

# Qt Development Tools



Developer  
Days  
2013

## GammaRay

Volker Krause  
[volker.krause@kdab.com](mailto:volker.krause@kdab.com)  
[@VolkerKrause](https://twitter.com/VolkerKrause)

# gdb is not enough

- Single-stepping through a scene graph?
- Debugging graphical glitches?
- Following asynchronous API?
- Keep a state chart in mind while looking at its generated code?

# printf isn't much better



Developer  
Days  
2013

- Generates huge amount of data
- No way of finding higher-level correlations
- Needs code changes

# We want more!

- Debugger should “understand” the frameworks we work with
  - Visualize state on the same conceptual level
- GammaRay

- QObject has runtime introspection support
  - list and edit properties
  - list and call slots
  - list and monitor signals
  - list existing signal/slot connections

# QObject



Developer  
Days  
2013

## Demo

- Powerful layouting framework for QWidget
  - box, form and grid layouts
  - margins and spacings
  - size policies
  - size hints
  - layout nesting
  - stretch factors



# QLayout

The image shows a screenshot of the Qt Creator IDE. The main window is the 'Options' dialog for the 'Analyzer' section, specifically the 'Valgrind' tab. The dialog is divided into several sections:

- Generic Settings:** Valgrind executable: valgrind
- Memory Analysis Options:**
  - Track origins of uninitialized memory
  - Backtrace frame count: 25
  - Suppression files: (empty list with Add... and Remove buttons)
- Profiling Options:**
  - Result view: Minimum event cost: 0.01%
  - Visualisation: Minimum event cost: 10.00%
  - Show additional information for events in tooltips
  - Enable cache simulation
  - Enable branch prediction simulation
  - Collect system call time
  - Collect global bus events

Overlaid on the right side of the dialog is the 'GammaRay (QtCreator)' window. It shows a search bar and a list of objects with their types. The 'Static Properties' tab is active, displaying a table of properties for the selected object.

Property	Value	Type
objectName	Valgrind__Inte...	QString
modal	false	bool
windowModality	0	Qt::WindowMo...
enabled	true	bool
geometry	0x0 905x574	QRect
frameGeometry	0x0 905x574	QRect
normalGeometry	0x0 0x0	QRect
x	0	int
y	0	int
pos	0x0	QPoint
frameSize	905x574	QSize
size	905x574	QSize
width	905	int
height	574	int
rect	0x0 905x574	QRect
childrenRect	4x4 675x283	QRect
childrenRegion		QRegion
sizePolicy	Preferred x Pre...	QSizePolicy
minimumSize	0x0	QSize
maximumSize	16777215x16...	QSize
minimumWidth	0	int
minimumHeight	0	int
maximumWidth	16777215	int
maximumHeight	16777215	int
sizeIncrement	0x0	QSize
baseSize	0x0	QSize
palette		QPalette
font	Sans Serif,9-1...	QFont
cursor		QCursor
mouseTracking	false	bool
isActiveWindow	false	bool
focusPolicy	0	Qt::FocusPolic...

The system taskbar at the bottom shows the following open applications: Akonadi Cons..., QtAssistant, Charm, Home - Gamm..., gammaray.g..., Mail - Kontak..., gammaray@..., Kopete, GammaRay (Q..., Qt Creator. The system clock shows 11:42.



- Comprehensive 2D painting API
  - various geometric drawing methods
  - countless fill and outline options
  - coordinate system transformations
  - often performance sensitive

