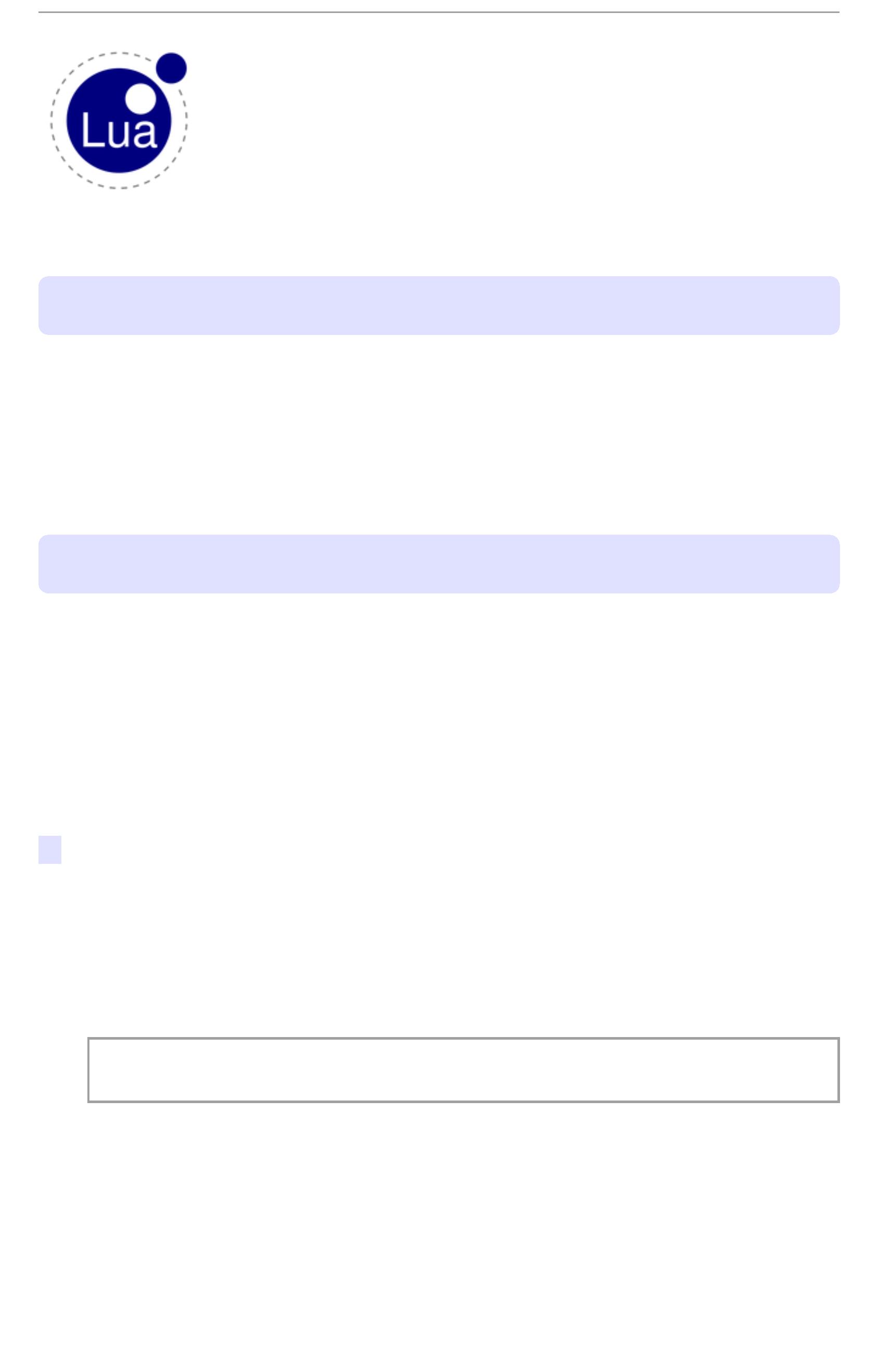
*Welcome to Lua 5.2*



about · installation · changes · license · [reference manual](file:///C:/Cpp/skia/third_party/lua/doc/contents.html)

*About Lua*

Lua is a powerful, fast, lightweight, embeddable scripting language developed by a [team](http://www.lua.org/authors.html) at[PUC-Rio](http://www.puc-rio.br/), the Pontifical Catholic University of Rio de Janeiro in Brazil. Lua is free softwareused in many products and projects around the world.

