User Interface Programming
Libraries for GUI programming

Jozef Mlích

-Department of Computer Graphics and Multimedia
Faculty of Information Technology, Brno University of Technology
Božetěchova 2, 612 66 Brno, Czech Republic
http://www.fit.vutbr.cz/~imlich
imlich@fit.vutbr.cz

26.11.2009

Agenda

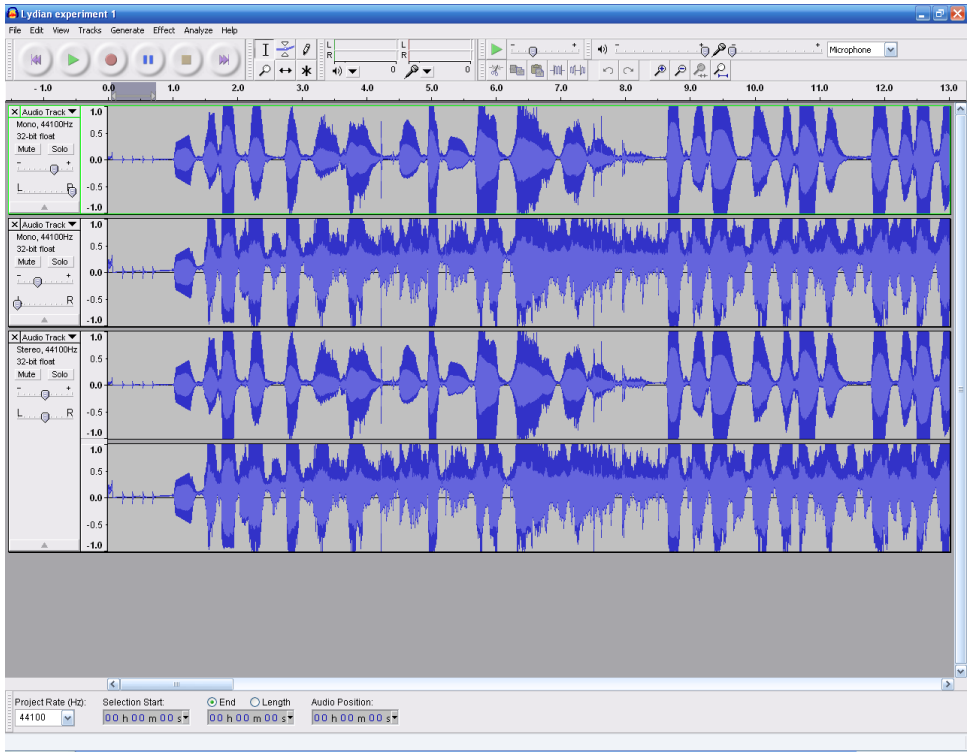
- wxWidgets
- GTK
- Qt

wxWidgets – features

- Cross-platform ability
- Open source
- Documentation and samples
- Wide range of control and window classes
- Powerful event system
- Printing facilities
- Debugging facilities
- Compiler support
- Database functionality
- Document/view classes
- Drag and drop
- Easy UI update
- Configuration files
- Multithreading
- Network programming
- HTML classes
- OpenGL integration
- Diagramming application support
- OLE automation support
- Multiple language support
- A user community

