented out, because it causes a problems (see issue #35):
    <script src="js/analytics.js"></script>
  //}
  {-head_scripts-}
  <script src="js/app.js"></script>
  {-head_scripts-}
</head>
<body>
{-raw-}
  Everything inside here is outputted as is, e.g. {{not evaluated}}
{-raw-}
{{(include/header.html)}}
{{[concat({"include/navigation", access.level, ".html"}, "-")]}}
<section id="main">
  {* view *}
</section>
{{(include/footer.html)}}
{%- for _, script in ipairs(scripts) do %}
  <script src="{{script}}></script>
{%- end %}
{*blocks.head_scripts*}
</body>
</html>
```

```
context=... or {}
local function include(v, c)
    return template.compile(v)(c or context)
end
local ___,blocks,layout={},blocks or {}
____[#____+1]==[[
<!DOCTYPE html>
<html lang=""]=
____[#____+1]=template.escape(lang)
____[#____+1]==[[
">
<head>
    <meta charset=""]=
____[#____+1]=template.escape(charset)
____[#____+1]==[[
"">
    <title>]=]
____[#____+1]=template.escape(title)
____[#____+1]==[[
</title>
    ]=]
____[#____+1]=template.output(blocks.styles)
blocks["head_scripts"]=include=[[
    <script src="js/app.js"></script>
    ]=]
____[#____+1]==[[
</head>
<body>
]=]
____[#____+1]==[[
    Everything inside here is outputted as is, eg. {{not evaluated}}
]=]
```

```
____[# ____+1]=include( [=include/header.html]=)
____[# ____+1]=[=[
]
____[# ____+1]=include(concat({"include/navigation", access.level, ".html"}, "-"))
____[# ____+1]=[=[
<section id="main">
    ]=
____[# ____+1]=template.output( view )
____[# ____+1]=[=[
</section>
]=
____[# ____+1]=include( [=include/footer.html]=)
____[# ____+1]=[=[
]
for _, script in ipairs(scripts) do
    ____[# ____+1]=[=[
        <script src="]="
        ____[# ____+1]=template.escape(script)
        ____[# ____+1]=[=[
    "></script>
]=
end
____[# ____+1]=template.output(blocks.head_scripts)
____[# ____+1]=[=[
</body>
</html>]=]

return layout and include(layout, setmetatable({view=template.concat(___), blocks=blocks}, {__index=context})) or template.concat(_)
```

# Validation and Filtering

## LUA-RESTY-VALIDATION

- ▶ Both Validation and Filtering
- ▶ Validation and Filter Chaining
- ▶ Reusable Validators  
(Define Once, Use in Many Places)
- ▶ Automatic Conditional Validators
- ▶ Group Validators and Filters
- ▶ Easy to Extend
- ▶ Report Errors in Frontend, JSON Friendly
- ▶ Send Data to Backend
- ▶ Stop Validators  
(e.g. optional)

```
local v      = require "resty.validation"
local validate = {
    nick      = v.string.trim:minlen(2),
    email     = v.string.trim.email,
    password  = v.string.trim:minlen(8)
}
-- First we create single validators
-- for each form field
local register = v.new{
    nick      = validate.nick,
    email     = validate.email,
    email2    = validate.email,
    password  = validate.password,
    password2 = validate.password
}
-- Next we create group validators for email
-- and password:
register:compare "email == email2"
register:compare "password == password2"
-- And finally we return from this forms module
return {
    register = register
}
```

# Sessions

## LUA-RESTY-SESSION

- ▶ Secure Defaults
- ▶ Configurable from Nginx Config or Lua Code
- ▶ Plugins for
  - ▶ Client and Server Storages  
(Cookie, Shared Memory, Memcache, Redis)
  - ▶ Ciphers  
(AES, HMAC)
  - ▶ Encoders  
(Base64 and Base16)
  - ▶ Serializers  
(JSON)
  - ▶ Identifiers  
(WIP, Random, JWT, UUID)

```
local session = require "resty.session"

-- Construct and start a session
-- (new, and open also available)
local s = session.start()

-- Store some data to session
s.data.name = "OpenResty Fan"

-- Save a session
s:save()

-- Destroy a session
s:destroy()

-- Regenerate a session
-- (e.g. when security context changes)
s:regenerate()
```



# Style Sheets

LUA-RESTY-SASS

- ▶ Syntactically Awesome Style Sheets
- ▶ LuaJIT FFI Bindings to libSass
- ▶ Automatic Online Compilers for OpenResty
- ▶ Caching to Normal CSS Files Once Compiled
- ▶ Save-And-Refresh Development
  - ▶ A Good and a Bad Thing: Errors in Sass File prevents compiling to CSS  
Errors are catched earlier, but works differently than with plain CSS
  - ▶ Deploy Sass Files Directly to The Production, No Build Needed
  - ▶ More about Sass at <http://sass-lang.com/>

# Demo

HOW THIS ALL WORKS TOGETHER?

Source Code Available at [github.com/bungle/iresty](https://github.com/bungle/iresty)

# Libraries

## COMMON NEEDS IN WEB DEVELOPMENT

- ▶ Excel
- ▶ Cryptography
- ▶ Universally Unique Identifiers
- ▶ Audio Metadata
- ▶ Markdown
- ▶ File Information
- ▶ Translations
- ▶ Unicode
- ▶ JSON Pretty Formatting
- ▶ LUA-RESTY-LIBXL
- ▶ LUA-RESTY-NETTLE
- ▶ LUA-RESTY-UUID
- ▶ LUA-RESTY-TAGLIB
- ▶ LUA-RESTY-HOEDOWN
- ▶ LUA-RESTY-FILEINFO
- ▶ LUA-RESTY-GETTEXT
- ▶ LUA-RESTY-UNISTRING
- ▶ LUA-RESTY-PRETTYCJSON

CHECK OUT MY GITHUB ACCOUNT FOR MORE

# Excel

## LUA-RESTY-LIBXL

LuaJIT FFI to a LibXL Library – a Library that Can Read and Write Excel Files



18

Developing OpenResty Framework  
Using Decoupled Libraries

- ▶ LibXL
  - ▶ Proprietary Library (from Slovakia)
  - ▶ Supports XLS and XLSX Formats
  - ▶ Pricing Starts from \$ 199.00
  - ▶ Source Code for The Library is Available for \$ 2499.00
  - ▶ High Performance
  - ▶ No Extra Dependencies
- ▶ lua-resty-libxl
  - ▶ Nice Lua API (almost full featured\*)

\* some Sheet APIs are still not finished

