

# Améliorer la portabilité & intégration d'applications dans Haiku

Qt 4 & 5, FLTK...

François Revol  
[revol@free.fr](mailto:revol@free.fr)

Capitole du Libre 2016

# Haiku ?

- Système d'exploitation libre
- Inspiré de BeOS
- Pas mal de POSIX
  - Mais on ne prétend pas être Unix®
- Des trucs sympa
  - Attributs étendus typés et indexables (BFS)


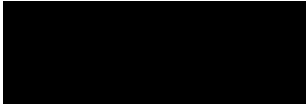









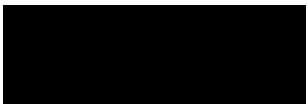

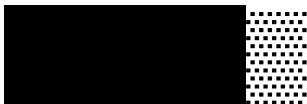





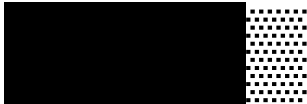



# Interface graphique native

- InterfaceKit
  - API C++ héritée de BeOS (similaire à Qt, + simple)
  - Classe BMessage (conteneur générique)
- Multithread
  - 1 thread principal (BApplication)
  - 1 thread par fenêtre (BLooper → BWindow)
- Quelques bindings existants
  - Yab (yabasic), FreePascal

# Pourquoi un Toolkit portable ?

- Conf. *Panorama des toolkits graphiques portés sous Haiku*
  - Prévue pour CdL 2015 → C'était le Bazar...
  - Présentée au [FOSDEM 2016](#)
- ✓ + d'applis, - de taf ?
  - ✓ + d'utilisateurs potentiels
  - ✓ + de devs ?
- ✓ Peut être bien fait
- ✗ Jamais vraiment natif
  - ✗ Pas de scripting
    - ✗ hey Foo get Frame of View...
    - ✗ Lecteur d'écran...
  - ✗ - de motivation / applis natives

# On en est où ?

 ncurses		 FLTK	
• <b>hdialog</b>		 LibreOffice	
 SDL 1.2		 Lazarus	
• SDL 2		 AWT, Swing	
 IUP		• GTK+	
 wxWidget		• Qt 4	
 Tk		• Qt 5	

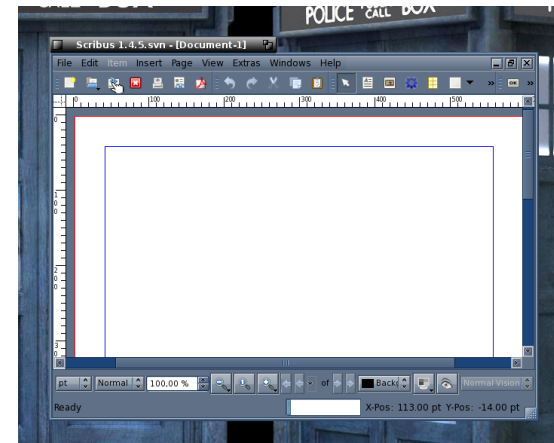
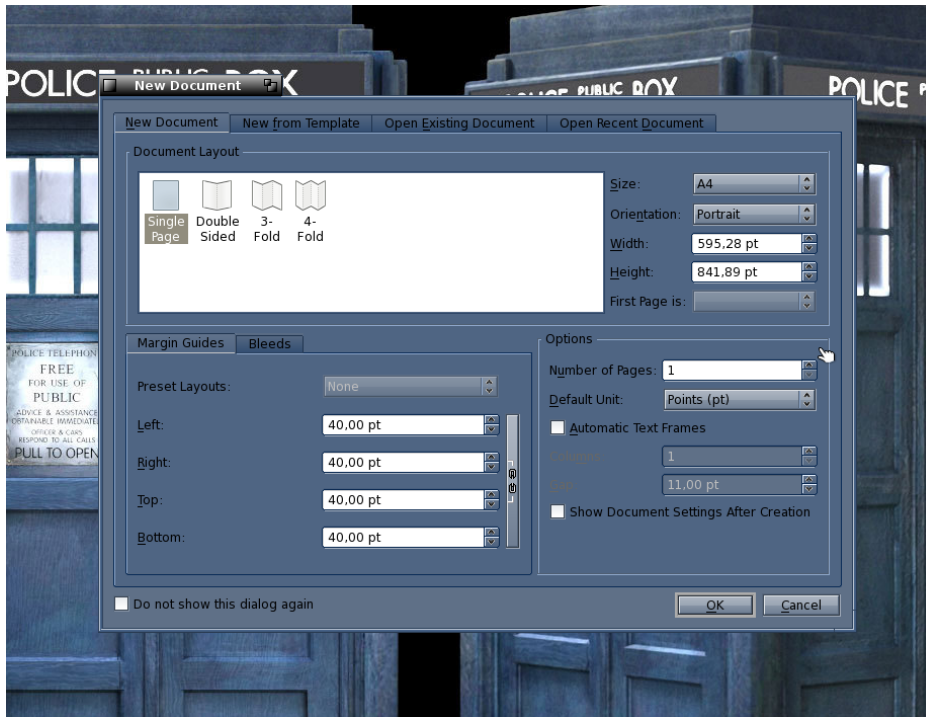
# Exemple : Gambas