wxWidgets – deployment

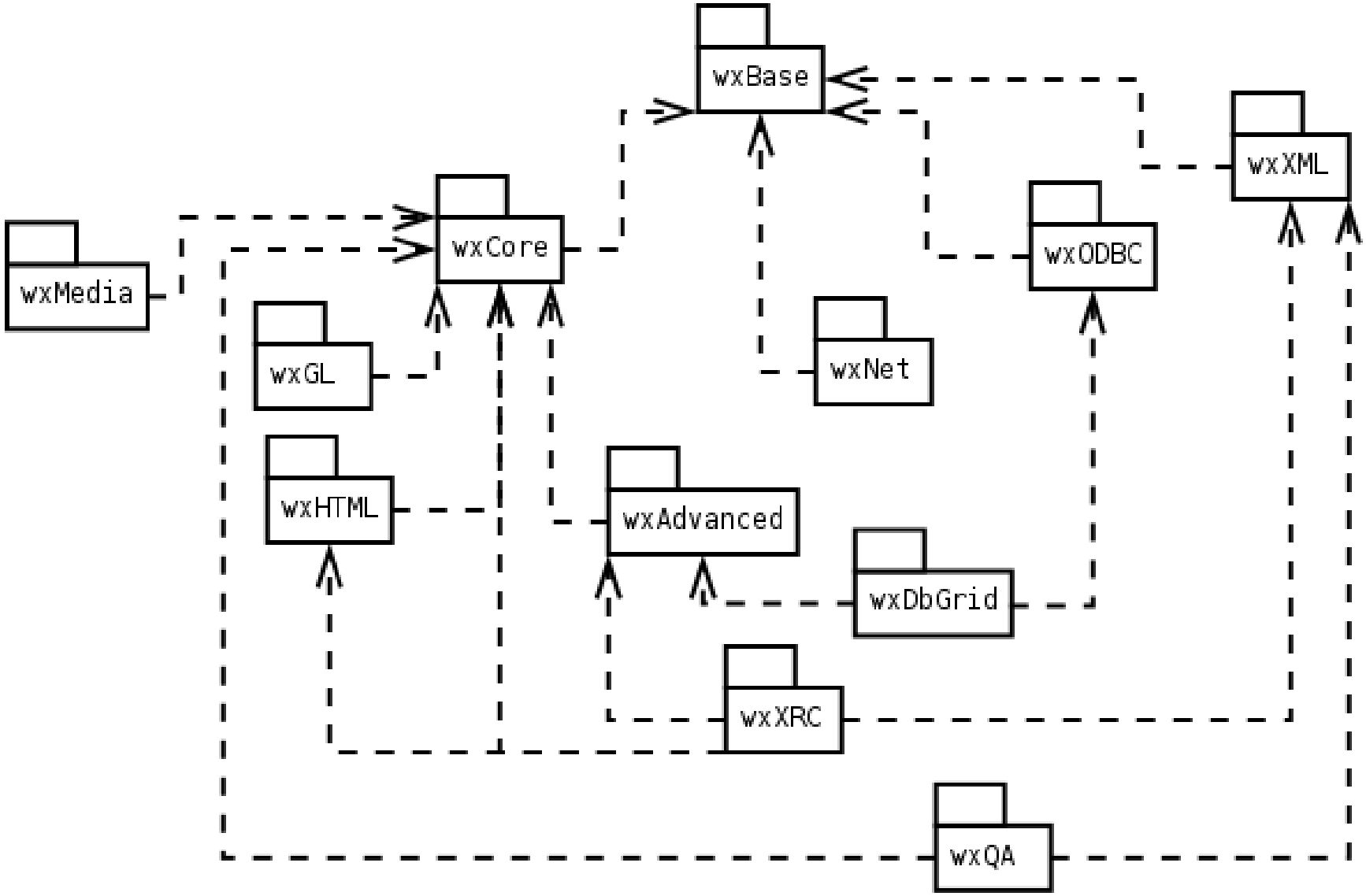
- Audacity
- AVG
- Code::Blocks
- RapidSVN
- VLC player alternative interface
- BitTorrent
- Filezilla



wxWidgets – development environments

- Boa Constructor (Python)
- wxGlade
- wxFormBuilder
- Code::Blocks (via wxSmith plugin)
- CodeLite
- wxDev-C++
- plugin for Microsoft Visual Studio

wxWidgets – class hierarchy



wxWidgets

- Low level classes for non visual objects
  - wxHashMap, wxString, wxList, wxPathList, etc.
- Support for RTTI (Run Time Type Information) – meta information about classes (wxClassInfo) == internal storage of references on some objects
- Unicode support (depends on flags in compilation time)
  - macro wxT for translation of characters into correct internal representation
- RichText editor, regular expressions, etc.

wxWidgets

- macro _T() or _() for application localisation
  - Hyper text help support
  - .HLP files support (wxHelpController)
- Multithread application support
  - wxThread, wxMutex, wxCriticalSection, wxCondition
- Network applications
  - sockets, http/ftp, wxURL
- Basic HTML viewer
  - wxHTML
- Database connection support
  - wxDb, wxODBC...

