Лекция №6

Кросс-платформенное программирование

Кроссплатформенное программное обеспечение

по материалам
http://ru.wikipedia.org/wiki/Кроссплатформенное_программное_обеспечение

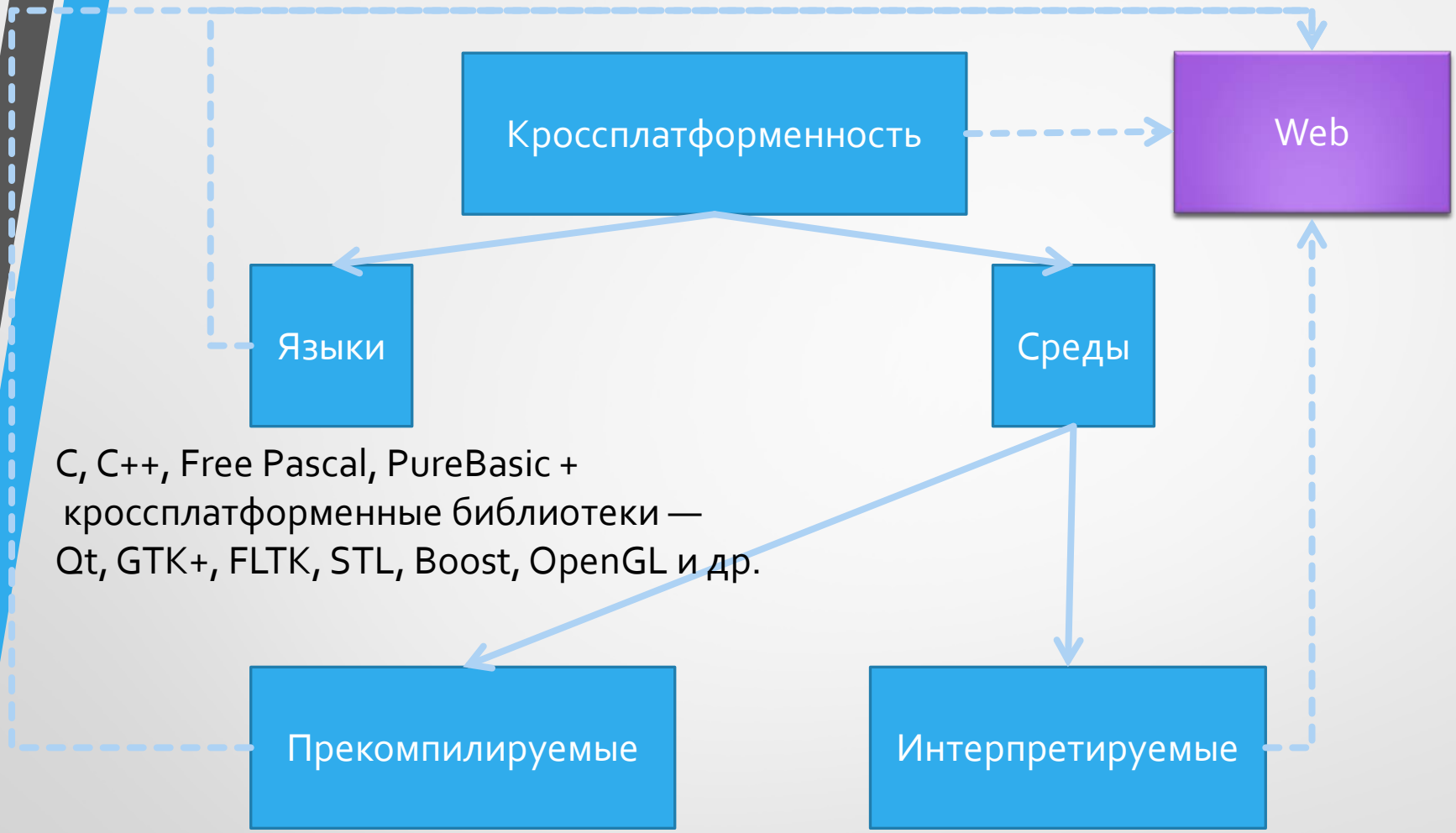
http://en.wikipedia.org/wiki/Cross-platform

Кроссплатформенное (межплатформенное) программное обеспечение

Кроссплатформенное (межплатформенное) программное обеспечение — программное обеспечение, работающее более чем на одной аппаратной платформе и/или операционной системе. Типичным примером является программное обеспечение, предназначенное для работы в операционных системах Linux и Windows одновременно.

Как обеспечить кроссплатформенность?

1. Кроссплатформенные языки программирования
2. Кроссплатформенные среды исполнения (компилируемые/интерпретируемые)


Кроссплатформенность
Web
Языки
Среды
C, C++, Free Pascal, PureBasic +
кроссплатформенные библиотеки —
Qt, GTK+, FLTK, STL, Boost, OpenGL и др.
Прекомпилируемые
Интерпретируемые
Java и C#
PHP, ActionScript, Perl, Python, Tcl и Ruby

Кроссплатформенный пользовательский интерфейс

На различных ОС — независимо от того, как технически достигнута работа в них — стандартные элементы интерфейса имеют разные размеры. Поэтому простое жёсткое позиционирование элементов интерфейса невозможно — под другой ОС они могут налезать друг на друга. Существует несколько подходов.

