barracuda manual

https://github.com/robitex/barracuda

Roberto Giacomelli
email: giaconet.mailbox@gmail.com

2019-11-28
Version v0.0.9

Abstract

Welcome to the barracuda software project devoted to barcodes printing.

This manual shows you how to print barcodes in your TeX documents or how to export such graphic content to an external file, using barracuda.

barracuda is written in Lua programming language and is free software released under the GPL 2 License.

1 Introduction

Barcode symbols are usually a sequence of vertical lines representing encoded data that can be retrived with special laser scanner or more simpler with a smartphone running dedicated apps. Almost every store item has a label with a printed barcode for automatic identification purpose.

So far, barracuda supported symbologies are as the following:

- Code 39,
- Code 128,
- EAN family (EAN 8, EAN 13, and the add-ons EAN 2 and EAN 5),
- ITF 2of5, interleaved Two of Five.

The name barracuda is an assonance with the name Barcode. I started the project back in 2016 for getting barcode in my TeX generated PDF documents, studying the LuaTeX technology.

1.1 Manual Content

The manual is divided into three parts. At section 2 the first look gives the user a proof of concept to how to use and how works the package while the next parts present details like how to change the *module* width of a EAN-13 barcode or how to implement a barcode symbology not already included in the package.

The section referenced plan of the manual is (but some sections are not completed yet):

Part 1: Get started

1.2 Required knowledge and useful resources

The barracuda is a Lua package that can be executed by any Lua interpreter. To use it, it's necessary some knowledge of Lua programming language and a certain ability with the terminal of your computer system in order to accomplish command tasks or software installations.

It's also possible to run barracuda directly from within a TeX source file, compiled with a suitable typesetting engine like LuaTeX. To do so a minimal TeX system knowledge is required. As an example of this workflow you simply can look to this manual because itself is typesetted with LuaLaTeX, running barracuda to include barcodes as a vector graphic object.

Here is a collection of useful learning resources...

2 Get Started with Barracuda

The starting point to work with barracuda is always a plain text file with some code, late processed by a command line program with a Lua interpreter.

As a practical example producing an EAN-13 barcode, in a text editor of your choice on a empty file called first-run.lua, type the following two lines of code:

first-run.lua

```
local barracuda = require "barracuda"
barracuda:save("ean-13", "8006194056290", "my_barcode", "svg")
```

What you have done is to write a *script.* If you have installed a Lua interpreter and barracuda, open a terminal and run the command:

```
$ lua first-run.lua
```

You will see in the same directory of your script, appearing the new file my_barcode.svg with the drawing:

Coming back to the script first-run.lua, first of all, it's necessary to load the library with the standard statement require(). What that Lua function returns is an object–more precisely a table reference–where are stored every package features.

We can now produce the EAN-13 barcode using the method save() of the barracuda object. The save() method takes in order the barcode symbology identifier, the data to be encoded as a string or also a whole number, the output file name and the optional output format.

2.1 Running LuaTEX

Barracuda can also running inside LuaTEX and the others Lua powered TEX engine. The text source file is a bit difference respect to a Lua script: Lua code have to bring place as the argument of directlua primitive... we must use a box register of type horizontal...

```
% !TeX program = LuaTeX
\nopagenumbers
\newbox\mybox
\directlua{
    local require "barracuda"
    barracuda:hbox("ean-13", "8006194056290", "mybox")
}\box\mybox
\bye
```

The method hbox() works only with LuaTEX.

2.2 A more deep look

Barracuda is designed to be modular and flexible. For example it is possible to draw different barcodes on the same canvas or tuning barcode parameters.

The code becomes more structured. structure of the user code. In fact, to draw an EAN-13 barcode we must do at least the follow steps:

1. get a reference to the Barcode class,
2. build an EAN-13 encoder,
3. build a EAN symbol passing data to a costructor,
4. get a reference to a canvas object,
5. draw barcode on canvas,
6. get a reference of driver object,
7. address canvas toward a driver.

```
local barracuda = require "barracuda"
local barcode = barracuda:get_barcode_class()

local ean13, err_enc = barcode:new_encoder("ean-13")
assert(ean13, err_enc)

local symb, err_symb = ean13:from_string("8006194056290")
assert(symb, err_symb)

local canvas = barracuda:new_canvas()
```

```
symb:append_ga(canvas)

local driver = barracuda:get_driver()
local ok, err_out = driver:save("svg", canvas, "my_barcode", "svg")
assert(ok, err_out)
```

3 Installing

The esier way to install on your system Barracuda, is via TeX Live tlcontrib, the home of among the others, experimental package.

4 Barracuda LaTeX Package

TODO

5 The Barracuda Framework

The barracuda package framework consists in indipendet modules: a barcode class hierarchy encoding a text into a barcode symbology; a geometrical library called libgeo representing several graphic object; an encoding library for the ga format (graphic assembler) several driver to ”print” a ga stream into a file or a TeX hbox register.

To implement a barcode encoder you need to write a component called *encoder* defining every parameters and producing the encoder class, while a driver must understand ga opcode stream and print the corresponding graphic object.

Every barcode encoder come with a set of parameters, some of them can be reserved and can be setting up by the user only through the encoder.

So, you can create many instances of the same encoder for a single barcode type, with its own parameter set.

The basic idea is getting faster encoder, for which the user may set up paramenters at any level: barcode abstract class, encoder, down to a single symbol.

Barcode class is completely indipendent from the ouput driver and viceversa.

6 Example and use cases

TODO

7 Barcode Reference

TODO

8 API reference

TODO