wxWidgets – wxString

- Optimized, unicode support, additional methods extending (AfterFirst, BeforeLast, operator<<, Printf, Format...)
  - wxString str1(”test1”);
  - wxString str2(”test2”);
  - wxString str3(”234.765”);
- str1.Upper();
- str1.Cmp(str2); // (case sensitive)
- str1.Freq(”e”); // count 'e' characters in string
- str3.ToDouble(&dbl);
- str.Trim(); // removes white spaces
- wxString::Format(wxT(”%s=%d\n”), str1.c_str(), dbl);

wxWidgets – positioning

- XML resource files (XRC)
  - Markup language for user interface
- layout system of element positioning – does not use fixed positioning, but Sizers
- User interface adapts position and size of elements according to specified conditions – easy customization
- The apperance is given by type of container wxGridSizer, wxBoxSizer, etc.

wxWidgets – example

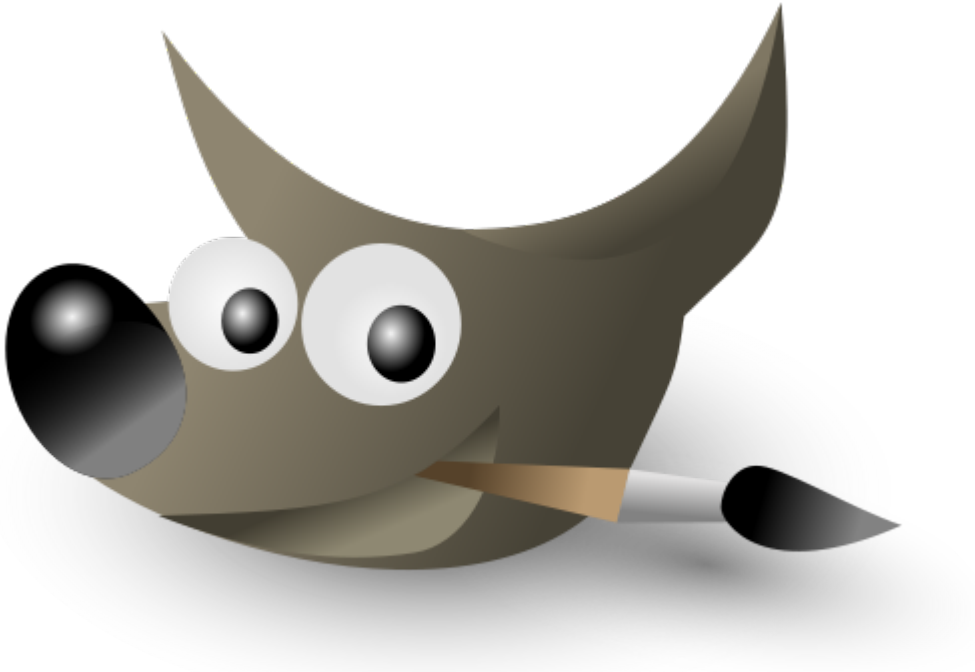
- Demonstration of
  - events
  - _T()
  - layouts

GTK

- Stability
- Cross Platform
- Language Bindings
- Mobile
- Accommodating
  - Look and feel, thread safe, utf8, ..
- Interfaces
  - Buttons, menus, combo boxes, layouts, trees, lists
- Foundations
  - Main loop, thread support, dynamic loading modules, lists, arrays, lexical scanner, base64