# QPainter



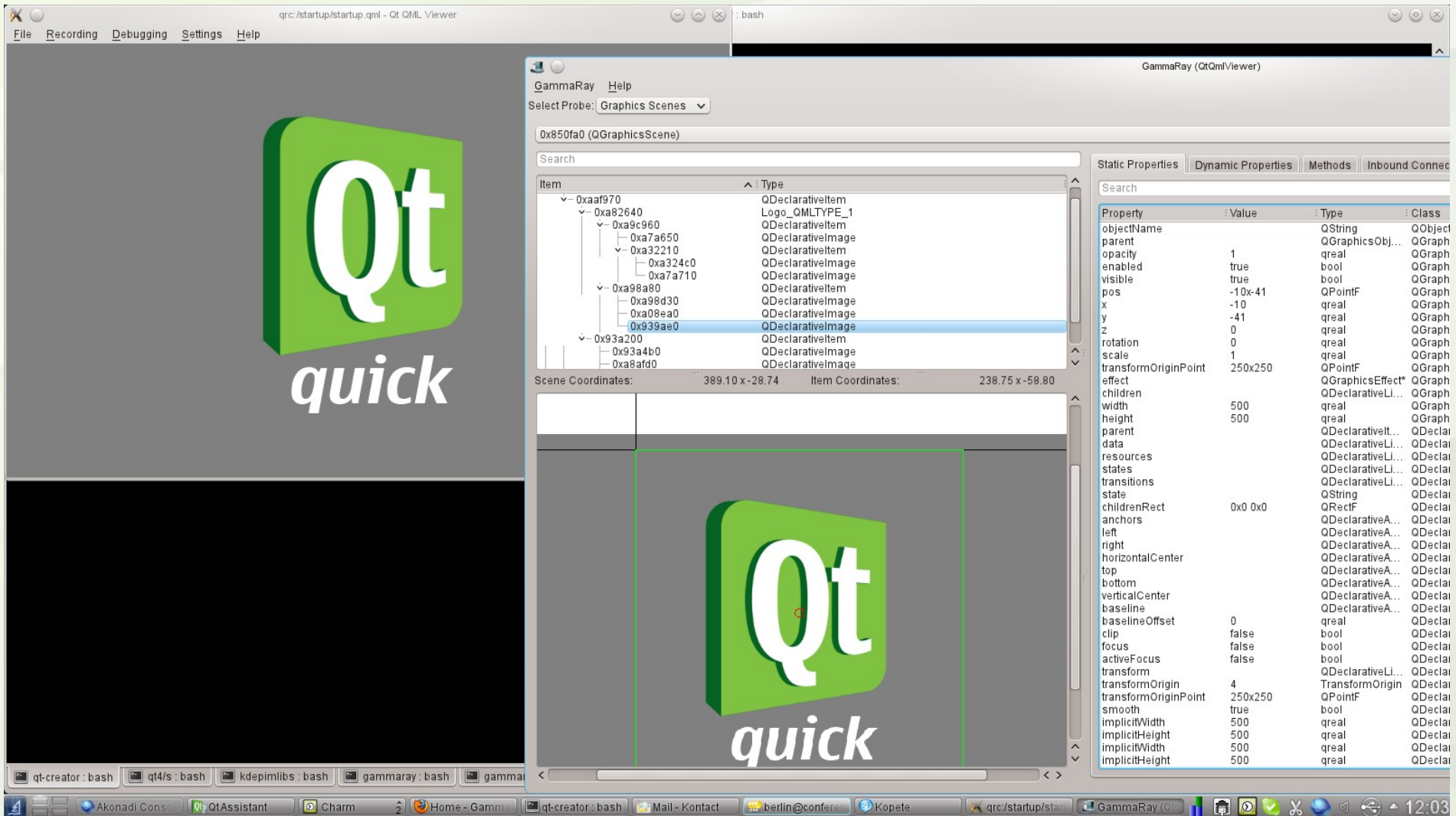
Developer  
Days  
2013

Demo

# QGraphicsView

- 2D visualization framework
  - scene graph with BSP index tree
  - tree of graphics items with nested transformations
  - graphics items can have complex shapes
  - level of detail support
  - ...

# QGraphicsView



The screenshot displays the Qt Creator IDE with a QGraphicsView widget. The widget contains a Qt logo and the text "quick". The QML Inspector is open, showing the scene graph for the QGraphicsView. The selected item is a QDeclarativeImage at memory address 0x939ae0. The Inspector shows the following properties:

Property	Value	Type	Class
objectName		QString	QObject
parent		QGraphicsObj...	QGraph
opacity	1	qreal	QGraph
enabled	true	bool	QGraph
visible	true	bool	QGraph
pos	-10x-41	QPointF	QGraph
x	-10	qreal	QGraph
y	-41	qreal	QGraph
z	0	qreal	QGraph
rotation	0	qreal	QGraph
scale	1	qreal	QGraph
transformOriginPoint	250x250	QPointF	QGraph
effect		QGraphicsEffect*	QGraph
children		QDeclarativeLi...	QGraph
width	500	qreal	QGraph
height	500	qreal	QGraph
parent		QDeclarativeIt...	QDeclar
data		QDeclarativeLi...	QDeclar
resources		QDeclarativeLi...	QDeclar
states		QDeclarativeLi...	QDeclar
transitions		QDeclarativeLi...	QDeclar
state		QString	QDeclar
childrenRect		QRectF	QDeclar
anchors		QDeclarativeA...	QDeclar
left		QDeclarativeA...	QDeclar
right		QDeclarativeA...	QDeclar
horizontalCenter		QDeclarativeA...	QDeclar
top		QDeclarativeA...	QDeclar
bottom		QDeclarativeA...	QDeclar
verticalCenter		QDeclarativeA...	QDeclar
baseline		QDeclarativeA...	QDeclar
baselineOffset	0	qreal	QDeclar
clip	false	bool	QDeclar
focus	false	bool	QDeclar
activeFocus	false	bool	QDeclar
transform		QDeclarativeLi...	QDeclar
transformOrigin	4	TransformOrigin	QDeclar
transformOriginPoint	250x250	QPointF	QDeclar
smooth	true	bool	QDeclar
implicitWidth	500	qreal	QDeclar
implicitHeight	500	qreal	QDeclar
implicitWidth	500	qreal	QDeclar
implicitHeight	500	qreal	QDeclar

# QStateMachine

- State machine framework
  - nested states
  - parallel states
  - history states
- Implementation is linear code







# And more!

- QTextDocument inspector
- (Proxy) model inspector
- Resource browser
- QStyle debugger
- Web inspector
- QScript debugger
- ...

# Where do I get it?

- <http://www.kdab.com/gammaray>
- Free Software (GPL)
- Code is on Github
- Works on Linux, Mac, Windows, QNX/BB10