[Lua's](http://www.lua.org/about.html) [official web site](http://www.lua.org/) [provides complete information about Lua, including an executivesummary and updated documentation, especially the reference manual, which may differ](http://www.lua.org/about.html)slightly from the [local copy](file:///C:/Cpp/skia/third_party/lua/doc/contents.html) distributed in this package.

*Installing Lua*

Lua is distributed in [source](http://www.lua.org/ftp/) form. You need to build it before using it. Building Lua should bestraightforward because Lua is implemented in pure ANSI C and compiles unmodified in allknown platforms that have an ANSI C compiler. Lua also compiles unmodified as C++. Theinstructions given below for building Lua are for Unix-like platforms. See also instructions forother systems and customization options.

If you don't have the time or the inclination to compile Lua yourself, get a binary from[LuaBinaries](http://lua-users.org/wiki/LuaBinaries). Try also [Lua for Windows](http://luaforwindows.luaforge.net/), an easy-to-use distribution of Lua that includes manyuseful libraries.

*Building Lua*

In most Unix-like platforms, simply do "make" with a suitable target. Here are the details.

1. Open a terminal window and move to the top-level directory, which is named lua- 5.2.2. The Makefile there controls both the build process and the installation process.

2. Do "make" and see if your platform is listed. The platforms currently supported are:

aix ansi bsd freebsd generic linux macosx mingw posix solaris

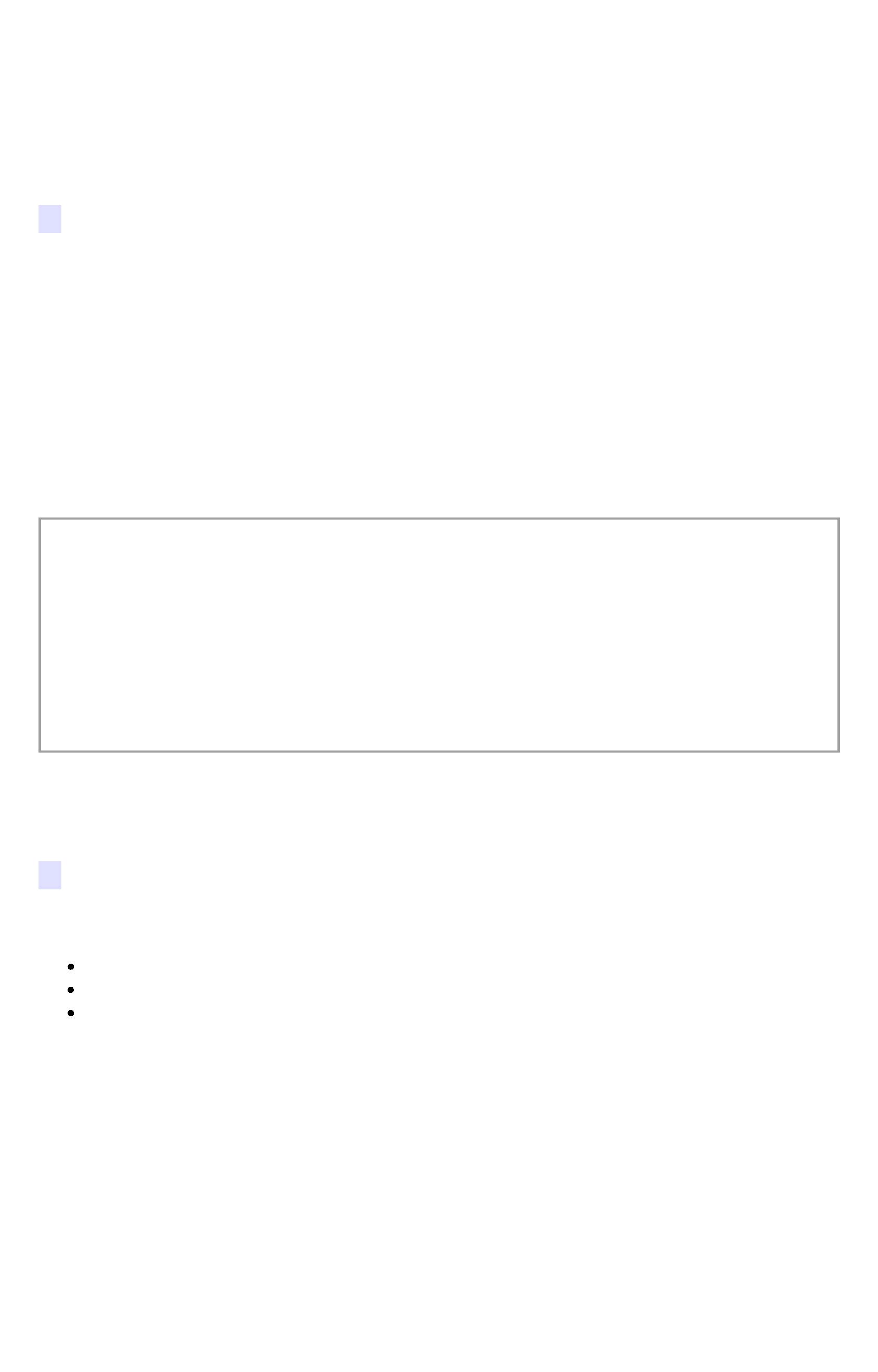
If your platform is listed, just do "make xxx", where xxx is your platform name.

