



Tk Mega-widgets

Charles Colbourn
charles.colbourn@googlemail.com

These days, an application intended for end users pretty much has to have a graphical user interface (GUI). In spite of a couple of decades of command line use, even I'm finding that it's easier for me to open a GUI window and click around a well-designed user interface (UI) than remember lists of switches and parameters.

There are a number of different GUI libraries available to the Perl programmer: Tk, Wx, Qt, GTK+, and perhaps some others that I've missed. Of the various alternatives, I prefer the conceptual design of Wx. It relies on defining your own objects that extend existing objects in the Wx class hierarchy and inherit the necessary GUI behavior, rather like AWT in Java. It also has a handy GUI builder available. This means that with little effort I can produce neatly encapsulated code that's easy to maintain.

WxWidgets

Wx has two main disadvantages. First, wxperl has relatively little documentation, relying instead on the documentation for the C++ API. Once I get used to adapting what I read in the C++ documentation to Perl, this doesn't present a particularly great problem, but it's a little off-putting to try perldoc Wx and not be able to find what I'm looking for.

Second, and far more problematic for me is the module dependency chain. Wxperl needs Wx, Alien::WxWidgets, and GTK+; each of which has its own list of dependencies. In addition, some of these dependencies are version specific.

If I'm running on a Linux platform, getting the right combination of versions of the various dependencies without breaking anything else can take a lot of tinkering. On at least one occasion I've given up after two days of trying with no success.

Installing on Microsoft Windows is equally patchy; some combinations of libraries and versions will just drop together perfectly, others will break horribly.

If I'm writing an application for end users, once I've got Wx installed on my development machine, and I've written the code, I then have to get Wx to install on the end users' machines too.

Tk

Tk, on the other hand, is very, very stable and commonplace. It comes with ActiveState's ActivePerl, and even if I compiled my own Perl or installed with apt-get, Tk seems to build pretty consistently without major hiccups. It also has a very extensive widget set, and is well documented, well-supported and more familiar to most Perl developers.

For me, the ideal situation would be to be able to get the neat encapsulation of Wx with the extensive widget set and near-universal availability of Tk, and a nice GUI builder as a bonus. Using ZooZ, courtesy of Ala Qumsieh, and a conversation with Ala some months ago, I think I've worked out a way to get all three.

Tk mega-widgets are intended to be used to create reusable widget combinations; if I often use a server name and status light for example, I might want a mega-widget made up of a Label and Statusbox. Another composite widget that often comes in handy is a combined Entry & File Browse button.

I quite often create my mega-widgets in the ZooZ GUI builder, export them as a .pm file, and do some hacking on it afterward. For anyone who hasn't encountered ZooZ, it's a Tk GUI builder written in pure Perl/Tk, and runs happily in every Perl 5.8+ environment. The output is either a .pl script, or a .pm mega-widget, and a file containing the details of the project I've created so I can reload it and work on it again. Figure 1 shows ZooZ in the process of creating a small application.

The ZooZ workspace is on the left, and the previewed application GUI is on the right. I can select widgets in the tool space on the far left and place them on the