SE3910 – REAL TIME SYSTEMS

Introduction to Qt

ROADMAP

- Today
  - Introduction to Qt ✓
- Monday
  - Exam Review / Catch Up ✓
- Wednesday
  - Midterm Exam

Friday:

No class

OBJECTIVES

- Explain the purpose for the GSTREAMER libraries
- Define the concept of pads, bins, and piplines
- Compare and contrast source, sink, and filter elements
- Explain how a pipline can be graphically represented
- Explain how we can use an oscilloscope to measure execution time of a method

HOW DO YOU BUILD A USER INTERFACE?

Java Swing

AWT

→ Command Line

Web App → html

.js

Android → XML

WHAT MAKES A GOOD TOOLKIT?

- Implementation language — Base language
- Easy to program
- Consistent interface
- Little code for large results
- Excellent documentation
- Availability of tools for code generation
- Portability → Different platforms
- Easy to extend

GUI TOOLSKITS FOR C / C++

- Qt
  - Written in C++
  - Qt programs are portable
  - Object oriented. Allows your window to be encapsulated into a real C++ class
  - Easy to hand code

OO things

- GTK
  - Written in C
  - GTK programs are mostly limited to UNIX though GTK does exist on other platforms
  - Not object oriented
- Motif
  - Written in C
  - Motif programs are limited to UNIX
  - Not object oriented
  - Difficult to hand code

old

- MFC
  - MFC programs are limited to Windows

C#/Windows

QT FEATURES

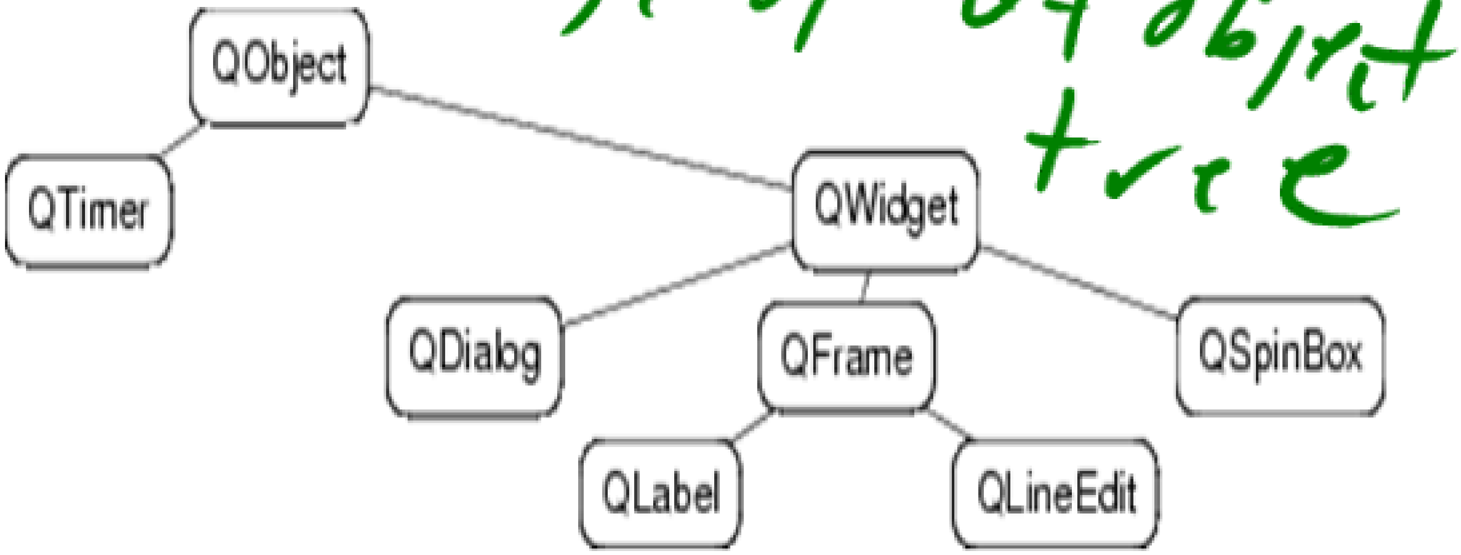
- Fully object-oriented
- Consistent interfaces
- Rich set of widgets (controls)
- Have native look and feel
- Drag and drop
- Customizable appearance
- Utility classes
- OpenGL support
- Network support
- Database support
- Plugin support
- Unicode/Internationalization support
- GUI builder

