C++, GUI Programming, and You.
Wayne Manselle

Outline

• GUI Programming in C++
• The Qt Library
• The WxWidgets Library
• Visual C++

Qt

• Developed by TrollTech
• Comes in Open Source and Commercial Flavours
• Adds nifty features to your C++
  • Garbage Collection!

Qt Event Handling

• QEventLoop
  • Can be instantiated arbitrarily
• QApplications always have a QEventLoop
  • This is the main event loop
• Filterable
  • User input, Socket Notifiers, Defer, X11Timer Events
• Active Event Loop pulls events off of queue
  • No events? Wait state
Events

- QEvent is the parent of all events.
  - Usually sent by Windowing System
    - But can be manually sent
- QEvent uses Types
  - An Int Enumeration
  - Over 100 System defined Types
  - Type IDs 1000-65535 reserved for User Defined Events

Qt Example Widgets

- Qt has access to a plethora of Widgets
- Here are some nice pictoral examples.

Group Boxes

A Configuration Dialog
**Menus in a Window**

*Example image of a menu with dropdown options and text:*

**Spin Boxes**

*Example spin box widget with options:*

**Qt Layout Management**

- **Layout Widget**
  - A widget that organizes the QObjects added to it.
  - `QHBox`, `QVBox`, `QGrid`
- **QLayout**
  - Much like Java’s layouts
  - `QGridLayout == GridBagLayout`
  - More Complicated to use, but freer
  - Possibly to extend QLayout for customization
#include <QApplication>
#include <QPushButton>

int main (int argc, char *argv[]) {
    QApplication app(argc, argv);
    QPushButton hello("Hello world!");
    hello.resize(100, 30);
    hello.show();
    return app.exec();
}
WxWidgets

- Yet another C++ GUI Library!
- Released by the Artificial Intelligence Applications Institute at the University of Edinburgh in 1992
  - Originally an in house tool

WxWidgets Event Handling

- wxEvtHandler
  - Takes events from the event queue
  - Invokes method in the event table on itself
- Why is WxWidgets different?
  - wxWindow is an ancestor of wxEvtHandler
    - Everything that’s a window is an event handler
    - In WxWidgets, everything that’s visible is a window!

WxWidgets Main Event Loop

- Every WxWidget application must inherit from the class wxApp
  - wxApp inherits from wxWindow
  - Forces the wxApp ancestor to be the main event loop

Events in WxWidgets

- Descendents of wxEvent class
  - Also uses int Type enumeration
    - Ints are dynamically assigned at run time
- Command Events vs. Non-Command
  - Command events are passed recursively up to a window’s parents
  - Non-Command events are not.
- Command events are almost always user generated.
WxWidgets Example Widgets

wxGLCanvas

wxDialog Example

WxWidgets Layout Management

- wxSizer is the parent Layout class
- Functions by adding wxWindow instances directly to an instance of a wxSizer
- 5 implementing ancestors
  - wxBoxSizer, wxStaticBoxSizer, wxGridSizer, wxFlexGridSizer, CreateButtonSizer
/*
 * hello-world sample by Robert Roebling
 */

#include "wx/wx.h"

class MyApp: public wxApp {
    virtual bool OnInit();
};

class MyFrame: public wxFrame {
public:
    MyFrame(const wxString& title, const wxPoint& pos, const wxSize& size);
    void OnQuit(wxCommandEvent& event);
    void OnAbout(wxCommandEvent& event);
    DECLARE_EVENT_TABLE()
};

BEGIN_EVENT_TABLE(MyFrame, wxFrame)
    EVT_MENU(ID_Quit, MyFrame::OnQuit)
    EVT_MENU(ID_About, MyFrame::OnAbout)
END_EVENT_TABLE()

IMPLEMENT_APP(MyApp)

bool MyApp::OnInit() {
    MyFrame* frame = new MyFrame("Hello World", wxPoint(50, 50), wxSize(450, 340));
    frame->Show(TRUE);
    SetTopWindow(frame);
    return TRUE;
}
Yeah. Finally.

```cpp
void MyFrame::OnQuit(wxCommandEvent& WXUNUSED(event))
{
    Close(TRUE);
}
void MyFrame::OnAbout(wxCommandEvent& WXUNUSED(event))
{
    wxMessageBox("This is a wxWindows Hello world sample",
                 "About Hello World", wxOK | wxICON_INFORMATION, this);
}
```