If your platform is not listed, try the closest one or posix, generic, ansi, in this order.

3. The compilation takes only a few moments and produces three files in the src

directory: lua (the interpreter), luac (the compiler), and liblua.a (the library).

4. To check that Lua has been built correctly, do "make test" after building Lua. This will



run the interpreter and print its version string.

If you're running Linux and get compilation errors, make sure you have installed thereadline development package. If you get link errors after that, then try "make linuxMYLIBS=-ltermcap".

*Installing Lua*

Once you have built Lua, you may want to install it in an official place in your system. In this case, do "make install". The official place and the way to install files are defined in the Makefile. You'll probably need the right permissions to install files.

To build and install Lua in one step, do "make xxx install", where xxx is your platform name.

To install Lua locally, do "make local". This will create a directory install withsubdirectories bin, include, lib, man, and install Lua as listed below. To install Lua locally,but in some other directory, do "make install INSTALL\_TOP=xxx", where xxx is yourchosen directory.

bin:

lua luac

include:

lua.h luaconf.h lualib.h lauxlib.h lua.hpp lib:

liblua.a

man/man1: lua.1 luac.1

These are the only directories you need for development. If you only want to run Luaprograms, you only need the files in bin and man. The files in include and lib are needed forembedding Lua in C or C++ programs.

*Customization*

Three kinds of things can be customized by editing a file:

