

Luciole v2

Pascal Raymond

2020-05-05

Contents

0.1	Differences with luciole v1	1
0.2	Components	1
0.3	Basic usage: <i>luciole</i> command line	2
0.3.1	<i>luciole</i> <file.dro>	2
0.3.2	<i>luciole</i> <file.ec>	2
0.3.3	<i>luciole</i> <file.lus>	2
0.3.4	<i>luciole</i>	2
0.4	Advanced usage	2
1	Default Reactive GUI	3
1.1	Overview	3
1.1.1	Menus	3
1.1.2	Panel	3
1.1.3	Widgets	3
1.2	Build-in features	3
1.3	Extra features: <i>luciolerc</i>	3
1.3.1	Real-time steps	3
1.3.2	<i>Sim2chro</i>	3
1.3.3	Customizing <i>luciolerc</i>	3
2	Lustubs library	3
3	Luciole package and Luciole scripts	3
4	Luciole script	3
5	Ptk package	3
6	Exemples and demos	3

. #Luciole v2 overview

Luciole v2 is a set of libraries and utilities for quickly build GUI's (Graphical User Interface) for reactive synchronous programs. It targets in particular Lustre programs, but can it can handle other reactive programs encapsulated into DRO libraries (Dynamic-Loaded Reactive Object).

0.1 Differences with *luciole* v1

This section quickly presents the similarities and differences with the previous version of *luciole*. If you are not familiar with the previous version, you can skip this part.

- The new top-level script *luciole* is intended to provide exactly the same features as the previous one. In particular it allows the execution of:
 - *dynamically-loaded reactive object* (.dro files), present since version 1.7,
 - *Lustre expended code* (.ec file), which is the historical supported format.
- Moreover, the script embeds *helpers* to build .ec or .dro (new) from lustre v4 and lustre v6 (new) programs.

Aside these similarities, the core of the tool has been completely remade. In particular, *luciole* v2 is almost entirely developed in pure tcltk, while v1 was mostly developed in c++. It makes the portability of the tool much simpler, since it only requires a standard install of tcltk.

- The executable *simec* no longer exists (but the *luciole* script provides the same functionality).
- The .iop (input/output panel) format is abandoned. It was used to customize gui layouts, and is replaced by a tcltk library (*luc*) that provides a similar programming style.

0.2 Components

Luciole is based on tcltk 8.6, and is made of several components:

- **lustubs.so** is a tcl-stub dynamic library that provides the necessary code for loading and running reactive programs. This is the only part which is machine dependent.
- **ptk** is a tcltk package for developing gui's in a more structured and abstract way than raw tcltk. It is developed in pure tcltk, and not specifically dedicated to *luciole*.
- **luc** is a tcltk package, based on *ptk*, that provides the bricks for building reactive gui's and link them to reactive program.
- **.luc files** are tcl scripts based on *luc* package. Each .luc file implements a particular gui for a particular reactive programs. They can be written by hand, or (transparently) generated by *luciole*.
- **luciole** is the top-level script, that builds all the necessary stuff to run a reactive gui according to the user arguments.

0.3 Basic usage: *luciole* command line

For a basic usage, the knowledge the *luciole* script is sufficient, in particular it does not require to manipulate a .luc script. The command-line arguments and the functionalities are the same as the previous *luciole* command version

0.3.1 *luciole* <file.dro>

- *luciole* foo.dro builds and run a default gui for running the reactive program foo.dro. The program inputs/outputs are implemented as graphical widgets (buttons, sliders, labels etc.), and displayed in a default 2-columns layout. The default

gui is similar to the one of the previous version of *luciole* (for those who are familiar with).

0.3.2 *luciole* <file.ec>

0.3.3 *luciole* <file.lus>

0.3.4 *luciole*

0.4 Advanced usage

For advanced usage, knowledge on *luciole* components is necessary. Some knowledge on tcl programming is indeed also necessary.

Each component is explained in details in the sequel. Here is a list of typical needs, together with the sections to look for:

- **Adding features to the default gui:** see *luciolerc.tcl*, in *default reactive gui*.
- **Customizing the default in/out layout:** some knowledge on the `luc` package is necessary, more precisely the *luc panel description* section.
- **Slightly/Completely modify the default gui:** if possible, you can partly rely the `luc` library; for really specific need you will have to program directly at the `ptk` level for the graphical part, and the `lustubs.so` level for executing reactive programs. In this case, a good level in tcl programming is mandatory.

1 Default Reactive GUI

1.1 Overview

1.1.1 Menus

1.1.2 Panel

1.1.3 Widgets

1.2 Build-in features

1.3 Extra features: luciolerc

1.3.1 Real-time steps

1.3.2 Sim2chro

1.3.3 Customizing luciolerc

2 Lustubs library

3 Luciole package and Luciole scripts

4 Luciole script

5 Ptk package

Ptk is a library to ease the programming of GUI's with tcltk. It is used by, but not specific to luciole.

See Ptk manual for details.

6 Exemples and demos