LuaDist

Providing the Batteries

Peter Drahoš

About LuaDist

- Successor to LuaForWindows ?
- Collection of Lua related modules and software
- Everything is built using CMake
- Optional Lua based package management CLI
- Portable and easy to distribute
- Download at www.luadist.org
- Repository at www.github.com/LuaDist

Talk Outline

- LuaDist goals
- Basic use cases
- Repositories in detail
- Modules in detail
- Problems
- Future plans

LuaDist Goals

- Unified build system for all modules
- No external dependencies, build the universe
- Mixed binary and source distribution
- Focus on auto-configuration and ease of building
- Focus on ease of application distribution

Basic use cases

- Batteries included binary distribution
- Deployment directories
- Installing modules using the CLI tool
- Building modules manually
- Using IDEs for development
- Integrating LuaDist modules into other apps
- Distribution of applications

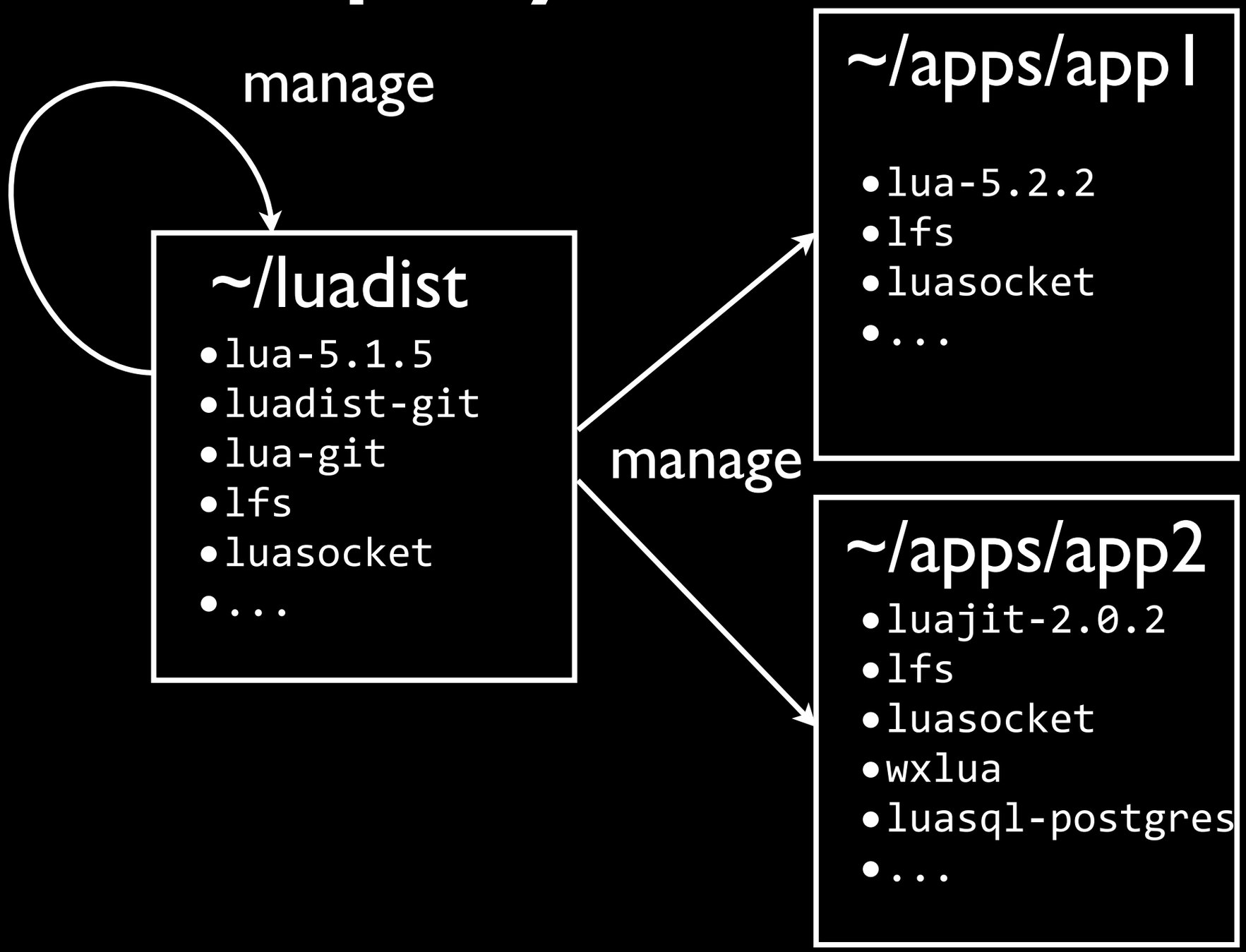
Batteries Included

- Simply download binary package from luadist.org
- Almost a drop in replacement for LuaForWindows
- Includes ZeroBraneStudio by default
- At the moment only for Lua-5.1
- It can however be used install Lua-5.2 and LuaJIT-2
- No git, CMake or C Compiler required*