Controls Widgets

WIDGETS

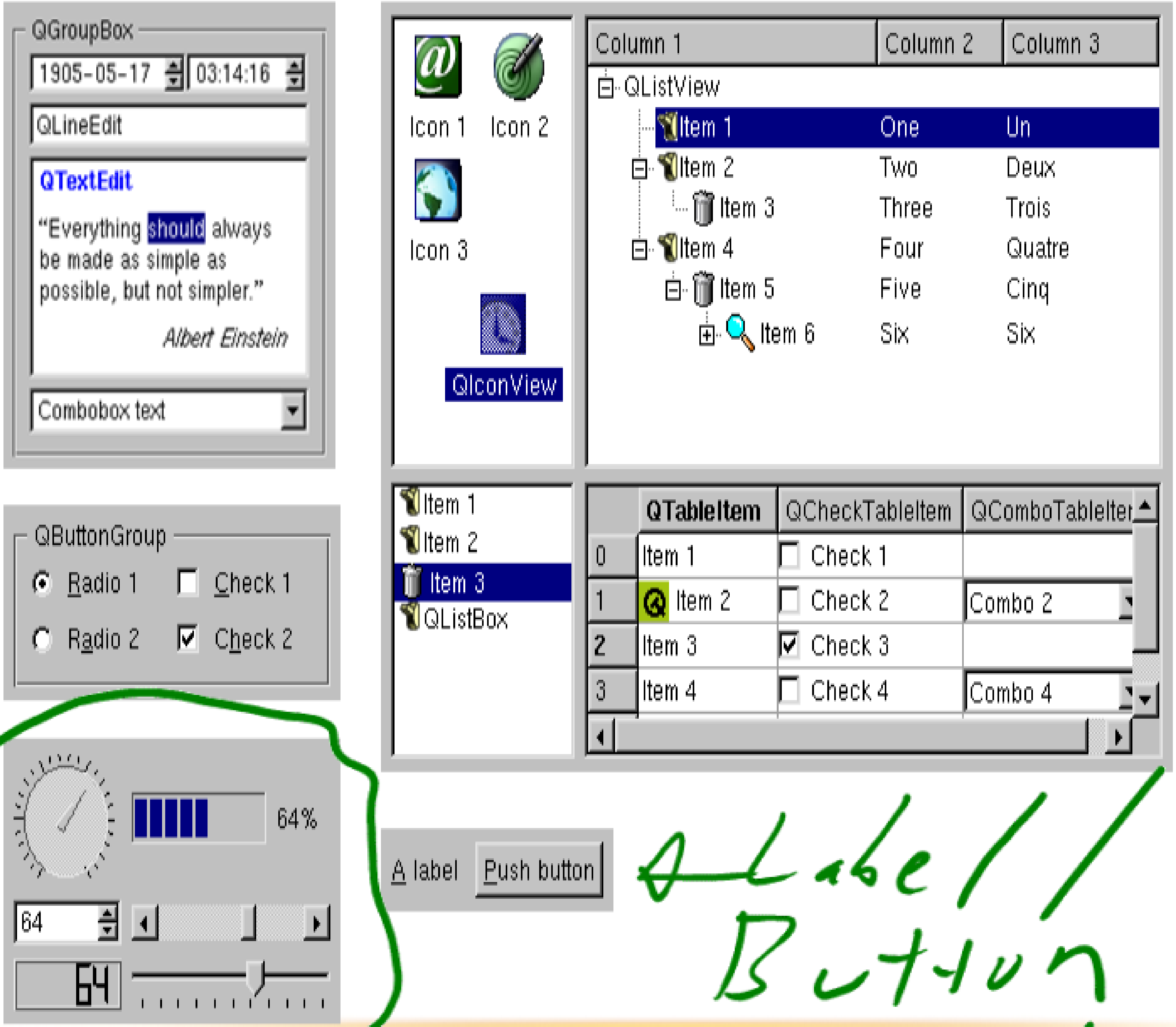
A widget is a user interface component such as a button or a scroll-bar

— Base of Qt object tree



- Reusable!
- Well defined interface
- Uses C++ inheritance
- All widgets derive from a common base
- Widgets may contain other widgets
- Custom widgets can be created from existing widgets or they can be created from scratch

EXAMPLE QT WIDGETS



QTextEdit

"Everything should always be made as simple as possible, but not simpler."

