Build your world with Qt

Qt Roadshow North America

April/May 2015

Lars Knoll

Qt Chief Maintainer
CTO, The Qt Company

What is Qt?

Everything you need to develop your application or device

Requirements change quickly.

Qt keeps you one step ahead.
Adapting to your strategy.

Qt The Qt Company

Your product on all screens.

All targets. Reuse & scale.

Qt The Qt Company

Qt in use

Tesla
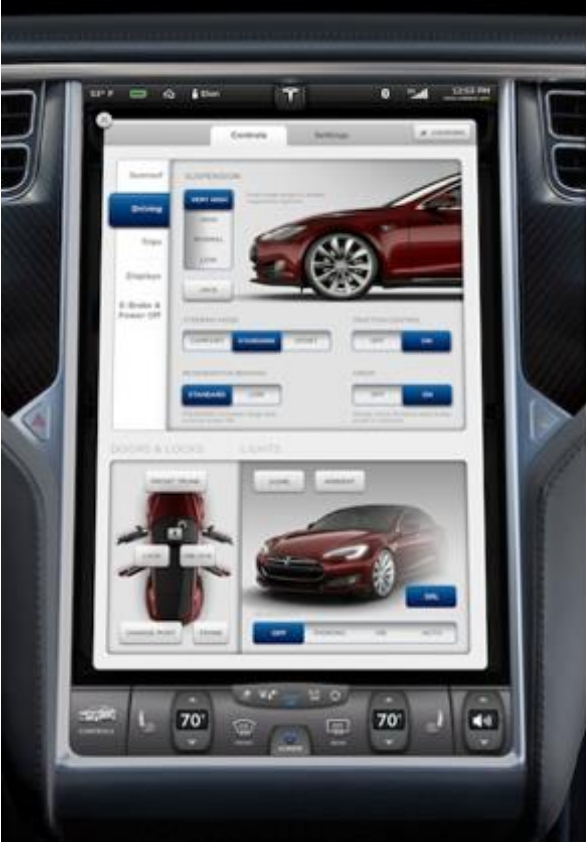
Air traffic control systems

In flight entertainment

Cross platform functionality

Non graphical APIs

- Provides extensive APIs to access low level functionality
  - All APIs are cross platform
- Functionality provided
  - Threading
  - File IO, Networking, SSL, HTTP
  - Shared memory, IPC
  - Internationalization, Unicode support
  - Event handling
  - Signals/Slots
  - Bluetooth
  - Etc…

Qt Creator and Tooling

Qt Creator

- Qt Creator is a full development environment that supports the whole development life-cycle
  - Drag'n'drop UI designer, with separated UI presentation and UI logic
  - Powerful code editor
  - Direct deployment to multiple targets
  - On-device debugging and profiling

game - Qt Creator

File Edit Build Debug Tools Window Help

Qt
Welcome

Edit

Debug

Projects

Help

Output

Build

Search

7 found

Projects

- game
  - game.config
  - game.files
  - game.include
  - ..
  - bsp.c
  - bsp.h
  - game.h
  - main.c
  - mine1.c
  - mine2.c
  - missile.c
 - qpn_port.h
  - ship.c
  - tunnel.c
  - video.c
  - video.h

File System

game

- game.creator
- game.creator.user
- game.files
- game.h
- main.c
- mine1.c
- mine2.c
- missile.c
- qpn_port.h
- ship.c
- tunnel.c
- video.c
- video.h

game.files   Line: 16, Col: 26

```
bsp.c
bsp.h
game.h
main.c
mine1.c
mine2.c
missile.c
qpn_port.h
ship.c
tunnel.c
video.c
video.h
../../../../include/qassert.h
../../../../include/qepn.h
../../../../include/qfn.h
../../../../include/qkn.h
```

missile.c   Missile_flying(Missile *)   Line: 89, Col: 32

```
/*.............................................
QState Missile_flyi
    switch (Q_SIG(m
        case TIME_T
            if (me-                              EEN_WIDTH) {
                me-
                /*t                              e Missile and test for wall hit
                QAC                              Tunnel, MISSILE_IMG_SIG,
                                                 E_BMP << 16)

                                                 / << 8));
            }
            else {                               nge, make it ready to fire agai
                ret                              );
            }
            return
        }
        case HIT_WA
            return                              );
        }
        case DESTRO
    }
    return Q_SUPER(
}
/*...........................................
QState Missile_exploding(Missile *me) {
    switch (Q_SIG(me)) {
        case Q_ENTRY_SIG: {
            me->exp_ctr = 0;
            return Q_HANDLED();
        }
        case TIME_TICK_SIG: {
```

Select Symbol dropdown:
- <Select Symbol>
- MissileTag
  - super: QActive
  - speed: uint8_t
  - x: uint8_t
  - y: uint8_t
  - exp_ctr: uint8_t
- Missile: MissileTag
- Missile_initial(Missile *)
- Missile_armed(Missile *)
- Missile_flying(Missile *)
- Missile_exploding(Missile *)
- AO_Missile: Missile
- Missile_ctor(uint8_t)
- Missile_initial(Missile *)
- Missile_armed(Missile *)
- Missile_flying(Missile *)
- Missile_exploding(Missile *)