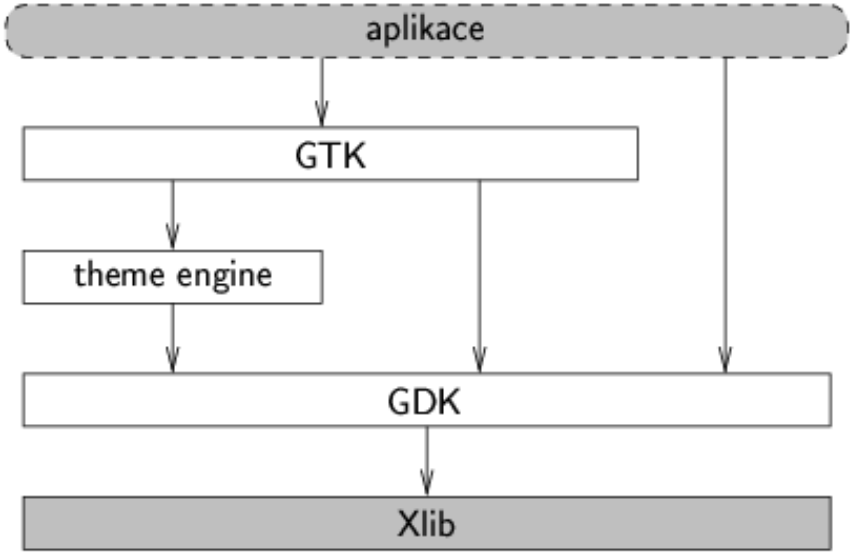
GTK – deployment

- Gnome
- Inkscape
- GIMP (The Gimp Toolkit == GTK)
- Firefox
- Pidgin



GTK – structure

- GLib – basic functions and data structures
- GDK – Xlib wrapper
- GObject – generic type and object system, signals, etc.
- GTK+ – core of GTK, definition of widgets
- Pango – rendering of text
- ATK - Accessibility Toolkit



GTK

- Application consist of tree of object instances. The interaction of application is done by delivering of events to corresponding objects. Application contains:
  - Initialization
  - Construction of widget instance tree
  - Registration of action functions
  - Realization and mapping
  - Application loop
- Initialization
  - gtk_init(argc, argv);

GTK

- Construction of object tree
  - Graphical widget term, which specifies any graphical object
  - Behavior and appearance of object is given by its class
  - The code of class itself handles most of operations
  - Widget is identified by data type GtkWidget
  - For each class exists function for creating of its instance
  - Base class is GtkWidgetClass

GTK

- containers
  - Are the elements, which allows to hold inner objects. The purpose of containers is for example positioning of objects.
  - The name of function is given by container type
  - gtk_container_add(GTK_CONTAINER(main), button);

GTK – object features

- There is naming convention
  - gtk_CLASS_get_xxx(...);, gtk_CLASS_set_xxx(...);
  - g_object_get(...);,g_object_set(...);
- The list of features is in reference manual
- Examples
  - gtk_widget_set_name(main, "main");
  - gtk_container_set_border_width(GTK_CONTAINER(mai
    n), 10);

GTK – callbacks

- signal
  - Signal is defined as a message sent by GtkObject instance as a reaction to any element in the environment (for example button click)
- event
  - The events are related to environment outside to application (for example timer)

GTK – callbacks

- Registration of callback is same for event and signal
  - g_signal_connect(G_OBJECT(MyNiceButton), "clicked", G_CALLBACK(my_nice_action), user_value);
    - Name of object
    - string name of signal or event
    - pointer to function
    - user value (gpointer)
    - Casts G_OBJECT() G_CALLBACK()
- Function for signal
  - void my_nice_action(GtkWidget *widget, gpointer user_data);
- Function for event
  - void delete_action(GtkWidget *widget, GdkEvent *event, gpointer user_data);

GTK – visibility of object

- Realization and mapping
  - gtk_widget_show(...);
  - gtk_widget_show_all(...);
  - gtk_widget_hide(...);
- Object management
  - The objects unmanaged by default
  - To be managed must be
  - po vložení do kontejneru se stanou spravovanými

    gtk_widget_show(button);

    gtk_widget_show(main);
- Application loop
  - Everything is in function – gtk_main();

GTK – glade

- GUI design tool
  - Skeleton of application, programmer fill in selected functions
  - Or XML file with user interface definition
- The XML file way is preferred
  - GUI is loaded from XML file in run time by libglade

Qt

- Intuitive C++ class library
- Portability across desktop and embedded operating systems
- Integrated development tools with cross-platform IDE
- High runtime performance and small footprint on embedded