Albert Einstein

Column 1	Column 2	Column 3
QListView		
Item 1	One	Un
Item 2	Two	Deux
Item 3	Three	Trois
Item 4	Four	Quatre
Item 5	Five	Cinq
Item 6	Six	Six

- Item 1
- Item 2
- Item 3
- QListBox

	QTableItem	QCheckTableItem	QComboTableIter
0	Item 1	□ Check 1	
1	Item 2	□ Check 2	Combo 2
2	Item 3	☑ Check 3	
3	Item 4	□ Check 4	Combo 4

A label /

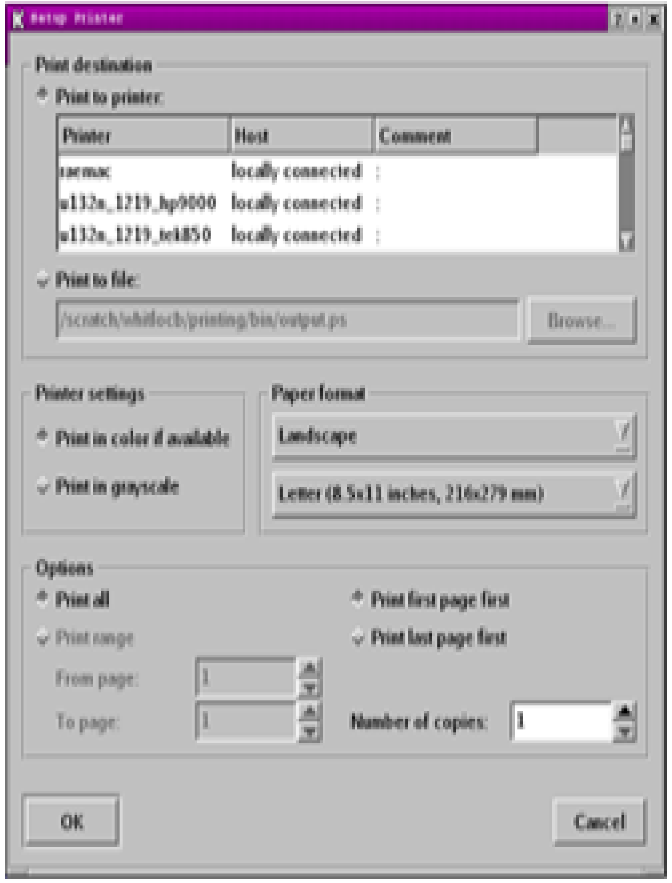
Button /

custom widget

QT BUILT IN DIALOG BOXES

- File dialog
- Font dialog
- Color dialog
- Printer dialog

font



Setup Printer

Print destination

● Print to printer:

Printer	Host	Comment
raemac	locally connected	:
u132n_1219_hp9000	locally connected	:
u132n_1219_tek850	locally connected	:

○ Print to file:

/scratch/whitlocb/printing/bin/output.ps

Printer settings

● Print in color if available

○ Print in grayscale

Paper format: Landscape; Letter (8.5x11 inches, 216x279 mm)