Deployment Directory

- Manage multiple Lua installations
  - Lua 5.1, Lua 5.2 and LuaJIT 2.0
- Each installation is fully contained in a directory
  - No need to install, No registry
  - No dependencies, No Visual Studio runtimes
- Each application can use its own deployment dir
  - Easy for distribution to end users

Deployment Directory



CLI interface

- The command-line interface is very simple

```
> # luadist deployment install package_name
> luadist ~/Lua-5.1.5 install lua-5.1.5
```

- Multiple modules can be installed

```
> luadist ~/LuaJIT-2.0.2 install luajit luasocket
```

- You can add modules to any deployment dir.
- Even to the one luadist is in

```
> luadist install luaexpat
```

Deployment Structure

- Follows Unix-like directory structure
- *bin* - Contains binaries (*lua*, *luac*, *yourapp*)
- *lib* - Contains libraries (*liblua.so*, *libexpat.so* ...)
- *include* - Contains headers (*lua.h* ..)
- *share* - Storage for additional files, tests, docs.
- *tmp* - Temporary storage, local to avoid security

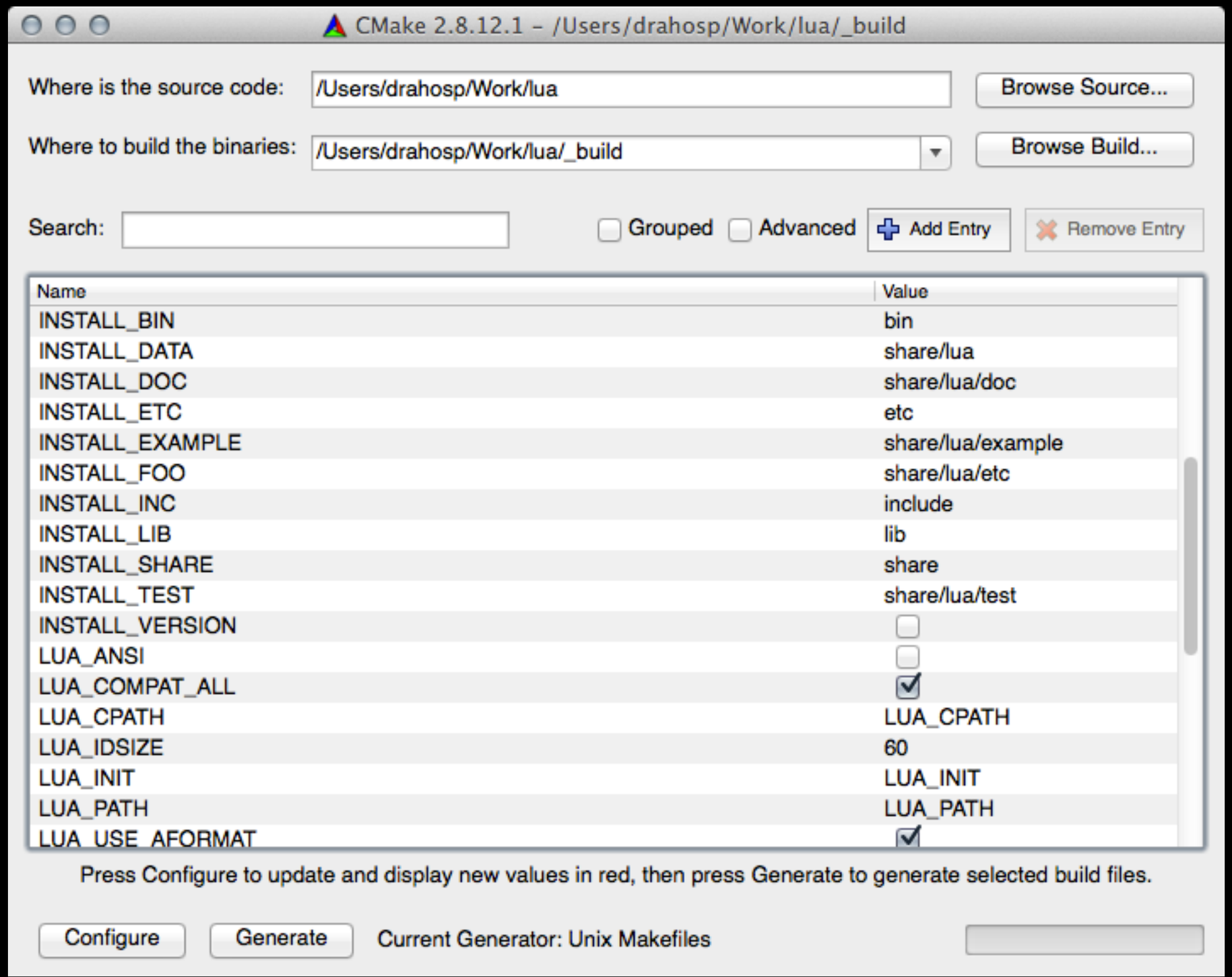
Building Manually

- I just want to build Lua and some modules !
- Make sure you have cmake, git and C compiler

