ECE 3574: Introduction to Qt

Changwoo Min

Administrivia

- **Grading of milestone 0 is done**
  - Send me an email if you have any question
- **Milestone 1 due date is extended**
  - Due: 2/26 by 11:50 PM
  - Start code was released via email
  - It has my reference implementation of `lexer.cpp` and the bare minimal start code for `parser.cpp`
  - **At final submission, you MUST use your own `lexer.cpp` and `parser.cpp`.**

Review Milestone 1 start code

- See code

Meeting 9: Introduction to Qt

The goal of today's meeting it to learn about about a popular cross-platform library called Qt.

- Windows and **Event Loops**
- Widgets
- **Signals and Slots**
- Meta-Object Compiler
- Exercise

User Interaction

In C++ (including the standard library), the built-in mechanisms for user input are

- specifying command line arguments (not interactive)
- standard input (interactive but synchronous)
- signals, e.g. Control-C (asynchronous)

C++ itself also has nothing to say about displays

- It assumes only standard output and standard error.
  - The OS provides the notion of a console a way to enter input into standard input one line at a time, and a way to view standard output/error. multiplexes different programs input/output this interaction dates to the very early days of computing
- This provides powerful language-style interaction but is limited the kind of user interaction that can be supported.

Modern OSs often provide some abstraction of a graphical display

- A library which interacts with the display hardware (vector or bitmap). It provides
  - a way to draw 2D shapes and/or images on the screen
  - a way to register user events related to those objects (clicks, etc)
  - a way to multiplex different programs on the same display (focus)

The dominant abstraction is called WIMP

- WIMP = windows, icons, menus, pointer
  - the display is made up of a set of windows
  - a program has access to one or more windows
  - a window is a collection of widgets
  - a pointing device is used to register actions on a widget (event) the program can change the visual appearance of the widget (draw or render)
- The main concept is the event-loop.

Event Loop

1. Draw the widgets
2. Collect all events
3. Process all events
4. Goto 1

- This loop takes over the main thread of the program.
- All work (in a single threaded application) happens in the event loop.
- Called *Event Driven Programming*. Event cause code to run changing the program state and causing side effects.

The windowing system library is platform dependent

- Common native windowing libraries:
  - On Windows: Win32, WinForms, MFC, WPF
  - On Mac: Carbon, Quartz
  - On Unix: X11
- Maintaining an application across all three platforms is cumbersome, but sometimes warranted.

An alternative is to use another library layer that abstracts away the platform

- GTK+
- WxWindows
- FLTK
- Qt

We will be discussing Qt, a *huge* library, focusing on the GUI part.

In Qt widgets and events are objects

- `QApplication` handles the event loop
- Your user interface code is embedded in a widget (using dynamic polymorphism)
- Events are delivered to your widget if appropriate (events are filtered)
- If your widget needs to change it calls a method called `update`
- Events can trigger other events. In this view a program is a collection of widgets communicating via events.
- See http://doc.qt.io/qt-5/eventsandfilters.html

Exercise 09: Part 1: A Basic Qt Window

- See the website

Qt also uses another parallel form of communication among widgets

- **Signals and Slots**
  - extends C++ syntax to add slots, special member functions
  - requires a code generator (meta-object compiler or `moc`)
  - code can emit signals, which are objects
  - these signals can be connected to slots, members of other objects
  - when an signal is emitted it is sent to all slots that it is connected to
- Allows dynamic and one-to many communication among objects as opposed to just calling a member (one-to-one).

Exercise 09: Part 2: Signal/Slot example

- See the website

Useful Qt links

- Overview of Qt
- Qt for Beginners
- Qt Examples And Tutorials
- Qt API documentation
- Qt Event System
- Qt Signals and Slots
- Lambda expression in C++: Link 1, Link 2

Next Actions and Reminders

- Read links on Dynamic Polymorphism