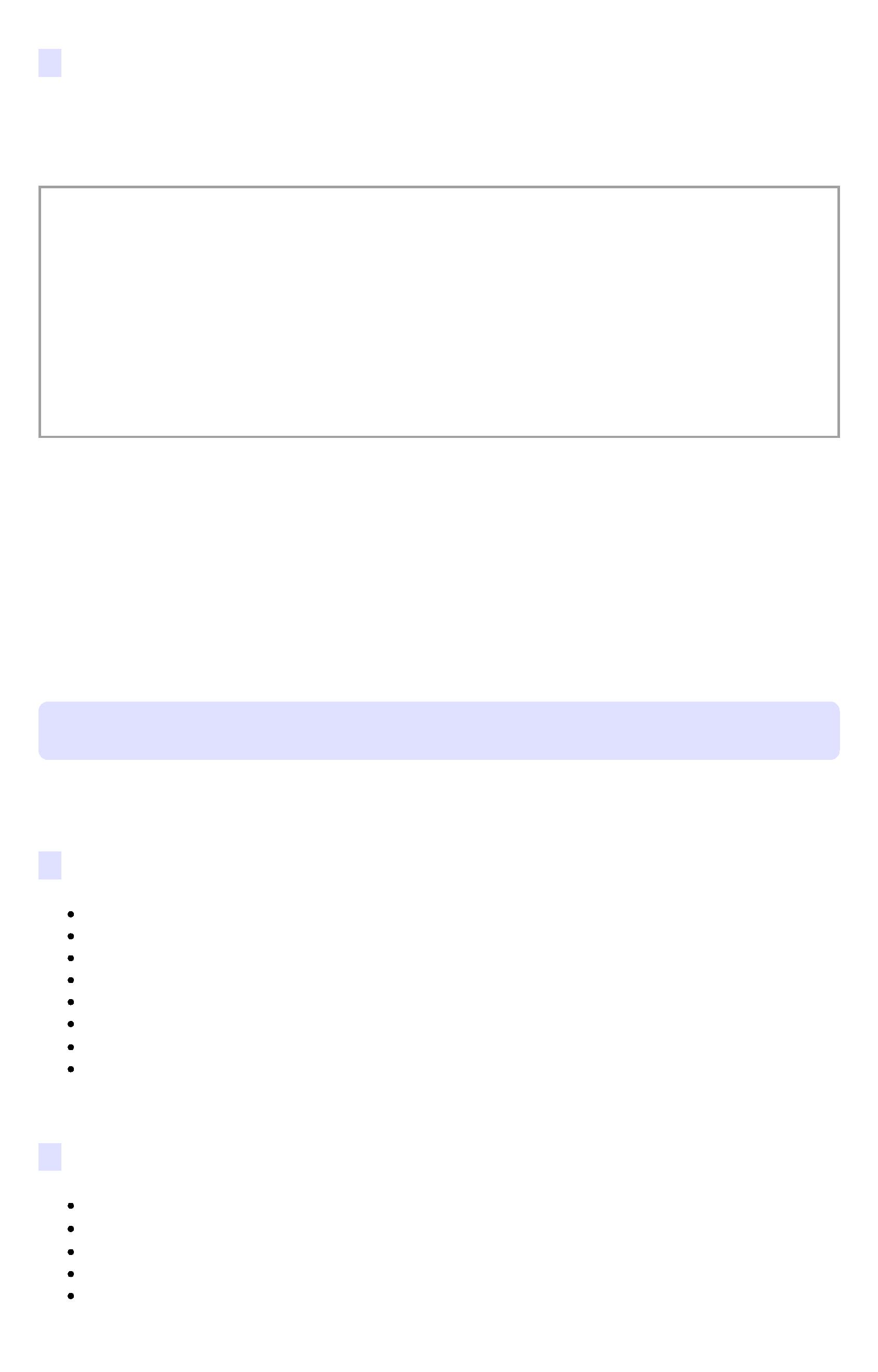
Where and how to install Lua — edit Makefile.How to build Lua — edit src/Makefile.Lua features — edit src/luaconf.h.

You don't actually need to edit the Makefiles because you may set the relevant variables inthe command line when invoking make. Nevertheless, it's probably best to edit and save theMakefiles to record the changes you need.

On the other hand, if you need to customize some Lua features, you'll need to edit src/luaconf.h before building and installing Lua. The edited file will be the one installed,and it will be used by any Lua clients that you build, to ensure consistency. Furthercustomization is available to experts by editing the Lua sources.

We strongly recommend that you enable dynamic loading in src/luaconf.h. This is done automatically for all platforms listed above that have this feature and also for Windows.