- Basic pour \*nix
- Nombreux modules (GTK1&2 ; Qt4&5...)
- Qt\* dépend de... X11 ☹

```
checking for X... (cached) no
no
configure: WARNING: gb.qt5 is disabled
no
configure: WARNING: Unable to met pkg-config requirement: x11
configure: WARNING: gb.qt5.webkit is disabled
no
configure: WARNING: Unable to met pkg-config requirement: gl
configure: WARNING: Unable to met pkg-config requirement: x11
configure: WARNING: gb.qt5.opengl is disabled
no
configure: WARNING: Unable to met pkg-config requirement: x11
configure: WARNING: gb.qt5.ext is disabled
checking that generated files are newer than configure... done
configure: creating ./config.status
```

```
[user@gb /work/gambas/trunk/gb.q5]# grep x11 -R src/
src/CDialog.cpp: dpiX = QPaintDevice::x11AppDpiX();
src/CDialog.cpp: dpiY = QPaintDevice::x11AppDpiY();
src/CDialog.cpp: QPaintDevice::x11SetAppDpiX(CFONTP_dpi);
src/CDialog.cpp: QPaintDevice::x11SetAppDpiY(CFONTP_dpi);
src/CDialog.cpp: QPaintDevice::x11SetAppDpiX(dpiX);
src/CDialog.cpp: QPaintDevice::x11SetAppDpiY(dpiY);
src/CDrawingArea.cpp: QX1Info xinfo = x11Info();
src/CFont.cpp: return size * (double)QPaintDevice::x11AppDpiY() / CFONTP_dpi;
src/CFont.cpp: return size * CFONTP_dpi / (double)QPaintDevice::x11AppDpiY();
src/CMouse.cpp: Display* dpy = QPaintDevice::x11AppDisplay();
src/CScreen.cpp:#include "x11.h"
src/CStyle.cpp:#include "x11.h"
src/OWidget.cpp:static QMap<int, int> _x11_to_qt_keycode;
src/OWidget.cpp:    _x11_to_qt_keycode.insert(MAIN_x11_last_key_code, CKEY_info_code);
src/OWidget.cpp:    if (_x11_to_qt_keycode.contains(MAIN_x11_last_key_code))
src/OWidget.cpp:        CKEY_info_code = _x11_to_qt_keycode[MAIN_x11_last_key_code];
src/OWidget.cpp:    _x11_to_qt_keycode.remove(MAIN_x11_last_key_code);
src/OWindow.cpp:#include "x11.h"
src/Makefile.am:gb_qt5_la_OBJECTS = gb_qt5_la-x11.lo gb_qt5_la-desktop.lo \
src/Makefile: x11.h x11.c \
src/Makefile:include ./$(DEPIR)/gb_qt5_la-x11.Plo
src/Makefile: $(AM_V_CC)$(LIBTOOL) $(AM_V_lt) --tag=CC $(AM_LIBTOOLFLAGS) $(LIBTOOLFLAGS) --mode=compile $(CC) $(DEFS) $(DEFAULT_INCLUDES) $
11.lo -MD -MP -MF $(DEPIR)/gb_qt5_la-x11.Tpo -c -o gb_qt5_la-x11.lo 'test -f 'x11.c' || echo '$(srcdir)/' 'x11.c
src/Makefile: $(AM_V_at)$(am_mv) $(DEPIR)/gb_qt5_la-x11.Tpo $(DEPIR)/gb_qt5_la-x11.Plo
src/Makefile: $(AM_V_CC)source'x11.c' object='gb_qt5_la-x11.lo' libtool=yes \