Options

● Print all

○ Print range

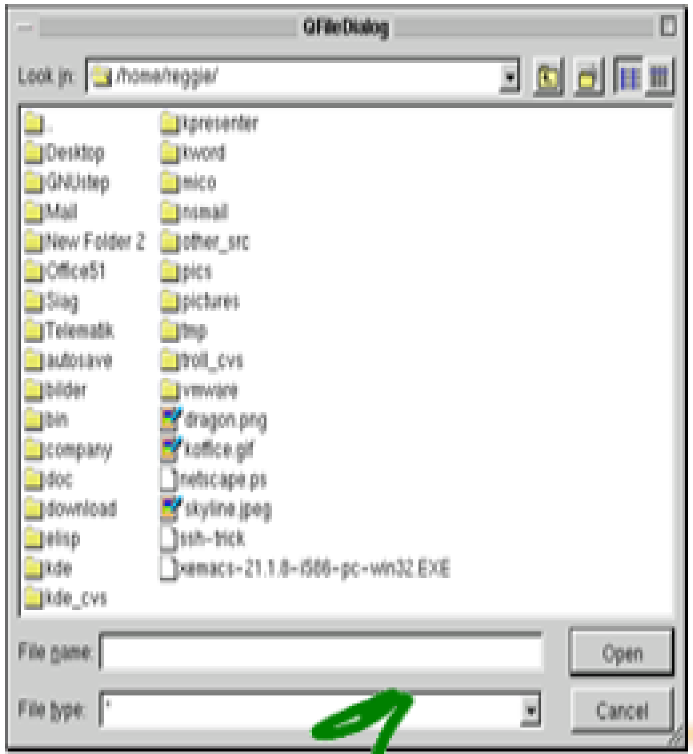
From page: 1

To page: 1

● Print first page first

○ Print last page first

Number of copies: 1



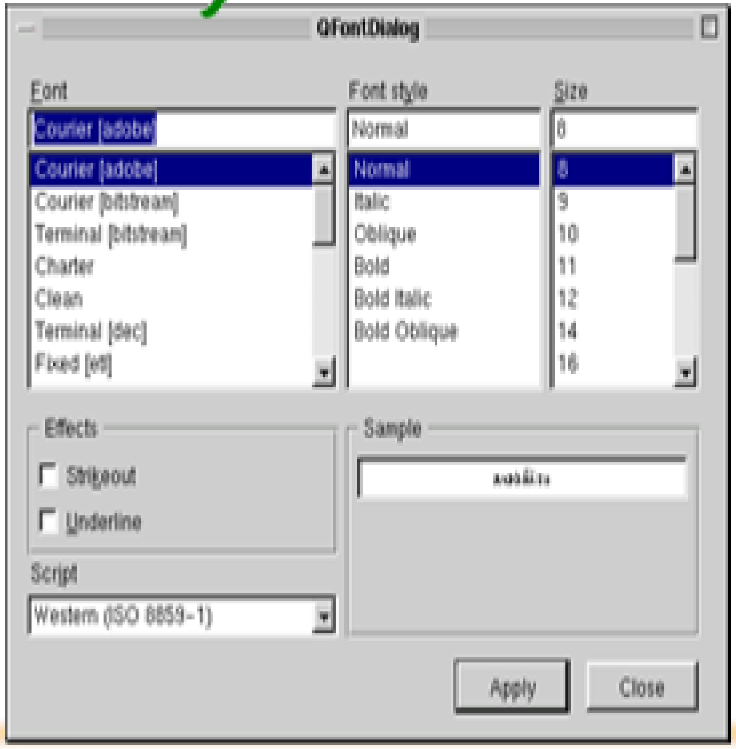
QFileDialog

Look in: /home/reggie/

```
..              kpresenter
Desktop         kword
GNUstep         mico
Mail            nsmail
New Folder 2    other_src
Office51        pics
Slag            pictures
Telematik       tmp
autosave        troll_cvs
bilder          vmware
bin             dragon.png
company         koffice.gif
doc             netscape.ps
download        skyline.jpeg
elisp           ssh-trick
kde             xemacs-21.1.8-i586-pc-win32.EXE
kde_cvs
```

File name:

File type: *



QFontDialog

Font: Courier [adobe]

Font style: Normal

Size: 8

```
Courier [adobe]       Normal        8
Courier [bitstream]   Italic        9
Terminal [bitstream]  Oblique       10
Charter               Bold          11
Clean                 Bold Italic   12
Terminal [dec]        Bold Oblique  14
Fixed [etl]                         16
```