*Building Lua on other systems*



If you're not using the usual Unix tools, then the instructions for building Lua depend on thecompiler you use. You'll need to create projects (or whatever your compiler uses) for buildingthe library, the interpreter, and the compiler, as follows:

library:

lapi.c lcode.c lctype.c ldebug.c ldo.c ldump.c lfunc.c lgc.c llex.c lmem.clobject.c lopcodes.c lparser.c lstate.c lstring.c ltable.c ltm.c lundump.c lvm.clzio.c lauxlib.c lbaselib.c lbitlib.c lcorolib.c ldblib.c liolib.c lmathlib.c loslib.clstrlib.c ltablib.c loadlib.c linit.c

interpreter:

library, lua.c compiler:

library, luac.c

To use Lua as a library in your own programs you'll need to know how to create and uselibraries with your compiler. Moreover, to dynamically load C libraries for Lua you'll need toknow how to create dynamic libraries and you'll need to make sure that the Lua APIfunctions are accessible to those dynamic libraries — but *don't* link the Lua library into eachdynamic library. For Unix, we recommend that the Lua library be linked statically into the host program and its symbols exported for dynamic linking; src/Makefile does this for the Lua interpreter. For Windows, we recommend that the Lua library be a DLL.

As mentioned above, you may edit src/luaconf.h to customize some features before building Lua.

*Changes since Lua 5.1*

Here are the main changes introduced in Lua 5.2. The [reference manual](file:///C:/Cpp/skia/third_party/lua/doc/contents.html) lists the[incompatibilities](file:///C:/Cpp/skia/third_party/lua/doc/manual.html#8) that had to be introduced.

*Main changes*

yieldable pcall and metamethodsnew lexical scheme for globalsephemeron tables

new library for bitwise operationslight C functions

emergency garbage collector gotostatement finalizers for tables

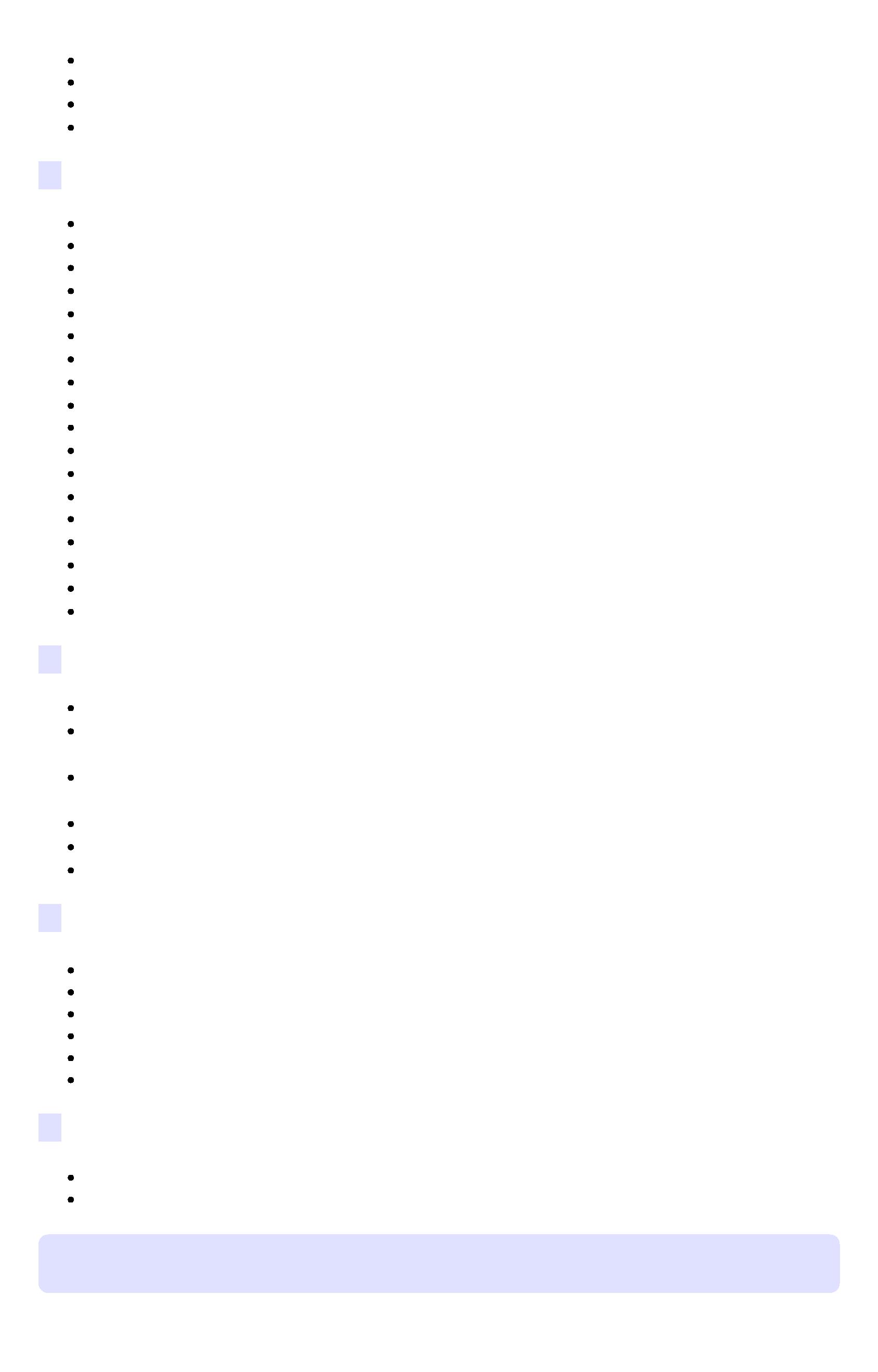
Here are the other changes introduced in Lua 5.2:

