

What's new in Qt 6 on the desktop?

Qt Desktop Days 2020



About me

- Senior Software Engineer, KDAB
- Developer & Trainer
- Qt Approver

Qt DESKTOP|DAYS

- Ask me about QtCore, QtGui, QtQuick, ...
 - And also about Modern C++,
 3D graphics



∄KDAB

The Road to Qt 6

Why Qt 6?

- Do architectural changes that simply cannot be done in Qt 5
- Binary compatibility break
 - Applications must be recompiled
- Re-engineer features
- But also do some necessary housecleaning, drop ballast



⊿KDAB

Design Goals

- Keep as much (source) compatibility with Qt 5 as possible
- Add property bindings in C++
- Improve QML & language bindings
 - Reduce overhead, increase type safety, compile to C++
- Tackle the changing landscape in 3D APIs
- Modularize Qt even more



Keep the Good Parts!

- Easy to use APIs
- General purpose, cross platform application framework
- Make 90% easy to achieve, and 99.9% possible
- Excellent developer support, documentation, tooling
- Nurture the ecosystem around Qt



nt Browser	Search		Properties M	ethods Conne	ctions QML Context	Enums C	lass Info QML Ty	pe
aphics Scenes			Search					
	Object	Туре						
cales	∽ ■ 🕫 0x1c865e0	QQuickRootItem	Property		Value		Type	Class
essages	0x1f0a190	QQuickOverlay	implicitHeight		54		double	00uickImplicitSi:
the Objects	stackView	OtOuick.Templates/StackView	text		Button presents a pus	h-button that c.	OString	OQuickText
ta Objects	> pane	QtQuick.Templates/Pane	font		Noto Sans.101.5.50.0	0.0.0.0	OFont	OOuidkText
ta Types	v page	ScrollablePage_QMLTYPE_3	color		#26282a		OColor	OOuickText
a Tunes	0x2a27620	QtuickItem	linkColor		#45a7d7		OColor	OOuickText
ic types	✓ ■ 0x2a72ca0	QtQuick/Flickable	style		Normal		TextStyle	ODuickText
dels	✓■ 0x2cb7d70	QQuickItem	styleColor		#000000		OColor	OQuickText
work	pane	QtQuick.Templates/Pane OtOuick/Column	borizontalAlia	nment	AlianHCenter		Halianment	QQUICKTEX
	T 0x2abb04	0 QtQuick.Templates/Label	offectiveliaria	nontal Alignment	AlianHConter		HAlignment	QuickText
ects	> 0x2093b1	0 QtQuick.Layouts/ColumnLayout	unitical Aliana	ont	AlignTon		VAlignment	OQuickText
	- 0x1f96620	QtQuick/Rectangle	verucarrigini	ren n	Wrap MWardBoundan	Ortemptorol	Wraphiete	QQUIKTER
	> ① / / / 0x1eca8a0	QtQuick.Templates/ToolBar	lineCount		a apaction of the output of the	on any where [int	QuickText
			Intecount		3 false		inc.	QuickText
k Scenes	Items Scene Graph		ouncated	C	1450		int	QuickText
	cm 2. 10 00 101 .1. 1.		macimumcine	Count	214/40304/		inc.	QQUICKTEXt
ources			text+ormat		AutoText		T extb-ormat	QQuickText
pt Engines	TeolBat on Put	Tool	etde		ElideNone		TextElideMode	QQuickText
ale			contentWidth		325.109375		double	QQuickText
1015	dutton presents a push-bu	tton that can be pushed or	contentHeigh		54		double	QQuickText
idard Paths	clicked by the user. Butt	ons are normally used to	paintedWidth		325.109375		double	QQuickText
	perform an action, or	to answer a question.	paintedHeigh		54		double	QQuickText
	6	1000	lineHeight		1		double	QQuickText
	Distance in the second	150	lineHeightMo	de	ProportionalHeight		LineHeightMode	QQuickText
t Codecs	Den en e		baseUrl		qrc/pages/ButtonPag	e.qml	QUH	QQuickText
		_200	minimumPixe	Size	12		int	QQuickText
	BUERA	ood EE	minimumPoin	tSize	12		int	QQuickText
ers		- 250	fontSizeMode		FixedSize		FontSizeMode	QQuickText
			renderType		QtRendering		RenderType	QQuickText
nstations	CLACK		hoveredLink				QString	QQuickText
gets	3		padding		0		double	QQuickText
4			L					

GammaRay Quick Scenes Settings Hel

Qt DESKTOP|DAYS

The Qt, OpenGL and C++ experts

Looking ahead

- Qt 4: released 2005, EOL 2015
 - ~30 modules
- Qt 5: released 2012, EOL 2023
 - ~50 modules

Qt DESKTOP

- Qt 6: released 2020, EOL 20??
- How to plan for the next decade?



