Etiene Dalcol

Email: dalcol@etiene.net
Twitter: @etiene_d
Why I started learning

→ It looks cool
(I heard you could make games with it)
Why I started learning

➔ It looks cool
   (I heard you could make games with it)
➔ It’s made in my home country
   (In my university to be more precise)
Why I started learning

➔ It looks cool
   (I heard you could make games with it)
➔ It’s made in my home country
   (In my university to be more precise)
➔ I could brag about it later
Why I started coding **Sailor**

→ I needed something real!
Why I started coding **Sailor**

- I needed something real!
- I could use my previous knowledge to go faster
Sailor!

Hello world!
Welcome to your first Sailor application

This is the default view file, you will find it under /views/main/index.lp. This layout is provided by Twitter Bootstrap. You can use this layout as is, modify it, or get new layouts and place them under /controllers/main.lua.
And what exactly is **Sailor**

- It’s an MVC web framework
- Completely written in Lua
- Compatible with Apache (mod_lua), Nginx (OpenResty) and Mongoose
- Compatible with Linux, Windows and Mac
- MIT License
- Pre alpha v0.2 - Mars
Yes, the version names are inspired in Sailor Moon
What does it do

➔ Simple URL routing
➔ Session, cookies, include, redirect…
➔ Lua Pages parsing from cgilua
➔ Mail sending
➔ Simple Objet Relational-Mapping (MySQL, PostgreSQL…)
➔ Input Validation (valua)
➔ Basic login and authentication
What does it do

➔ Form generation
➔ Twitter bootstrap integration
➔ App generator (Linux and Mac only)
What does it do

→ Form generation
→ Twitter bootstrap integration
→ App generator (Linux and Mac only)

Premium feature!

→ Lua at client
What’s bad about it

➔ Early development
➔ Things are changing fast
➔ It still lacks features
➔ Documentation
Server-side languages

<table>
<thead>
<tr>
<th>Language</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHP</td>
<td>82.1%</td>
</tr>
<tr>
<td>ASP.NET</td>
<td>17.3%</td>
</tr>
<tr>
<td>Java</td>
<td>2.7%</td>
</tr>
<tr>
<td>ColdFusion</td>
<td>0.7%</td>
</tr>
<tr>
<td>Perl</td>
<td>0.5%</td>
</tr>
<tr>
<td>Ruby</td>
<td>0.5%</td>
</tr>
<tr>
<td>Python</td>
<td>0.2%</td>
</tr>
<tr>
<td>JavaScript</td>
<td>0.1%</td>
</tr>
<tr>
<td>Erlang</td>
<td>0.1%</td>
</tr>
</tbody>
</table>

Note: a website may use more than one server-side programming language.

W3Techs.com, 13 September 2014

Client-side languages

<table>
<thead>
<tr>
<th>Language</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>11.6%</td>
</tr>
<tr>
<td>JavaScript</td>
<td>88.2%</td>
</tr>
<tr>
<td>Flash</td>
<td>13.0%</td>
</tr>
<tr>
<td>Silverlight</td>
<td>0.2%</td>
</tr>
<tr>
<td>Java</td>
<td>0.1%</td>
</tr>
</tbody>
</table>

Note: a website may use more than one client-side programming language.

W3Techs.com, 13 September 2014
## Content Management Systems

<table>
<thead>
<tr>
<th>System</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>62.1%</td>
</tr>
<tr>
<td>WordPress</td>
<td>60.9%</td>
</tr>
<tr>
<td>Joomla</td>
<td>8.0%</td>
</tr>
<tr>
<td>Drupal</td>
<td>5.2%</td>
</tr>
<tr>
<td>Blogger</td>
<td>2.9%</td>
</tr>
<tr>
<td>Magento</td>
<td>2.7%</td>
</tr>
<tr>
<td>TYPO3</td>
<td>1.6%</td>
</tr>
<tr>
<td>PrestaShop</td>
<td>1.2%</td>
</tr>
<tr>
<td>vBulletin</td>
<td>1.2%</td>
</tr>
<tr>
<td>Bitrix</td>
<td>1.0%</td>
</tr>
<tr>
<td>DataLife Engine</td>
<td>0.9%</td>
</tr>
<tr>
<td>OpenCart</td>
<td>0.8%</td>
</tr>
<tr>
<td>DotNetNuke</td>
<td>0.6%</td>
</tr>
<tr>
<td>ExpressionEngine</td>
<td>0.6%</td>
</tr>
<tr>
<td>phpBB</td>
<td>0.6%</td>
</tr>
<tr>
<td>osCommerce</td>
<td>0.6%</td>
</tr>
<tr>
<td>Discuz!</td>
<td>0.6%</td>
</tr>
<tr>
<td>Shopify</td>
<td>0.6%</td>
</tr>
<tr>
<td>Other</td>
<td>0.2%</td>
</tr>
</tbody>
</table>
## Web servers