*Language*

no more fenv for threads or functions tables honor the \_\_lenmetamethodhex and \zescapes in strings support for hexadecimal floats

order metamethods work for different types

no more verification of opcode consistencyhook event "tail return" replaced by "tail call"empty statement



breakstatement may appear in the middle of a block

*Libraries*

arguments for function called through xpcall

optional 'mode' argument to load and loadfile (to control binary x text)optional 'env' argument to load and loadfile (environment for loaded chunk) loadlibmay load libraries with global names (RTLD\_GLOBAL)new function package.searchpath modules receive their paths when loaded

optional base in math.log optional separator in string.repfile:writereturns file closing a pipe returns exit status

os.exitmay close state

new metamethods \_\_pairsand \_\_ipairsnew option 'isrunning' for collectgarbageand lua\_gcfrontier patterns \0in patterns

new option \*Lfor io.readoptions for io.lines

debug.getlocalcan access function varargs

*C API*

main thread predefined in the registry

new functions lua\_absindex, lua\_arith, lua\_compare, lua\_copy, lua\_len,lua\_rawgetp, lua\_rawsetp, lua\_upvalueid, lua\_upvaluejoin, lua\_version.new functions luaL\_checkversion, luaL\_setmetatable, luaL\_testudata,luaL\_tolstring.

lua\_pushstringand pushlstringreturn stringnparamsand isvarargavailable in debug API new lua\_Unsigned

*Implementation*

max constants per function raised to 226generational mode for garbage collection (experimental)NaN trick (experimental)

