basics

Contents

1 Introduction

Not long after we released LuaTEX 1.0, we started experimenting a bit more with so called foreign function interface: ffi. Originally that interface to external libraries was only available in LuajitTEX, but a good and compatible alternative is now also available in the normal engine too. For users it is not that relevant to know how it works, as long as it works. It means that in addition to SwigLib we have a method that doesn't demand compilation as it uses normal (public) libraries.

Of course one needs to make sure that the right version of a library is used. And, as there is the danger of the api having been changed in an incompatible way one can wonder if such a dependency is really what one wants. On the other hand one can expect ConTEXt to keep up.

Do you really need libraries? For instance does it really make sense to use curl, ghostscript or graphicmagic libraries while the command line version is (normally) just as efficient and avoids a dependency. This is even more true if you realizes that for instance a fetch or conversion only needs to happen once per run or in fact only when there is some change in the resource.

On the other hand, when accessing databases one can avoid the often slower command line calls and save the hassle of intermediate files. Here efficiency wins. Also, when ConTEXt is used in a high performance database backend application a distribution and the used libraries are not updated on a daily basis.

2 Supported

Apart from some experiments we currently can use ffi interfaced libraries in:

module	**library**	**windows**	**unix**
util-crl	curl	libcurl	libcurl
util-sql-imp-ffi	mysql	libmysql	libmysqlclient
util-sql-imp-sqlite	sqlite	sqlite3	sqlite3

The profiler that we occasionally use to identify bottlenecks in the engine (for instance when we upgrade Lua) uses ffi to provide access to the high resolution timers but this is typically different per platform.

One problem with libraries, especially on Windows is that the library is found on some system path and it can happen that multiple programs ship the same library but in different versions.

You can try to play safe and put libraries in the TEX tree, for instance on my system they are in:

```
c:/data/tex-context/tex/texmf-win64/bin/lib/luatex/lua/whatever/libwhatever.dll
```

You can trace where libraries are looked for with:

```
\enabletrackers[resolvers.ffilib]
```

or in Lua with:

```
trackers.enable("resolvers.ffilib")
```

The library is first located on one of the valid tds paths (these are sort of standardized in TEX distributions) and then using the normal ffi loader.

As this is all still experimental in LuaTEX there is not much more to say about it now. Of course this kind of specialized support to a large degree depends on the need to use it.

3 Colofon

author	Hans Hagen, PRAGMA ADE, Hasselt NL
version	December 4, 2017
website	www.pragma-ade.nl – www.contextgarden.net
comment	many thanks to Luigi Scarso for taking care of ffi support in the engines