1. Единый стиль, общий для всех ОС. Программы выглядят одинаково под всеми ОС. Так работают интерфейсные библиотеки Java наподобие Swing.

+ можно жёстко расставлять элементы управления на манер Delphi; оригинальный стиль.

- системе приходится иметь свои экранные шрифты; стиль отличается от стиля ОС.

2. Самоадаптирующийся интерфейс, подстраивающий сетку под реальные размеры элементов управления. Типичные примеры — wxWidgets, XUL.

\+ стандартный стиль ОС, очень быстрый и «скинующийся» под Windows XP, Vista и 7; некоторая автоматизация локализации.

\- чтобы собрать самоадаптирующуюся сетку, требуется квалифицированный программист; затруднена плотная компоновка.

3. Гибридный подход реализован в GTK+.

+ шрифты можно брать из системы, а не «тащить» свои; некоторая автоматизация локализации.

- берёт все недостатки от первых двух подходов. Стиль отличается от стиля ОС; затруднена плотная компоновка

Эмуляция

Если программа не предназначена для исполнения (запуска) на определённой платформе, но для этой платформы существует эмулятор платформы, базовой для данной программы, то программа может быть исполнена в среде эмулятора.

Обычно исполнение программы в среде эмулятора приводит к снижению производительности по сравнению с аналогичными программами, для которых платформа является базовой, так как значительная часть ресурсов системы расходуется на выполнение функций эмулятора.

Например, для запуска Windows программ под Linux используется эмулятор Wine

Условная компиляция

Препроцессор языка Си предоставляет возможность компиляции с условиями. Это допускает возможность существования различных версий одного кода. Обычно, такой подход используется для настройки программы под платформу компилятора, состояние (отлаживаемый код может быть выделен в результирующем коде), или возможность проверки подключения файла строго один раз.

Условная компиляция

- В общем случае, программисту необходимо использовать конструкцию наподобие этой:

```
# ifndef FOO_H
# define FOO_H
...(код заголовочного файла)...
# endif
```

Такая «защита макросов» предотвращает двойное подключение заголовочного файла путем проверки существования этого макроса, который имеет то же самое имя, что и заголовочный файл. Определение макроса FOO_H происходит, когда заголовочный файл впервые обрабатывается препроцессором. Затем, если этот заголовочный файл вновь подключается, FOO_H уже определен, в результате чего препроцессор пропускает полностью текст этого заголовочного файла.

То же самое можно сделать, включив в заголовочный файл директиву:

- # pragma once

Условия препроцессора можно задавать несколькими способами, например:

```
# ifdef x
...
# else
...
# endif
```

Этот способ часто используется в системных заголовочных файлах для проверки различных возможностей, определение которых может меняться в зависимости от платформы; например, библиотека Glibc использует макросы с проверкой особенностей с целью проверить, что операционная система и оборудование их (макросы) корректно поддерживает при неизменности программного интерфейса.

wxWidgets

wxWidgets (ранее известная как wxWindows) — это кросс-платформенная библиотека инструментов с открытым исходным кодом для разработки кроссплатформенных на уровне исходного кода приложений, в частности для построения графического интерфейса пользователя (GUI).

wxWidgets разработана не только для того, чтобы создавать GUI. Она также имеет набор классов для работы с графическими изображениями, HTML, XML документами, архивами, файловыми системами, процессами, подсистемами печати, мультимедиа, сетями, классы для организации многопоточности, отладки, отправки дампов и множество других инструментов.

wxWidgets

По традиции, версии wxWidgets для каждой платформы обозначаются добавлением префикса wx к сокращенному названию платформы.

Например, wxWidgets для Windows обозначается как wxMSW, wxWidgets для GTK – как wxGTK, wxWidgets для X11 – как wxX11, и т. д. Еще одна интересная возможность, связанная с многоплатформенностью wxWidgets – кросскомпиляция. На сайте проекта можно найти инструкции по компиляции wxWidgets-программ для Windows из-под Linux.

wxWidgets

- Помимо собственно визуальных компонентов, wxWidgets предоставляет в распоряжение программиста классы для работы с базами данных (поддерживаются интерфейсы ODBC, XBase, SQLite), классы для работы с сокетами и популярными сетевыми протоколами, а также специальные классы для работы с HTML.

