Lua/APR: An extended standard library* for Lua

Peter Odding

September 9, 2011

Apache
Portable Runtime Project
http://apr.apache.org/

Abstract

Lua is a very elegant programming language, both because of its conceptual simplicity and the small size of its implementation, but this small size comes at a price: Lua's operating system interfaces are quite minimal and (in a sense) this makes Lua a second-class citizen on popular platforms like Windows and UNIX systems. My solution was to write a binding to the Apache Portable Runtime.

Contents

About me

- Hi all, I'm Peter Odding from the Netherlands
- Been programming since I was 12 (I'm now 24)
- Just finished a computer science study & received my bachelor's degree this July
- Started working as a Python developer and parttime server system administrator
- In case anyone wants to contact me: peter@peterodding.com

Why the Apache Portable Runtime?

- Around 2006 I fell in love with Lua :-)
- However I was quickly disappointed by the lack of cross platform operating system interfaces!
- In 2007 I decided to create a binding to one of the well known ‘portable runtimes’:
  - Apache Portable Runtime (APR)
    - Very comprehensive, lots of tests
  - Netscape Portable Runtime (NSPR)
    - Seemed less comprehensive than APR
  - ACE, commonc++, Qt (all C++)
    - All disqualified because they’re written in C++ which is way over my head...

The origins of APR

- Started life in the Apache web server code base
- Eventually split off into a separate library
- Insists on using memory pools everywhere (which makes sense in a server context)
- Very comprehensive, dozens of modules:

  directory handling, filename matching, file I/O, network sockets, multi threading, shared memory, process management, signal handling, option parsing, cryptography, date handling, relational database interfaces, LDAP connection handling, option parsing, ...

Getting started ... took a while

- Started writing in 2007
- Didn't publish until September 2010
- What happened in between?
    - Back in 2007 I didn't know C and very naively thought "How hard can it be?!"
    - Learned more than I ever wanted to know about memory (de)allocation, off by one errors, segmentation faults, debugging binary code, etc.
    - Basically "I bit of more than I could chew", or rather it took me quite a while to digest :-)
- In the end I'm glad I persisted – user feedback now motivates me to keep developing Lua/APR

Design choices & technical challenges

- **Memory pools:** completely hidden from Lua
- **Multi threading:** using a very simplified model (`create()`, `status()`, `join()`)
- **I/O interface:** same as Lua, a real pain to implement on top of APR (worth it though!)
- **Error handling:** APR error codes are not portable, so using strings instead
- **Code generation:** boring stuff like mapping of error codes and signal numbers to strings
- **Inline documentation:** Docs in comments, extracted using custom script to generate HTML docs

Example: HTTP client

```
function download(url)
  local socket = apr.socket_create()
  local components = apr.uri_parse(url)
  local port = components.port or apr.uri_port_of_scheme(components.scheme)
  local pathinfo = apr.uri_unparse(components, 'pathinfo')
  socket:connect(components.hostname, port)
  socket:write('GET ', pathinfo, ' HTTP/1.0\r\n',
               'Host: ', components.hostname, '\r\n',
               '\r\n')
  local _, status, reason = socket:read():match '^(%S+)%s+(%S+)%s+(.-)$'
  local headers, data = apr.parse_headers(socket:read '*a')
  if status:find '^30[123]$' and headers.Location then
    return download(headers.Location)
  elseif status == '200' then
    return data
  else
    error(reason)
  end
end
print(download('http://lua.org/'))
```

Master plan: Rewrite Apache in Lua

My ultimate goal with Lua/APR is to be able to rewrite the core of Apache in Lua. If I ever succeed I can consider Lua/APR to be finished. Until a new version of APR is released that is :-)

Thank you! Questions anyone?

Thanks for listening! If you're interested in Lua/APR you can find more information in the following places:

- peterodding.com/code/lua/apr
- github.com/xolox/lua-apr

If you want to try Lua/APR, the following packages are available:

- `luarocks install lua-apr`
  (mind the dependencies)
- `apt-get install liblua5.1-apr1`
  (available on Debian and Ubuntu)