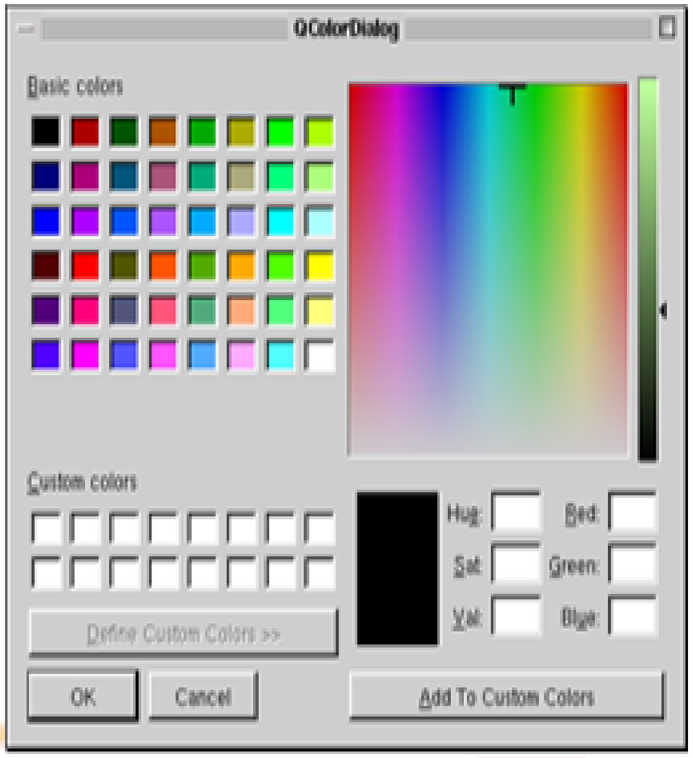
Effects

□ Strikeout

□ Underline

Sample

Script: Western (ISO 8859-1)