Technical foundations

• C++17

DESKTOF

Qt

- MSVC 2019, GCC 8, Apple Clang
- CMake buildsystem for Qt
 - qmake still supported for end user applications
- 3D API abstraction (Qt RHI)



Release Plan

September 2020	October 2020	November 2020	December 2020
Alpha	Beta	Release Candidate	Qt 6.0 Final Release

- Qt 6.0 feature freeze reached
- Binary weekly snapshots (already) available via the installer
- Reduced set of modules available for 6.0 release
 - More coming back in 6.1/6.2, or via the Marketplace





Qt Core

Property bindings in C++

- The defining feature of QML, available in C++
- Make properties *depend* on other properties without manually writing slots and setting up connections

Mandatory note: actual syntax / feature set still evolving

Qt DESKTOP|DAYS

Property bindings in C++

- The defining feature of QML, available in C++
- Make properties *depend* on other properties without manually writing slots and setting up connections

Current bid: 50 EUR
Your offer: 55 EUR
\checkmark I accept the terms and conditions.
X <u>C</u> ancel <u>V</u> K

Mandatory note: actual syntax / feature set still evolving

Qt DESKTOP|DAYS

Property bindings in C++

• The defining feature of QML, available in C++

);

 Make properties *depend* on other properties without manually writing slots and setting up connections

Current bid: 50 EUR	
Your offer: 55 EUR	▲ ▼
\checkmark I accept the terms and conditions.	
≭ <u>C</u> ancel <u>√</u> <u>O</u> K	

okButton->enabled = Qt::makePropertyBinding(
 [&](){ return offerEdit->value > currentOffer
 && acceptBox->isChecked; }

Mandatory note: actual syntax / feature set still evolving

Qt DESKTOP|DAYS

∡KDAB

Qt container refactorings

- Support more than 2G elements
 - Index type is qsizetype
- Cleanups:
 - QList == QVector
 - QMap / QHash are single valued
 - QMultiMap / QMultiHash no longer inherit from QMap / QHash
 - New QHash, QMap, QPair, QVector implementations
 - QLinkedList dropped



String processing & I18N

- More flexibility in Qt string / byte array APIs
 - QStringView , QByteArrayView
 - QAnyStringView: views over UTF-8 / UTF-16 data
 - QStringTokenizer: non allocating string splitter
 - QStringRef / QRegExp superseded by QStringView / QRegularExpression
- Source code is UTF-8
- Support for Unicode 12, CLDR 36
- QtCore classes can now only deal with Unicode encodings
 - For other encodings: QTextCodec in Qt5Compat

Qt DESKTOP|DAYS

∄KDAB

Qt Widgets

Widgets

• No new widget class has been added to Qt since 4.x!

What's going on?



⊿KDAB

Widgets: focus on stability

- Widgets are essentially frozen
 - Focus on long term stability, bugfixing, small API improvements
- Entry barrier for new features (esp. new widgets!) extremely high
- Qt Marketplace, KDE Frameworks, etc. offer many extra goodies
- Contributions welcome!





KDDockWidgets

🔳 funho	ouse				- □ >				
<u>F</u> ile <u>V</u> ie	w Alpha	Beta	Charlie						
Style: P					Jse style's standard palette Disable widge				
Group 1					Group 2				
• Rad	lio button 1								
Tri-	iio button 2 state check h	ox			Default Push Button				
✓ Che	ck box				Flat Push Button				
					Toggle Push Button				
Table	Text Edit				✔ Group 3				
					•••••				
1					Normal line edit gjpqy Ll1				
2					40 🗘				
3					4/2/2019 3:40 AM 🌲				
4									
•					Tool Button Auto Raise 💻				
				40					

PhantomStyle

Still: keep up with new platform trends

- High-DPI in hybrid screen configurations must "just work"
 - Incl. fractional factors
 - May require usage of non-native styles
- New native styles, easier theming
 - E.g. dark mode / themes support on all OSes
- Handle hybrid input systems
 - Mouse + Stylus + Touch + Voice + Virtual Keyboard + CIM + Eye + ...
- Accelerate widget rendering & compositing through RHI
 - And/or platform native APIs

Qt DESKTOP|DAYS



QML and Qt Quick

QML: new features in Qt 5.15

- required properties
- Inline components
- Registration of classes and objects from C++ to QML at build time
 - Enables checks on usage points
 - Improves tooling

DESKTOP

qmake only in Qt 5.15, CMake support in Qt 6

```
class RadioController : public QObject
{
    Q_OBJECT
    QML_ELEMENT
    // ...
};
CONFIG += qmltypes
QML_IMPORT_NAME = com.kdab.carstuff
QML_IMPORT_MAJOR_VERSION = 1
```

