Softwaretechnologie II (C++, Teil 2)

Øyvind Eide: oeide@uni-koeln.de

Andias Wira-Alam andias@gmail.com

QApplication

- Manages the GUI application
  - control flow
  - main settings
- One Qt GUI application:
  - precisely one QApplication object
  - not connected to the number of windows
- Used for QWidget based Qt applications
  - linked to QWidget, but different roles
- Does a lot of initialisation
  - must be created before any other objects related to the user interface

QApplication: responsibility

- Initializes the application
  - user's desktop settings
  - keeps track of properties
  - defines look and feel
- Performs event handling
  - receives events from the underlying window system
  - dispatches them to relevant widgets
- Knows about the application's windows
  - can ask it which widget is at a certain position
  - get lists of
- Manages mouse cursor handling

QApplication: some function groups

- System settings
  - example: setDoubleClickInterval()
- Event handling
  - example: processEvents()
  - example: sendEvent()
  - example: postEvent()
- GUI Styles
- Widgets
  - example: allWidgets()
  - example: focusWidget()

QWidget

- The base class of all user interface objects
  - many pre-defined subclasses
- The atom of the user interface
  - receiving events (mouse, keyboard, etc.)
  - paints a representation of itself on the screen
  - always rectangular
- Can have parent and children
  - not embedded in a parent widget: a window
- Composite widget
  - child widgets usually managed by a layout

QWidget: painting and size

- QWidget is a subclass of QPaintDevice
  - can display content composed in painting operations
  - painting with an instance of the QPainter class
  - each widget: all painting operations: paintEvent()
  - called whenever the widget needs to be redrawn
- Size
  - hints and policy
  - default when no size hint: sized by needs of child widgets

QWidget: events

- Widgets respond to events
  - typically caused by user actions
- Qt event delivery to widgets
  - calling event handler functions
  - include instances of QEvent subclasses
  - contains information about each event
- Event examples
  - paintEvent(): the widget needs to be repainted
  - resizeEvent(): the widget has been resized
  - mousePressEvent(): mouse button pressed while cursor inside widget
  - also: release mouse button, keyboard events, etc. etc.

Inheritance from standard classes

- There is more to a class than what is declared
- Read the documentation for an overview
  - http://doc.qt.io/qt-5/qapplication.html
  - http://doc.qt.io/qt-5/qwidget.html

Events and their travels: example

- Create a timer and connect it to a slot
  - timer=new QTimer(this);
  - connect(timer, SIGNAL(timeout()), this, SLOT(update()));
- Start the timer
  - timer->start(10);
- Every 10 ms:
  - update() is called, which triggers a
  - paintEvent
  - *The paintEvent is triggered by other events too*
  - Is called **at least** every 10 ms
  - Other calls are usually neglectable for frequent timers

Macros

- QObject is the heart of the Qt Object Model
  - All Qt widgets inherit QObject
- Used for seamless object communication
  - signals and slots
- connect()
  - connects a signal to a slot
- QObjects receive events through event() (+ children)
  - also provides a basic timer: QTimer
- *The Q_OBJECT macro is mandatory for any object that implements signals, slots or properties*
  - just use it in all subclasses of QObject

Aufgabe Woche

- Modifizieren Sie das den Kreis bewegende Programm so, dass am Anfang KEIN Kreis sichtbar wird.
- Es gibt einen zusätzlichen Button "Objekt". Jedesmal, wenn er gedrückt wird, entsteht ein neuer Kreis, vom zuletzt erzeugten um 10 Pixel nach rechts und 10 nach unten versetzt.
- Start / Stop wirkt auf *alle* Kreise.