```
local excel = require "resty.libxl.book"
local book = excel.new()
book:load "demo.xlsx"
local sheet = book.sheets[1]
local value = sheet:read(1, 1)
sheet:write(1, 1, value + math.pi)
book:save "output.xlsx"
```

- ▶ Features
  - ▶ Pictures
  - ▶ Formats
  - ▶ Fonts and Colors
  - ▶ Hyperlinks
  - ▶ Equations

# Cryptography

## LUA-RESTY-NETTLE

LuaJIT FFI Bindings to GNU Nettle Library – a Low Level Cryptography Library

- ▶ Hash Functions
  - ▶ SHA1, SHA2, SHA3
  - ▶ MD2, MD4, MD5
  - ▶ RIPEMD160, GOSTHASH94
- ▶ Key Derivation Functions
  - ▶ PBKDF2
  - ▶ SHA1
  - ▶ SHA2
- ▶ Keyed Hash Functions
  - ▶ HMAC
    - ▶ SHA1, SHA2
    - ▶ MD5
    - ▶ RIPEMD160
  - ▶ UMAC
  - ▶ Poly1305
- ▶ Randomness
  - ▶ Yarrow
  - ▶ Knuth's Lagged Fibonacci
- ▶ ASCII Encoding
  - ▶ Base16, Base64, URL-safe Base64

# Cryptography

LUA-RESTY-NETTLE

- ▶ Cipher Functions
  - ▶ AES
  - ▶ ARCFOUR
  - ▶ ARCTWO
  - ▶ BLOWFISH
  - ▶ Camellia
  - ▶ CAST128
  - ▶ ChaCha
  - ▶ DES
  - ▶ DES3
  - ▶ Salsa20
  - ▶ SERPENT
  - ▶ TWOFISH
- ▶ Cipher Modes
  - ▶ ECB
  - ▶ CBC
  - ▶ CTR
- ▶ AEAD
  - ▶ Authenticated Encryption with Associated Data
    - ▶ EAX for AES
    - ▶ GCM for AES / Camellia
    - ▶ CCM for AES
    - ▶ ChaCha-Poly1305
- ▶ Public Key Algorithms
  - ▶ Bindings for these are still work in progress, ☺.

# Kronometrix

A REAL-TIME ANALYTICS APPLIANCE DESIGNED FOR TIME SERIES DATA ANALYSIS

- ▶ Build on OpenResty and Lua
- ▶ Using Redis as a Backend Store
- ▶ 1,000 Recording Devices per Appliance in Real-Time
- ▶ Data Recorders
  - ▶ Aviation Meteorology
  - ▶ Computer Performance Metrics
    - ▶ Windows
    - ▶ Linux
    - ▶ BSD
    - ▶ Solaris
  - ▶ Climatology
- ▶ Architecture of The Appliance
  - ▶ Authentication Layer
    - ▶ OpenResty Server
    - ▶ Redis Authentication Database
  - ▶ Messaging Layer
    - ▶ OpenResty Server
  - ▶ Kernel Layer
    - ▶ OpenResty Server
    - ▶ Redis Statistics Database
  - ▶ Aggregate Layer
    - ▶ OpenResty Server
    - ▶ Redis Aggregates Database

# OpenResty in a Future

IT'S BRIGHT FOR SURE

## What I Will Be Doing?

- ▶ Adding Documentation
- ▶ Adding Tests
- ▶ Maintaining Existing Libraries
- ▶ More Libraries, and Bindings
  - ▶ lua-resty-password / -auth
  - ▶ lua-resty-chromelogger
  - ▶ lua-resty-cloudflare
- ▶ Start Contributing on **ngx\_lua**
- ▶ A Modern Open Source CMS

## What I Would Like to See?

- ▶ Uniting Chinese and Western Communities
- ▶ Package Management
- ▶ File APIs (Using Nginx File APIs?)
- ▶ Official Packages for Platforms
- ▶ LuaJIT Enhancements
  - ▶ Less NYIs
  - ▶ String Buffer
  - ▶ Lua 5.2 / 5.3 Support / Uncertainty
- ▶ Optimizing for LuaJIT is Hard

# Questions?



aapo.talvensaari@gmail.com



bungle@github