AKDAB

Towards a better QML

- QML 2 has had a great run
- Still lots of room for improvement for big-scale (desktop) applications
- Lessons learned from QML 2:
 - Precompile as much as possible
 - JavaScript is fun, until it isn't
 - More build system / tooling support
 - Do more static analysis



Coming soon (6.1/6.2): QML 3

- JavaScript engine optional
- More static typing

DESKTOP

Qt

- QML compiled to C++ code
 - By exploiting the new property system, new meta object, etc.

```
import QtQuick 2
Item {
    property bool active: checkBox.checked
}
class MyItem : public QQuickItem
{
    public:
        QProperty<bool> active;
};
```

AKDAB

∄KDAB

Qt Quick

- API of elements is mostly unchanged
- Unified 2D/3D scenegraph
 - Tighter integration for 3D content
- Rendering happens exclusively through RHI
 - Targeting OpenGL (ES), Direct3D, Vulkan, Metal
 - Software rendering still available
 - Desktop applications mixing raw 3D and Qt Quick possibly affected
- Planned in Qt 6.x: C++ API for Qt Quick elements





Qt Quick Controls 2: Native Desktop Style

- Not every Qt Quick application wants a custom look and feel
- In Qt 6.0: native desktop style for Qt Quick Controls 2

				Desktop	Gallery			
		Default cor	ntrols			Customized	controls	
But	tons							
[Default	Disabled	Small	Mini				
Exp	plicit height	Exp	olicit width					
Hiç	ghlighted	Flat	Checkable					
Che	eckBoxes							
1	Default	Disabled	Tri-state	Small	✓ Mini			
Rad	dioButtons							
۲	Default	Disabled	🔘 Small	O Mini				
0	Default	Disabled	🔘 Small	Mini				
0	Default	Disabled	 Small 	🔿 Mini				
Spii	nBoxes							
	503	*	0	5	*	0		
Text	tFields							
De	fault	Disabled	k	Placeholder te	ext Small		Mini	



∄KDAB

3D Graphics

Qt and 3D graphics: a bit of history

- OpenGL, OpenGL, OpenGL
 - As a cross-platform toolkit, Qt has always had deep OpenGL integration
 - OpenGL widgets, QOpenGLContext, OpenGL helper classes, etc.
- Qt 5 bet on OpenGL (ES) as the universal enabler API for 3D
 - OpenGL first-class citizen in Qt APIs
 - QtGui featuring OpenGL classes
 - QtWidgets for OpenGL content

DESKTOF

- Qt Quick uses OpenGL for rendering



3D graphics today

- The OpenGL bet didn't quite pay off
- Today's world: multiple competing 3D standards
 - OpenGL
 - Vulkan
 - Metal
 - Direct3D
- Some support for Vulkan and Metal already available in Qt 5





Qt 6 strategy for 3D graphics

- Move everything but the core 3D classes out of QtGui, into their own libraries
 - Lots of OpenGL goodies still available
- Introduce RHI as foundational 3D API for Qt itself

Qt DESKTOP

- Qt Quick, Qt 3D, etc. using it in 6.0



AKDAB

⊿KDAB

3D: summary

- Qt RHI initially (mostly) private API, for Qt's own libraries
 - Qt Quick, Qt Quick 3D, Qt 3D, etc.
- OpenGL enablers and higher-level classes are still available to applications
 - Some simply got moved; minor changes required
- New: low-level enablers for other 3D APIs, available in Qt Gui
- Application mixing raw 3D with Qt-rendered 3D may need some changes





How to upgrade to Qt 6?

∄KDAB

The way towards Qt 6

- Recognize the success of Qt 5
 - Qt 5.15 has *Long Term Support* (3 years)
- Minimize disruptions for end-users towards Qt 6
- Not every Qt 5 module will be ready for 6.0
 - Notable desktop-related modules missing: QtWebEngine, QtVirtualKeyboard, QtMultimedia
 - Some of those may move to the Marketplace



.≇KDAB

Planning the upgrade

- Upgrade your software to Qt 5.15 LTS
- Upgrade your toolchain
- Enable the deprecation warnings in Qt, fix them all
- Use docs, changelogs, etc. to identify pain areas ("Important Behavior Changes")
 - Have a plan to address them before getting your hands dirty



Migrate, not upgrade

- Upgrading towards Qt 6 is still a migration
 - Therefore: avoid adding features, doing refactorings, etc. while porting
 - If needed, port to Qt 5.15 features *before* porting to Qt 6
- In theory, have enough #ifdefs and/or build system support to keep your code work on both 5.15 and 6.x
- Some unsupported features moved to Qt5Compat module
 - Use it (temporarily!) to ease the transition





Thank you!

Questions?