Subject: Re: Graduation thesis - Camouflage - a replacement of(a part of) Chameleon
Posted by andrei_natanael on Sat, 13 Feb 2010 19:30:02 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Sat, 13 February 2010 20:19Hi,

Please, never add any dependencies to any SOs/DLLs in packages belonging to uppsrc. In other 'optional' nests, I don't care, but just keep the uppsrc clean of them.

I seriously think the basic GUI framework should be kept as simple and robust as possible. Ultimate++ has become a vitally important tool for me to create commercial software products and anything that makes things more complex, usually causes loads of customer support that can't be charged for.

I can understand that 'eye candy' is needed for certain segment of software products, but simplicity and reliability is more important for me and my clients, so I suggest the 'eye candy' packages come in only as an option for those interested -- not by default.

Kind regards,

Tom

Hello Tom,
So the solution would be one exe for KDE (if I'll implement KDE look using KDE libs), one for GTK+, because right now gtk+ is statically linked in U++ and one exe if i don't care about platform integration (X11 only with flagNOGTK). I know it may looks more robust but I'll have to provide 3 different executables for (i.e.) same client using Gnome, KDE or other desktop managers sometime. My solution was to avoid statical linkage and provide only an exe which would work in all 3 cases. If user is running Gnome then it will use gtk+ (indirectly through camo-gtk.so), in case that the same user is signing off from Gnome and start his KDE desktop environment then the same exe will get the look for system using camo-qt.so. If the user isn't using a D.M. at all or is using a DM which doesn't use either gtk+ or qt then the application will use U++ look. Existence of gtk or qt will not be a problem for user like it is now (with gtk static link). Is not a problem if my program doesn't find at runtime one of those two libs (camo-gtk/qt.so) because it will use U++ look but now if we take an U++ program which need gtk+ and move the exe in KDE it will fail to start if gtk libs are not installed. I see the robustness more in my cases than in currently U++ state which force the developer to specify which libraries will be available on system.
Perhaps an pseudo-code of implementation will make my design approach more clear;

```
if (CurentRunningDesktopManagerIsGnome) {
  ChHostSkin will make use of camo-gtk.so
} else {
  if (CurrentRunningDesktopManagerIsKDE) {
    ChHostSkin will make use of camo-qt.so
  } else {
    ChHostSkin will use U++ style (ChStdSkin)
    // falling back to U++ style if qt/gtk aren't installed
```

```
  }
}
```
// BTW, this kind of check is running once at application start-up so there will be no penalties using an "external" so/dll
// And because camo-qt and camo-gtk share the same interface ChHostStyle of both would look identical, in fact it will be only one ChHostSkin implementation

camo-gtk/qt.so are required for portability, if there is another possibility to make use of gtk and qt in the same application without using an "external" so/dll then let me know. IMO it's not possible that because you have to link with qt at compile time because qt lib are C++ libs and you cannot use dlopen/dlsym with them.

Andrei