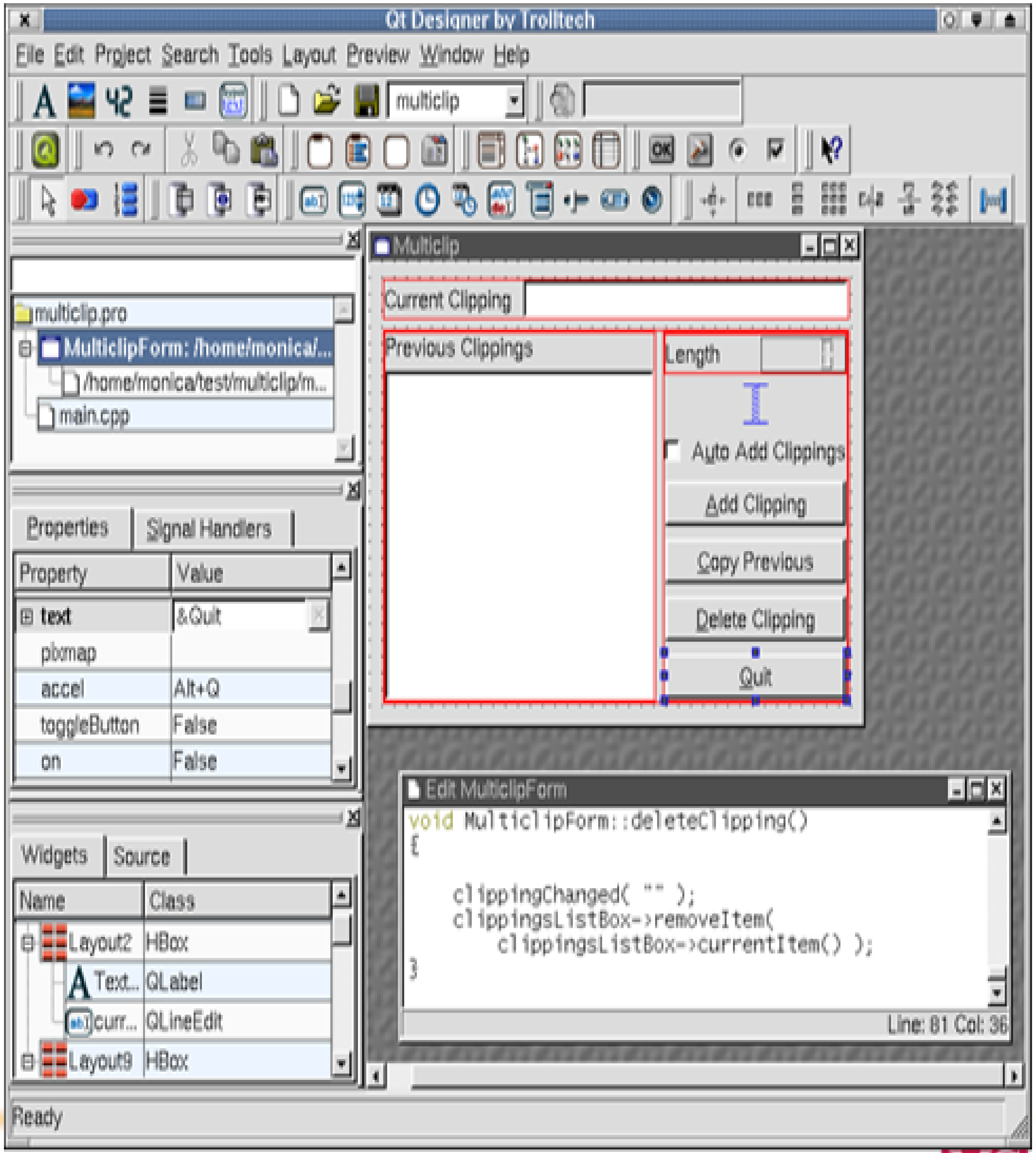
QColorDialog

SE3910 REAL TIME SYSTEMS

File

QT DESIGNER

- Written using Qt so it is available on all platforms where Qt is available
- Used to speed design of Qt applications
- Supports all Qt widgets and can be used to incorporate custom widgets



Property	Value
text	&Quit
pixmap	
accel	Alt+Q
toggleButton	False
on	False

Name	Class
Layout2	HBox
Text...	QLabel
curr...	QLineEdit
Layout9	HBox

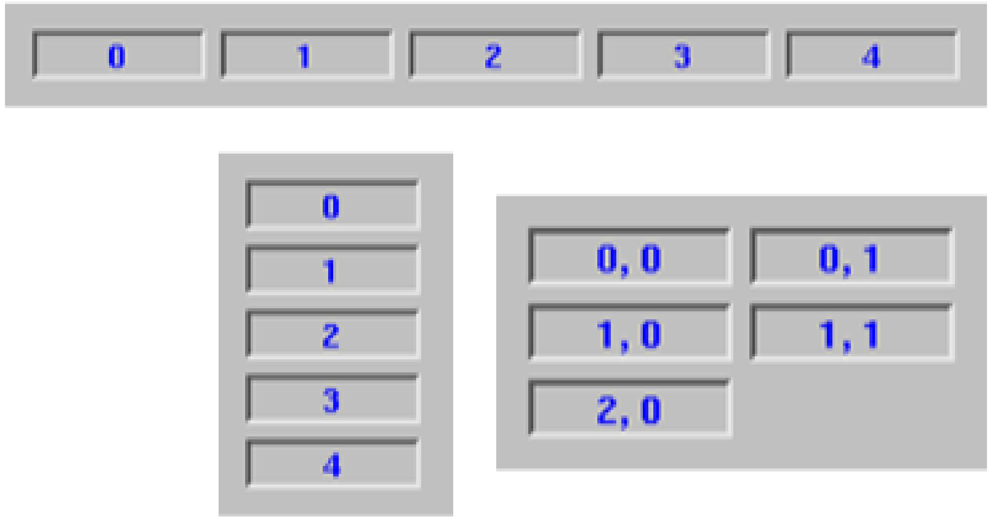
```
void MulticlipForm::deleteClipping()
{
    clippingChanged( "" );
    clippingsListBox->removeItem(
        clippingsListBox->currentItem() );
}
```