internal (immutable) version of ctypes simpler implementation for string buffers

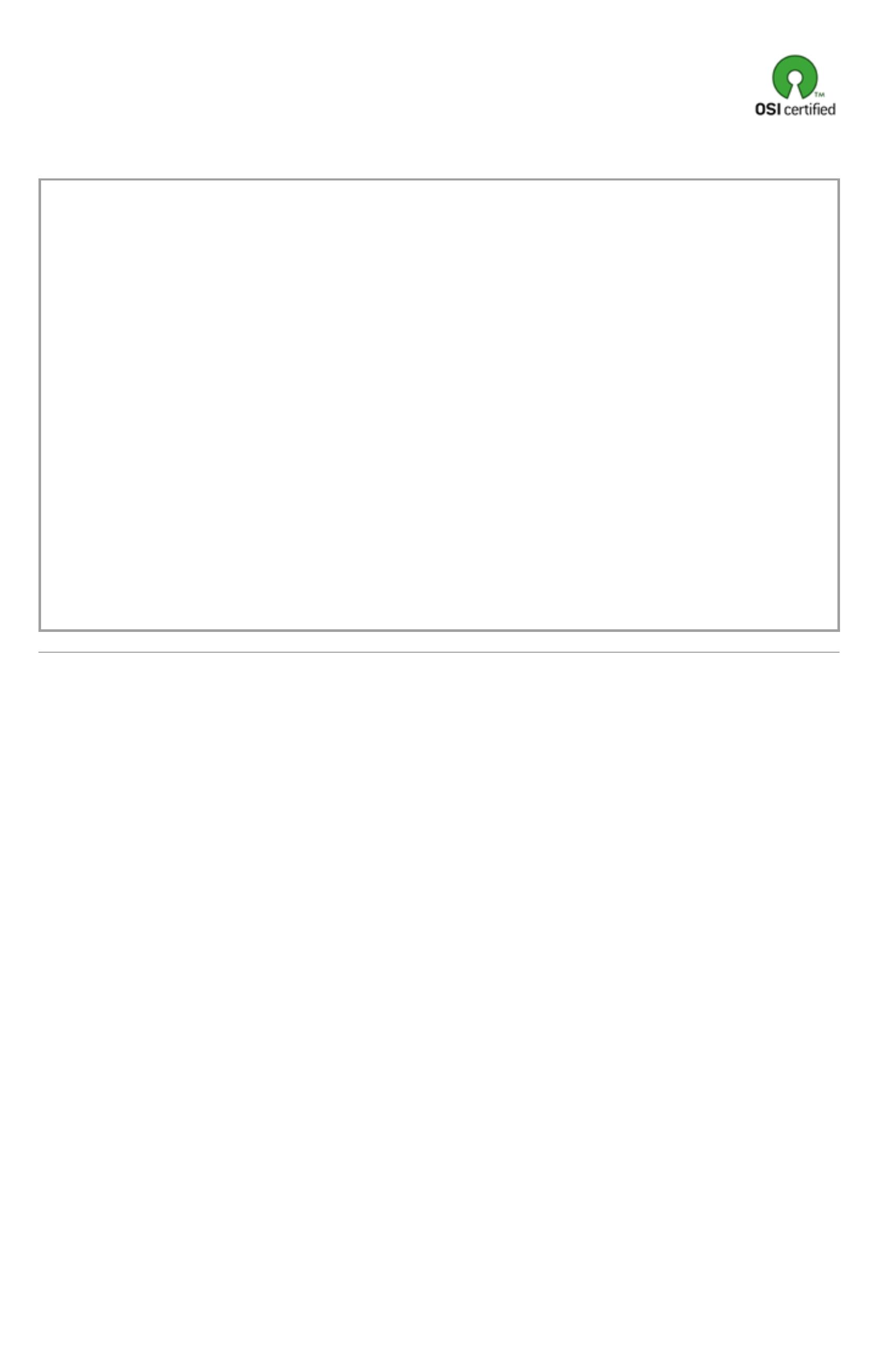
parser uses much less C-stack space (no more auto arrays)

*Lua standalone interpreter*

new -Eoption to avoid environment variables handling of non-string error messages

*License*

Lua is free software distributed under the terms of the [MIT license](http://www.opensource.org/licenses/mit-license.html) reproducedbelow; it may be used for any purpose, including commercial purposes, atabsolutely no cost without having to ask us. The only requirement is that ifyou do use Lua, then you should give us credit by including the appropriatecopyright notice somewhere in your product or its documentation. For details, see [this](http://www.lua.org/license.html).



Copyright © 1994–2013 Lua.org, PUC-Rio.

Permission is hereby granted, free of charge, to any person obtaining a copy of thissoftware and associated documentation files (the "Software"), to deal in the Softwarewithout restriction, including without limitation the rights to use, copy, modify, merge,publish, distribute, sublicense, and/or sell copies of the Software, and to permit personsto whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies orsubstantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OFMERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE ANDNONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHTHOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY,WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISINGFROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OROTHER DEALINGS IN THE SOFTWARE.

Last update: Fri Feb 22 09:24:20 BRT 2013