Qt – deployment

- KDE (+ KOffice)
- Google Earth
- VLC
- VariCAD
- Skype (linux)
- Amarok 2

Qt – tools

- Qt Designer
  - Tool for Graphical User Interface design
- Qt Assistant
  - documentation
- Qt Linguist
  - Localization tool
- qmake
  - makefile

Qt - components

- core (included by default) QtCore module
- gui (included by default) QtGui module
- network QtNetwork module
- opengl QtOpenGL module
- phonon Phonon Multimedia Framework
- sql QtSql module
- svg QtSvg module
- xml QtXml module
- webkit WebKit integration
- qt3support Qt3Support module
  (.pro “QT += xml”, apod.)

Qt – MOC Meta Object Compiler

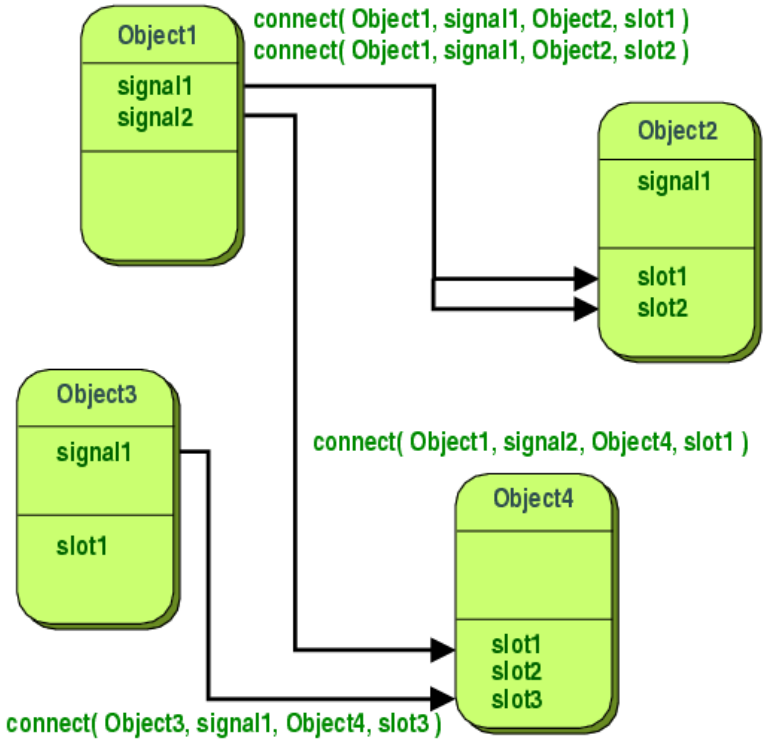
- Extension of C++ for user interface related functions
  - Macro language
  - Special preprocessing (meta object compiler – moc)
- class QObject – base class of all Qt objects
- macro Q_OBJECT
  - in private part of class
  - Dynamic properties of objects
  - Slots and signals
  - Recommended in all QObject child classes
- The code is inside of C++ code

Qt – QObject

- The objects are organized in tree hierarchy
  - The “parent” (outter) objects are defined in constructor
  - The destructors are called automatically on whole tree
- Support for reflection
  - QMetaObject describes QObject – it is possible to “inspect” the objects and its features and browse whole tree hierarchy
- Event handling
  - The events are sent to thread in the was object created
- Support for timers
  - After calling of startTimer(interval) is called method timerEvent(QTimerEvent *event)
- Dynamic properties
- Slots and Signals

Qt – Slots and Signals

- Provides communication between objects
- Type safe, but slower
- Signal is emited when event occurs
- Slot is the method called after catching of singal
- connection:



Qt – demo

- Compilation
- GUI design
- Localisation – tr() function
- Signals and slots
- Layouts

Acknowledgements

- http://www.root.cz/clanky/programovani-x-window-system-gtk-plus/
- http://www.wxwidgets.org/
- http://www.gtk.org/
- http://qt.nokia.com/products
- Smart J., Hock K., Csomor S.: Cross-Platform GUI Programming with wxWidgets. 2005, ISBN 0131473816, 744 pages.