Установка wxWidgets

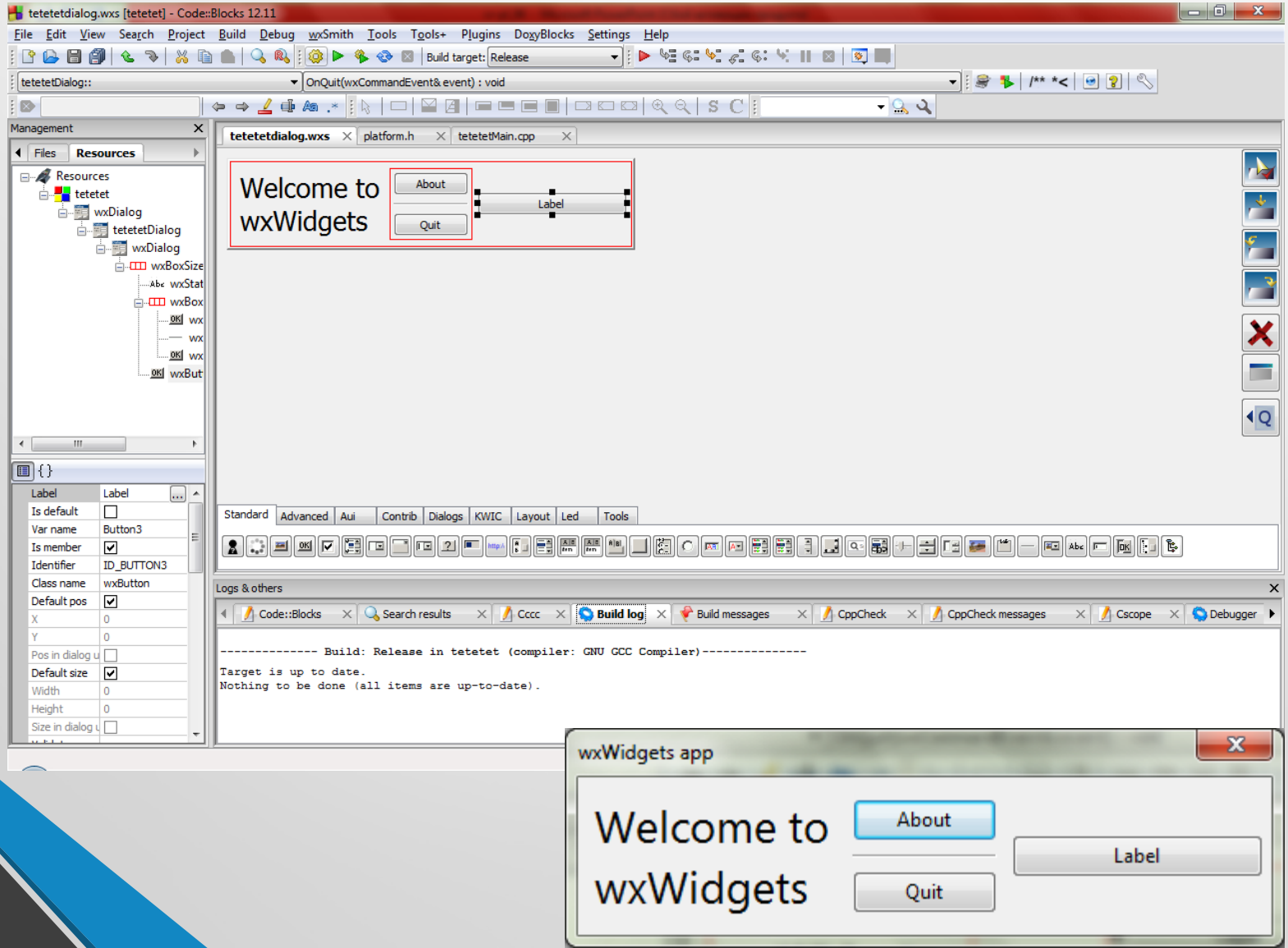
Подробная видеоинструкция

- http://www.youtube.com/watch?v=QuPiZ86EFhQ

Что необходимо:

CodeBlocks + MingW http://codeblocks.org/ (~100 Mb)

WxWidgets http://wxwidgets.org/


tetetetdialog.wxs [tetetet] - Code::Blocks 12.11
File Edit View Search Project Build Debug wxSmith Tools Tools+ Plugins DoxyBlocks Settings Help
Build target: Release
tetetetDialog::
OnQuit(wxCommandEvent& event) : void
Management
Files
Resources
Resources
tetetet
wxDialog
tetetetDialog
wxDialog
wxBoxSize
wxStat
wxBox
wxBut
tetetetdialog.wxs
platform.h
tetetetMain.cpp
Welcome to wxWidgets
About
Quit
Label
Label
Is default
Var name
Button3
Is member
Identifier
ID_BUTTON3
Class name
wxButton
Default pos
X
Y
Pos in dialog u
Default size
Width
Height
Size in dialog u
Standard
Advanced
Aui
Contrib
Dialogs
KWIC
Layout
Led
Tools
Logs & others
Code::Blocks
Search results
Cccc
Build log
Build messages
CppCheck
CppCheck messages
Cscope
Debugger
---------- Build: Release in tetetet (compiler: GNU GCC Compiler)----------
Target is up to date.
Nothing to be done (all items are up-to-date).
wxWidgets app
Welcome to wxWidgets
About
Quit
Label

Для Linux

- Установить Linux или VirtualBox, а затем установить в нем Linux
- Установить CodeBlocks, wxSmith, libwxwidgest