<table>
<thead>
<tr>
<th>Server</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache</td>
<td>59.6%</td>
</tr>
<tr>
<td>Nginx</td>
<td>21.9%</td>
</tr>
<tr>
<td>Microsoft-IIS</td>
<td>13.6%</td>
</tr>
<tr>
<td>LiteSpeed</td>
<td>2.0%</td>
</tr>
<tr>
<td>Google Servers</td>
<td>1.3%</td>
</tr>
<tr>
<td>Tomcat</td>
<td>0.4%</td>
</tr>
<tr>
<td>Lighttpd</td>
<td>0.3%</td>
</tr>
<tr>
<td>Yahoo Traffic Server</td>
<td>0.2%</td>
</tr>
<tr>
<td>Tengine</td>
<td>0.1%</td>
</tr>
<tr>
<td>IBM Servers</td>
<td>0.1%</td>
</tr>
<tr>
<td>Node.js</td>
<td>0.1%</td>
</tr>
<tr>
<td>Oracle Servers</td>
<td>0.1%</td>
</tr>
<tr>
<td>Jetty</td>
<td>0.1%</td>
</tr>
<tr>
<td>Zeus</td>
<td>0.1%</td>
</tr>
<tr>
<td>Resin</td>
<td>0.1%</td>
</tr>
</tbody>
</table>

*Percentages of websites using various web servers*
“I have myself developed Web sites with pure C++, Java, C#, PHP, and Python. The easiest way to go was definitely Python. If the libraries existed, Lua would be not quite as easy to use as Python, but probably quite a bit more efficient; I think it would become my first choice... if the libraries existed.” Michael Gogins

“Recently there was some discussion about mod_lua on the Apache developers mailing list. I mentioned there that I feel Lua could replace PHP as the number one web scripting language if mod_lua were stable (i.e. not still in beta) and it were implemented well (not making some of PHP's mistakes such as putting everything in the global scope with no consistent naming or parameter schemes). I've wanted to use Lua for all the things I currently use PHP for ever since I discovered it.” Rena
Average across all speed tests
(less is better)

Pretty fast
Super fast!
More Lua for Web

~ Early stage

cgilua (1995)

Kepler Project (2003)
Other frameworks

➔ Orbit (2007)
  ◆ Least known
  ◆ No significant updates since 2010
  ◆ Closest to Sailor’s concept

➔ Luvit (2011)
  ◆ Most popular
  ◆ Intense development
  ◆ node.js port 2-4x faster
Other frameworks

➔ Lapis (2012)
  ◆ Intense development
  ◆ Moonscript and Lua
  ◆ Very well documented
  ◆ OpenResty only
  ◆ Templater
  ◆ Not MVC

➔ Others
  ◆ Complicated, abandoned, poorly documented, license issues or I never heard about it...
Back to **Sailor!**

Getting it:

→ LuaRocks or Github
A Setup Example

➔ Install Lua
➔ Install LuaRocks
➔ Install database server (MySQL for example)
➔ Install apache2 (>=2.4.7)
➔ Create app!

$ luarocks install sailor
$ luarocks install luasql-mysql
$ sailor_create 'Test App' /var/www
App Structure

/conf
/controllers
/layouts
/models
/pub
/runtime
/views
local site = {}

function site.index(page)
    local ctr_msg = "Hi!"
    page:render('index',{ctr_msg = ctr_msg})
end

return site
local site = {}

function site.index(page)
    local ctr_msg = "Hi!"
    page:render('index',{ctr_msg = ctr_msg})
end

return site

<html>
<head>
    <title>Example Page</title>
</head>
<body>
    <?lua
    if ctr_msg then
        page:print("A message from the controller: "..ctr_msg)
    end
    ?>

    <?lua@both
    msg = "Hello, World!"
    ?>

    <br/>
    A message from the server: <strong><?=msg?></strong>

    <?lua@client
    js.window.alert("A message from the client: "..msg)
    ?>
</body>
</html>
Sailor! A Lua MVC framework.

Signup to be updated on our releases

you@yourmail.com
Notify me

Etiene Dalcol
sailorproject.org
github.com/Etiene/sailor
dalcol@etiene.net
twitter.com/etiene_d