Development notes of the Guile-SFML bindings

Eduardo Acuña Yeomans

April 2014

About

This is a development documentation, transcribed from the notes I write for the planning and programming related to this proyect. The reader may encounter ideas, struggles, and references related to the development of the bindings.

This notes can be useful for me in the future and to other people that want to use a C library from GNU Guile. However my first language isn't english so any suggestions or corrections can be mailed to `eduardo.acye@gmail.com`.

Development plan

The goal of this proyect is constructing a clean and simple library for graphics and audio interactive programming.

I have C++ background but currently getting more involved with the Lisp family of languages, in school I had to write a simple game in C++ and chose the SFML library to handle the graphics, input and audio. I liked it so much that i wanted to use the library from a Scheme environment. The first thing i tried was to write the bindings using the static FFI using the Guile manual as a starting point, but it was a lot of not fun C and C++ programming (I have suspended the development of this bindings but the things i got to work live in my `https://github.com/eduardoacye/Guile-SFML-1` repository).

Now the intention is to write the bindings using the Dynamic FFI. The process for getting the proyect done is divided into three steps:

1. Mapping the CSFML structures and functions directly to Guile with the Dynamic FFI.
2. Wrapping the previous bindings to create modules that use GOOPS and Scheme idioms to work with the SFML implementation.
3. Expanding the previous modules with useful functions, macros and classes that are not part of SFML.

There isn't much documentation in the official Guile Reference Manual about writting bindings with the Dynamic Foreign Function Interface, however there are a couple of proyects of bindings for Guile using this method that I think are very helpful, in particular Andy Wingo's guile-figl repository and David Thompson guile-allegro5 and guile-2d repositories.

I recomend you check those out:

- guile-figl at `https://gitorious.org/guile-figl`
- guile-allegro5 at `https://github.com/davexunit/guile-allegro5`
- guile-2d at `https://github.com/davexunit/guile-2d`

1. Raw bindings from CSFML

This is the first step of the development process. And it consist of getting all the SFML structures and functions working from Guile. I'm using the C bindings as a starting point. To start working i had to compile Guile (currently using the 2.0.9 version), SFML 2.1 and CSFML 2.1.

The structure of the bindings is very similar to the directory structure of the CSFML source code. There is one module for each file in CSFML, however there might be some files missing that provide C++ and C functionalities that Guile already have (such as multithreading).

guile-sfml directory tree

- **sfml2**
  - **audio**
    - `todo`
  - **graphics**
    - `todo`
  - **network**
    - `todo`
  - **system**
    - clock.scm
    - common.scm
    - sleep.scm
    - time.scm
    - vector2.scm
    - vector3.scm
  - system.scm
  - utils.scm
  - **window**
    - `todo`

Common code

The code that has to be aviable from every sub-library, like auxiliary procedures and useful macros are in the `sfml2/utils.scm` file. The code that has to be aviable from every module in a specific sub-library, like the macro for defining a foreign function of that sub-library is in the `sfml2/sub-lib/common.scm` file, where `sub-lib` can be: `audio`, `graphics`, `network`, `system` or `window`.

1.1. SFML/System

1.2. SFML/Window

1.3. SFML/Graphics

1.4. SFML/Audio

1.5. SFML/Network

1.6. Testing bindings: SFML 2.1 tutorials

Basic application

Graphics tutorial

Autio tutorial

2. SFML adapted for Guile

2.1. SFML/System

2.2. SFML/Window

2.3. SFML/Graphics

2.4. SFML/Audio

2.5. SFML/Network

2.6. Testing library: Programs constructed with SFML

CodeFrame port

Keyboard piano

3. Automatón