1 7 - Incompatibilities with the Previous Version

Here we list the incompatibilities that you may find when moving a program from Lua 5.0 to Lua 5.1. You can avoid most of the incompatibilities compiling Lua with appropriate options (see file `luaconf.h`). However, all these compatibility options will be removed in the next version of Lua.

1.1 7.1 - Changes in the Language

- The vararg system changed from the pseudo-argument `arg` with a table with the extra arguments to the vararg expression. (See compile-time option `LUA_COMPAT_VARARG` in `luaconf.h`.)

- There was a subtle change in the scope of the implicit variables of the **for** statement and for the **repeat** statement.

- The long string/long comment syntax (`[[string]]`) does not allow nesting. You can use the new syntax (`[=[string]=]`) in these cases. (See compile-time option `LUA_COMPAT_LSTR` in `luaconf.h`.)

1.2 7.2 - Changes in the Libraries

- Function `string.gfind` was renamed `string.gmatch`. (See compile-time option `LUA_COMPAT_GFIND` in `luaconf.h`.)

- When `string.gsub` is called with a function as its third argument, whenever this function returns **nil** or **false** the replacement string is the whole match, instead of the empty string.

- Function `table.setn` was deprecated. Function `table.getn` corresponds to the new length operator (`#`); use the operator instead of the function. (See compile-time option `LUA_COMPAT_GETN` in `luaconf.h`.)

- Function `loadlib` was renamed `package.loadlib`. (See compile-time option `LUA_COMPAT_LOADLIB` in `luaconf.h`.)

- Function `math.mod` was renamed `math.fmod`. (See compile-time option `LUA_COMPAT_MOD` in `luaconf.h`.)

- Functions `table.foreach` and `table.foreachi` are deprecated. You can use a for loop with `pairs` or `ipairs` instead.

- There were substantial changes in function require due to the new module system. However, the new behavior is mostly compatible with the old, but require gets the path from package.path instead of from LUA_PATH.

- Function collectgarbage has different arguments. Function gcinfo is deprecated; use collectgarbage("count") instead.

1.3 7.3 - Changes in the API

- The luaopen_* functions (to open libraries) cannot be called directly, like a regular C function. They must be called through Lua, like a Lua function.

- Function lua_open was replaced by lua_newstate to allow the user to set a memory-allocation function. You can use luaL_newstate from the standard library to create a state with a standard allocation function (based on realloc).

- Functions luaL_getn and luaL_setn (from the auxiliary library) are deprecated. Use lua_objlen instead of luaL_getn and nothing instead of luaL_setn.

- Function luaL_openlib was replaced by luaL_register.

- Function luaL_checkudata now throws an error when the given value is not a userdata of the expected type. (In Lua 5.0 it returned NULL.)