src/Makefile: $(AM_V_CC_no)$(LIBTOOL) $(AM_V_lt) --tag=CC $(AM_LIBTOOLFLAGS) $(LIBTOOLFLAGS) --mode=compile $(CC) $(DEFS) $(DEFAULT_INCLUDES
_la-x11.lo 'test -f 'x11.c' || echo '$(srcdir)/' 'x11.c
src/Makefile.am: x11.h x11.c \
src/Makefile.in:an_gb_qt5_la_OBJECTS = gb_qt5_la-x11.lo gb_qt5_la-desktop.lo \
src/Makefile.in: x11.h x11.c \
src/Makefile.in:@AMDEP_TRUE@am__include@ am__quote./$(DEPIR)/gb_qt5_la-x11.Plo@am__quote@
src/Makefile.in:gb_qt5_la-x11.lo: x11.c
src/Makefile.in: @am__fastdepCC_TRUE@ $(AM_V_CC)$(LIBTOOL) $(AM_V_lt) --tag=CC $(AM_LIBTOOLFLAGS) $(LIBTOOLFLAGS) --mode=compile $(CC) $(DEF
(CFLAGS) -MT gb_qt5_la-x11.lo -MD -MP -MF $(DEPIR)/gb_qt5_la-x11.Tpo -c -o gb_qt5_la-x11.lo 'test -f 'x11.c' || echo '$(srcdir)/' 'x11.c
src/Makefile.in: @am__fastdepCC_TRUE@ $(AM_V_at)$(am_mv) $(DEPIR)/gb_qt5_la-x11.Tpo $(DEPIR)/gb_qt5_la-x11.Plo
src/Makefile.in: @AMDEP_TRUE@am__fastdepCC_FALSE@ $(AM_V_CC)source'x11.c' object='gb_qt5_la-x11.lo' libtool=yes @AMDEPBACKSLASH@
src/Makefile.in: @am__fastdepCC_FALSE@ $(AM_V_CC@am__nodep@)$(LIBTOOL) $(AM_V_lt) --tag=CC $(AM_LIBTOOLFLAGS) $(LIBTOOLFLAGS) --mode=compile :
_M_CFLAGS) -c -o gb_qt5_la-x11.lo 'test -f 'x11.c' || echo '$(srcdir)/' 'x11.c
src/main.cpp:#include "x11.h"
src/main.cpp: int MAIN_x11_last_key_code = 0;
src/main.cpp: static void (*_x11_event_filter)(XEvent *) = 0;
src/main.cpp: static void x11_set_event_filter(void (*filter)(XEvent *))
src/main.cpp: {
src/main.cpp:     _x11_event_filter = filter;
src/main.cpp:     MAIN_x11_last_key_code = ((xcb_key_press_event_t *)ev)->detail;
src/main.cpp: }
src/main.cpp: if (_x11_event_filter)
src/main.cpp:     (*_x11_event_filter)(&ev);
src/main.cpp: bool MyApplication::x11EventFilter(XEvent *)
src/main.cpp: {
src/main.cpp:     MAIN_x11_last_key_code = e->xkey.keycode;
src/main.cpp:     MAIN_x11_last_key_code = e->xkey.keycode;
src/main.cpp: if (_x11_event_filter)
src/main.cpp:     (*_x11_event_filter)(e);
src/main.cpp: //extern void qt_x11_set_global_double_buffer(bool);
src/main.cpp: //qt_x11_set_global_double_buffer(false);
src/main.cpp: *value = (void *)x11_set_event_filter;
src/main.h: extern int MAIN_x11_last_key_code;
src/main.h: virtual bool x11EventFilter(XEvent *e);
src/x11.c: x11.c
src/x11.c:#include "x11.h"
src/x11.h: x11.h
```