Visual C++

- Microsoft's Implementation of C++
- With a shiny front end of Visual Studio
```
#pragma once
namespace Labels {
    using namespace System;
    using namespace System::ComponentModel;
    using namespace System::Collections;
    using namespace System::Windows::Forms;
    using namespace System::Data;
    using namespace System::Drawing;
public ref class Form1 : public System::Windows::Forms::Form
{
    public:
        Form1(void)
        {
            InitializeComponent();
        }
```

Form 1 Continues

protected:
- ~Form1()
  
  if (components)
    do this to delete components;

private: System::Windows::Forms::Label^ label1;
protected:
private: System::Windows::Forms::WebBrowser^ webBrowser1;
private:
System::ComponentModel::Container^ components;

More Form 1

#pragma region Windows Form Designer generated code
    using System::Windows::Forms;
    using System::ComponentModel;
    using System::Collections;
    using System::Windows::Forms;
    using System::Text;
    using System::Reflection;
    using System::Runtime::InteropServices;
    using System::Runtime::SafeHandles;
    #region Windows Form Designer generated code
    void InitializeComponent()
    {
      this->label1 = gcnew System::Windows::Forms::Label();
      this->webBrowser1 = gcnew System::Windows::Forms::WebBrowser();
      this->label1->AutoSize = true;
      this->label1->Font = gcnew System::Drawing::Font("Microsoft Sans Serif", 12, System::Drawing::FontStyle::Regular);
      this->webBrowser1->Location = System::Drawing::Point(22, 73);
      this->label1->AutoSize = true;
      this->label1->Font = gcnew System::Drawing::Font("Microsoft Sans Serif", 12, System::Drawing::FontStyle::Regular);
      this->webBrowser1->MinimumSize = System::Drawing::Size(20, 20);
      this->label1->AutoSize = true;
      this->label1->Font = gcnew System::Drawing::Font("Microsoft Sans Serif", 12, System::Drawing::FontStyle::Regular);
      this->webBrowser1->Url = gcnew System::Uri("http://www.google.com", System::UriKind::Absolute);
      this->webBrowser1->Size = System::Drawing::Size(352, 279);
      this->webBrowser1->TabIndex = 1;
      this->webBrowser1->Location = System::Drawing::Point(22, 73);
      this->webBrowser1->MinimumSize = System::Drawing::Size(20, 20);
      this->webBrowser1->Name = L"webBrowser1";
      this->webBrowser1->Url = gcnew System::Uri("http://www.google.com", System::UriKind::Absolute);
      this->webBrowser1->Size = System::Drawing::Size(352, 279);
      this->webBrowser1->TabIndex = 1;
      this->webBrowser1->Location = System::Drawing::Point(22, 73);
      this->webBrowser1->MinimumSize = System::Drawing::Size(20, 20);
      this->webBrowser1->Name = L"webBrowser1";
    }
#pragma endregion

Last of Form 1

#pragma endregion Windows Form Designer generated code
    using System::Windows::Forms;
    using System::ComponentModel;
    using System::Collections;
    using System::Windows::Forms;
    using System::Text;
    using System::Reflection;
    using System::Runtime::InteropServices;
    using System::Runtime::SafeHandles;
    #endregion
    #pragma endregion

Labels.cpp

#include "stdafx.h"
#include "Form1.h"

using namespace Labels;

[STAThreadAttribute]
int main(array<System::String ^> ^args)
{
  // Enabling Windows XP visual effects before any controls are created
  Application::EnableVisualStyles();
  Application::SetCompatibleTextRenderingDefault(false);
  // Create the main window and run it
  Application::Run(gcnew Form1());
  return 0;
}
Hey, where’s GTK+?

• /GTK+.configure
• Mac: You need Cairo
• /Cairo/.configure
• Mac: You need libpng
• /libpng/.configure; make; make install
  • Success!
• /Cairo/.configure
• Mac: You need libpng

You Are Now Entering: Dependency Hell!

destroy();

• Questions?