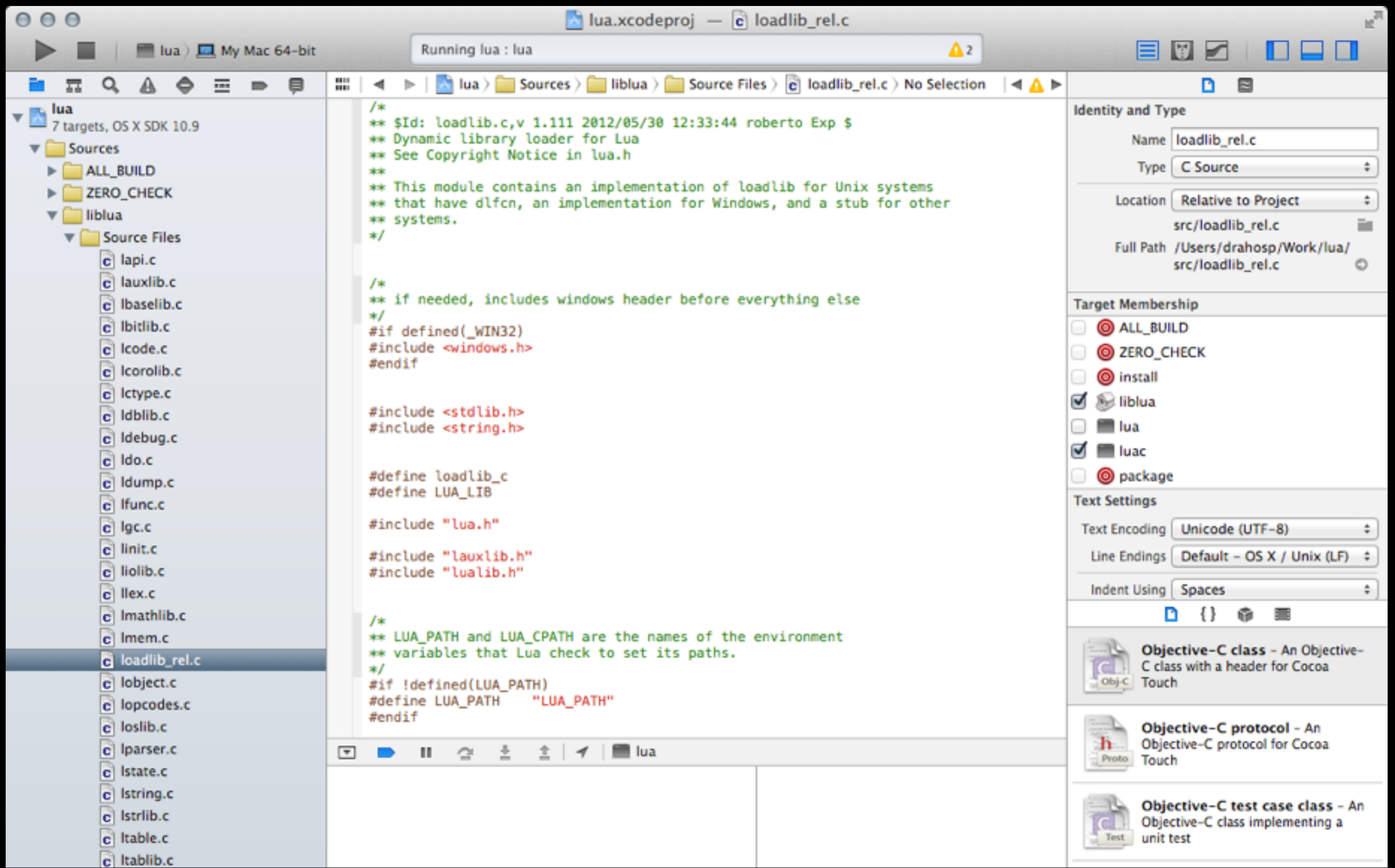
```
> cd ~/Work
> git clone https://github.com/lua.git
> cd lua
> cmake . -DCMAKE_INSTALL_PREFIX=~/LuaDist
> # alternatively use ccmake/cmake-gui to customize
> cmake --build . --target install
```