# Exemple : Scribus (Qt 4)



- Couleurs système 😊
- Impression : PDF 😞

# Exemple : MuseScore (Qt 5)

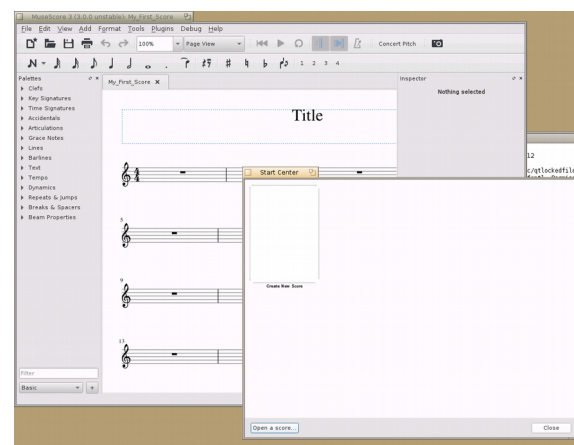
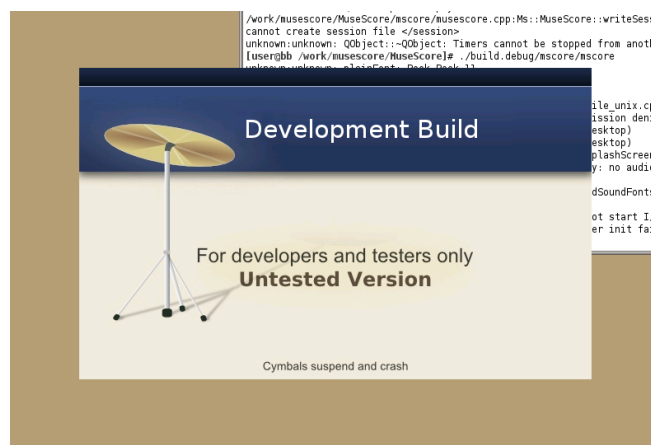
- Du **Make** généré par du **CMake** généré par du **qmake**...
- Du hardcoding
  - else (MINGW)

```
target_link_libraries(mscore
    ${ALSA_LIB} ${QT_LIBRARIES}
    mscore_freetype z dl pthread)
```
  - # 'gold' does not use indirect shared libraries for symbol resolution, **Linux only**

```
if (NOT APPLE)
    if(USE_JACK)
        target_link_libraries(mscore dl)
    endif(USE_JACK)
    target_link_libraries(mscore rt)
endif (NOT APPLE)
```



# Mais on y arrive !



- Look presque natif
  - mais pas d'intégration des couleurs
    - Utilisation de l'API `ui_color()` dans le thème Qt5 ?

# Exemple : FreeMedForms (Qt5)

- **FreeHealth** : fork utilisant Qt 5.6+QtWebEngine
  - Plus récent mais...
  - QtWebEngine dépend de Chromium
    - Un monstre qui dépend d'un monstre qui...