Line: 81 Col: 36

USING LAYOUTS

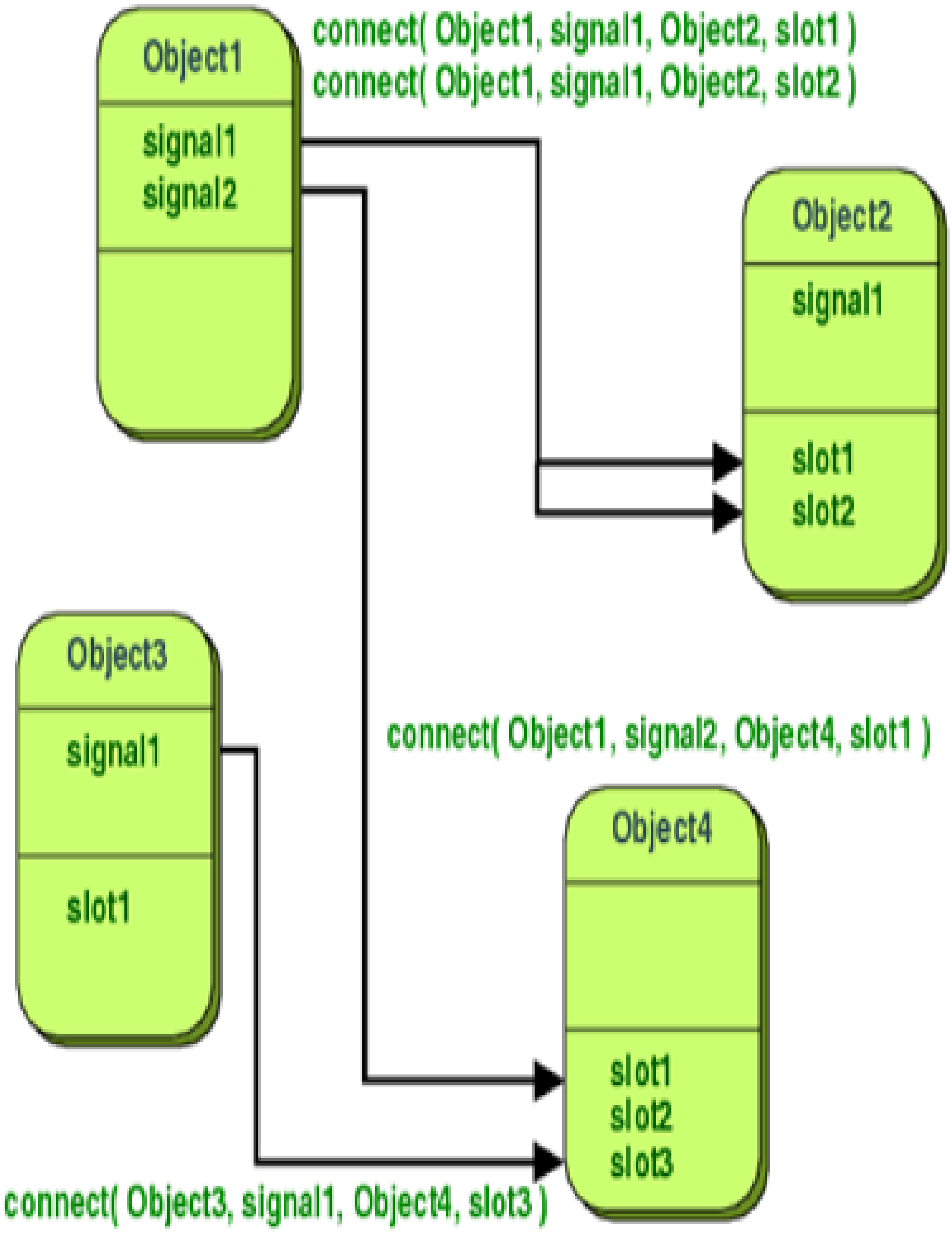
Layouts should be used to manage widget position and resize behavior

- Layout types
  - Horizontal
  - Vertical
  - Grid
- Layouts can be nested
- User can control layout spacing, stretch, and strut



SIGNALS AND SLOTS

- Signal
  - A signal is emitted whenever an event occurs
- Slot
  - A function which is invoked in response to a signal


connect(Object1, signal1, Object2, slot1)
connect(Object1, signal1, Object2, slot2)
Object1
signal1
signal2
Object2
signal1
slot1
slot2
connect(Object1, signal2, Object4, slot1)
Object3
signal1
slot1
Object4
slot1
slot2
slot3
connect(Object3, signal1, Object4, slot3)

OF SIGNALS AND SLOTS

- SIGNAL
  - Preprocessor directive which is replaced during compilation
  - A void function declaration
- SLOT
  - A preprocessor directive which is removed at compilation
  - A void function declaration
- Emit
  - A preprocessor directive (again)
  - A keyword which indicates that a signal is emitted in response to an event.

SIGNALS AND SLOTS

- Facilitate interprocess communications
- An object emits a signal with a certain prototype

Strongly typed

- Other objects may be connected to that object's signal
  - connect(button, SIGNAL(clicked()), qApp, SLOT(quit()));
- MOC preprocessor
  - Reads class header file and creates supplementary C++ code to support signals/slots

Autogenerates

  - All header files for classes defining signals/slots need to use MOC
  - Easily incorporated into Make rules
  - Transparent to user

QMAKE

- Tool that helps simplify the build process for development project across different platforms. → Crossplatform
  - Automates the generation of Makefiles so that only a few lines of information are needed to create each Makefile.
  - qmake can be used for any software project, whether it is written in Qt or not.
- Project can later be made using a simple make call.l

make

EXAMPLE

- Hello Class

A SECOND EXAMPLE

- A push button to quit

RUNNING ON THE BEAGLEBONE

- The same code can be made on the beaglebone
  - And we can demo it there as well.