- Can be included in CMake projects using:
  - See: EXTERNALPROJECT_ADD

Building Manually



IDE Support


lua.xcodeproj — loadlib_rel.c
lua 〉 My Mac 64-bit
Running lua : lua
2
lua
7 targets, OS X SDK 10.9
Sources
ALL_BUILD
ZERO_CHECK
liblua
Source Files
lapi.c
lauxlib.c
lbaselib.c
lbitlib.c
lcode.c
lcorolib.c
lctype.c
ldblib.c
ldebug.c
ldo.c
ldump.c
lfunc.c
lgc.c
linit.c
liolib.c
llex.c
lmathlib.c
lmem.c
loadlib_rel.c
lobject.c
lopcodes.c
loslib.c
lparser.c
lstate.c
lstring.c
lstrlib.c
ltable.c
ltablib.c
lua 〉 Sources 〉 liblua 〉 Source Files 〉 loadlib_rel.c 〉 No Selection
/*
** $Id: loadlib.c,v 1.111 2012/05/30 12:33:44 roberto Exp $
** Dynamic library loader for Lua
** See Copyright Notice in lua.h
**
** This module contains an implementation of loadlib for Unix systems
** that have dlfcn, an implementation for Windows, and a stub for other
** systems.
*/
/*
** if needed, includes windows header before everything else
*/
#if defined(_WIN32)
#include <windows.h>
#endif
#include <stdlib.h>
#include <string.h>
#define loadlib_c
#define LUA_LIB
#include "lua.h"
#include "lauxlib.h"
#include "lualib.h"
/*
** LUA_PATH and LUA_CPATH are the names of the environment
** variables that Lua check to set its paths.
*/
#if !defined(LUA_PATH)
#define LUA_PATH   "LUA_PATH"
#endif
lua
Identity and Type
Name loadlib_rel.c
Type C Source
Location Relative to Project
src/loadlib_rel.c
Full Path /Users/drahosp/Work/lua/src/loadlib_rel.c
Target Membership
ALL_BUILD
ZERO_CHECK
install
liblua
lua
luac
package
Text Settings
Text Encoding Unicode (UTF-8)
Line Endings Default – OS X / Unix (LF)
Indent Using Spaces
Objective-C class – An Objective-C class with a header for Cocoa Touch
Objective-C protocol – An Objective-C protocol for Cocoa Touch
Objective-C test case class – An Objective-C class implementing a unit test

Repositories

- Everything is stored in GIT
- Each module has its own repository
- Repositories contain source and binaries
- Currently using github.com as host
- The repository manifest is also a GIT repository
  http://github.com/LuaDist/Repository

The Repository

- Links all modules to central GIT repository using submodule functionality
- Central Issue Tracker on GitHub
- Module manifest for the CLI tool
- Useful for development purposes
- Provides *install* shell script for quick manual installs

GIT Tags

- Simple versions are for source only
  - tag “5.1.5.” marks the version of the source
- Binaries are tagged with Architecture and Type
  - tag “5.1.5-Windows-x86” marks the binary
- Binaries are also stored in separate branches
  - branch “Windows-x86” for the above
- Yes, you can download these manually or using git

Dist Info

- Each repository contains *dist.info* file at its root
- The file contains metadata for the module
- VERY similar to rockspec files used in LuaRocks
- Primary purpose is dependency specification
- DOES NOT contain any build information
- When module is installed the file is stored in /share/luadist/git/[module]/dist.info
- Once installed it also contains all associated files

Advanced Features

- Packages can “provide” multiple modules.
  e.g LuaJIT 2.0.2 provides Lua-5.1.5 and bitop
- Packages can install only certain “components”
  runtime, library, documentation, test etc..
- The luadist-git package provides a Lua interface
  dist = require “dist”
  dist.install (“luaexpat”)

Problems

- LuaDist is maintained - Hard to add modules to
  - Modules get outdated fast
    - Partial solution is to support LuaRocks “builtin” type rock
- Binary packages need API related information in dependencies
  - Especially when source is Lua-5.2 and Lua-5.1 compatible

Related Projects

- LuaCI - Continuous Integration Service for Lua
  - Multiple VMs that test and generate binaries
    - We plan to test module quality
      - eg. No globals, Has Docs, Has Tests ...
- ZeroBrane Studio - LuaDist integration in progress

Thanks

- Peter Kapec
- Marek Šrank
- Michal Kottman
- David Manura
- Steve Donovan
- Hisham Muhammad
- Andrew Starks
- Paul Kluchenko