# Exemple : OpenToonz

- Logiciel d'animation libéré
- Utilisé par le studio Ghibli
- **Porté sur GNU/Linux** en moins de 2 semaines
- `tfont_{mac,nt}.cpp` : bypass de Qt
- → `tfont_qt.cpp`
  - `QRawFont`
  - Même pas besoin d'accéder à FreeType
- Portage en cours sur Haiku

# Qt 4

- Assez mature
- qsysstray (compilé avec gcc2) pour l'intégration

# Qt 5

- Depuis Qt 5 : **Qt Platform Abstraction** \o/
  - → QPA Haiku
  - SVP Passez par **QplatformNativeInterface**
    - Et séparez le code. Pensez à Wayland aussi ☺
- Encore des .cmake à corriger
  - Libs statiques dans `develop/lib/x86/`
- QtWebEngine ???

# FLTK

- `fl_open_file(callback)`
  - Ouvrir un fichier avec l'instance courante
  - À l'origine pour OSX
  - ~~`#ifdef __APPLE__`~~
- Ajout de ressources
  - À l'origine pour OSX
    - ~~`$(OSX_ONLY)`~~ `../fltk-config --post $@`

# DBus

- Et si on a pas DBus ?
- BMessage prédate
- Écrire un wrapper ?

# Systeme de fichier

- Haiku « à peu près » \*nix mais
  - `/system/develop/lib[/x86]/libfoo.a`
  - `/system/develop/lib[/x86]/libfoo.so → /system/lib...`
  - `/system/develop/headers/.../foo.h`
  - ~~`~/config/`~~ `~/config/settings`
  - ~~`~/local/`~~ `~/config/data ...`
- SVP pas de `install toto.so $prefix/lib` ☹
  - `autoconf` gère ça très bien



# inotify...

- Pour rappel, c'est la 3(4?)ème API pour Linux...
- Le node monitoring BeOS le fait depuis 15 ans
- Et puis y a [QFileSystemWatcher](#) !
  - Mais pas encore implémenté pour Haiku on dirait...

# « The Platform Problem »

- [LWN, 2011](#)
- On considère que la plateforme est intouchable
  - Donc on la contourne
  - Au lieu d'envoyer un patch !
- D'autant plus vrai sur un OS comme  
{apt,rpm...}+{Gnome,KDE...}+{GTK+,Qt}+...  
+GNU/Linux
  - Projets (management, butrackers...) différents

# FreeDesktop.org

- Discussion sur les bureaux « X11 » / \*nix
- Standardisation^WSpécifications
- Haiku n'est pas \*nix et n'utilise pas X11
  - Mais on fait des efforts

# xdg-utils

- Outils en ligne de commande
  - `xdg-open <URI|path>`
  - `xdg-email`
  - `xdg-mime`
  - ...
- Début de portage sur Haiku
  - Mapping BdD MIME ?
  - Extraction des `.desktop` ?

# Icônes



- Haiku : [HVIF](#)
  - Format vectoriel optimisé
  - Possible conversion de/vers SVG
- Heureusement, Inkscape sait assez bien vectoriser...
- Inclure la source SVG dans le dépôt, merci !

# Y a une Qclass pour ça !

- QSerialPort ([apt:libqt5serialport5-dev](#))
- QX11\* ([apt:libqt5x11extras5-dev](#))
  - Mais en option SVP !
- QSvg ([apt:libqt5svg5-dev](#))
- QSensor ([apt:libqt5sensors5-dev](#))
- XDG-utils ([apt:libqt5xdg-dev](#)) (merci LXQt)
- XDG-IconLoader ([apt:libqt5xdgiconloader-dev](#))

# Conclusion + Questions

- Moins de hardcodage
- Plus de codage (mais smart 😊)
- Y a pas un truc déjà pour ça ?
- C'est possible sans l'option ?