Search Results

```
92    me->mine2_pool[n] = Mine2_ctor(n); /* instantiate Mine2 in the pool */
93    me->mines[n] = (QHsm *)0;                  /* mine 'n' is unused */
93    me->mines[n] = (QHsm *)0;                  /* mine 'n' is unused */
95    me->last_mine_x = 0;   /* the last mine at the right edge of the tunnel */
95    me->last_mine_x = 0;   /* the last mine at the right edge of the tunnel */
96    me->last_mine_y = 0;
103   for (n = 0; n < GAME_MINES_MAX; ++n) {
104       QHsm_init(me->mine1_pool[n]);          /* initial tran. for Mine1 */
```

Type to locate   1 Build Issues   2 Search Results   3 Application Output   4 Compile Output

Creating great user experiences

UI technologies

- Classic desktop look-and-feel
  - **Qt Widgets**
- Fluent, modern touch-based UX
  - **Qt Quick**
- 3D becomes more and more important
  - **Qt OpenGL** has provided a great OpenGL integration for many years
  - Introducing **Qt Canvas 3D** and **Qt 3D** with Qt 5.5
    - Greatly simplifies integration and creation of 3D content
- Integrate HTML5 efficiently with **Qt WebEngine**
- Real-time multimedia and camera features with **Qt Multimedia**

Qt Widgets

- Create classic desktop applications
- Full set of widgets
  - From simple buttons to complex unicode capable text edits
  - Menus, main window, toolbars, dock areas etc.
- Styling abstraction
  - Fully cross platform code base
  - Native look&feel on all Desktop platforms

Qt Quick
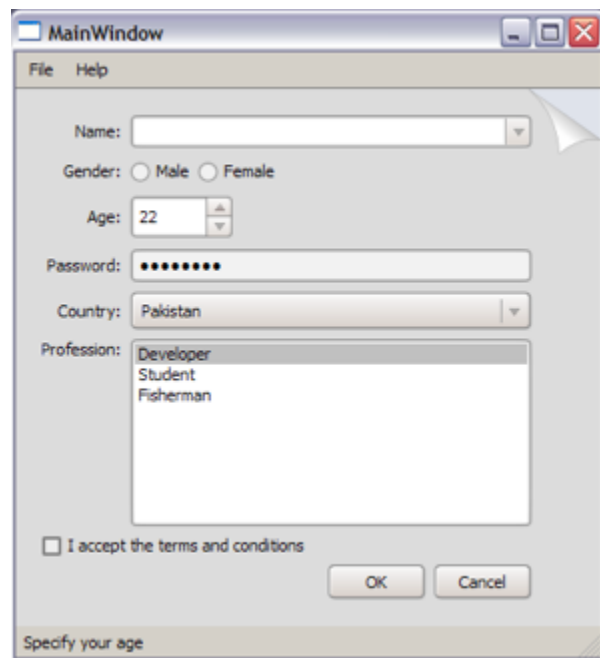
- ## Create modern fluid user interfaces
  - QML UI description language
  - Animations, States and Transitions
  - OpenGL based scene graph
    - Optionally also supported with pure 2D acceleration
  - Graphical effects

- ## Tightly cooperate with UX designers
  - UX designers can directly participate in the development
  - Direct feedback and testing of the developed user experience

OpenGL integration

- ## Qt OpenGL library
  - Provides all required low level enablers to integrate OpenGL content
  - Simplifies dealing with different GL versions
  - Abstractions for GL context, FBOs etc
  - 2D drawing on top of OpenGL
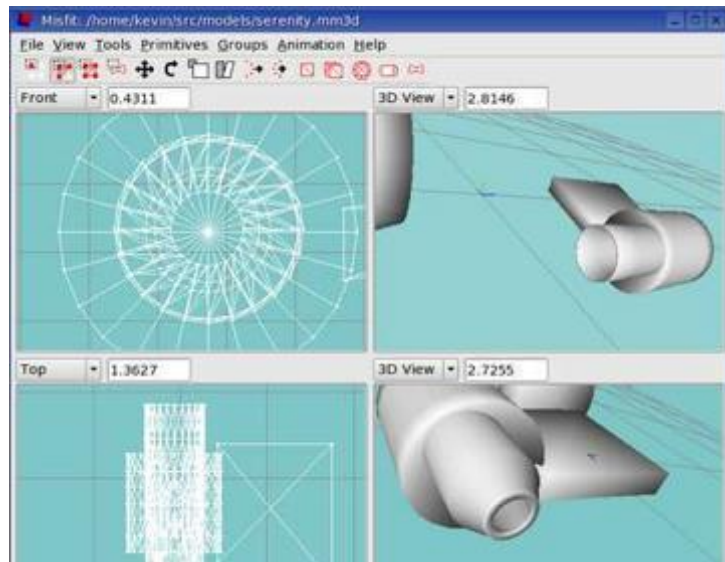
Qt Canvas 3D – Fully Supported with Qt 5.5!

- QML canvas element implementing WebGL-like API

- Enables 3D app development using JavaScript

- Very powerful when combined with a 3D JavaScript Library
  - Three.js provides a great set of functionality and support for common 3D object formats

- Compact implementation

- 3D with low amount of dependencies

- Suitable for: simple visualizations & controls, performance constrains, small footprint

Qt 3D – First Technology Preview with Qt 5.5!

- 3D engine specifically tailored for Qt/QML application development

- Higher level abstraction for generating interactive 3D content

- Multithreaded and extensible architecture

- Native code performance

- 3D Object loaders for popular formats (based on Assimp library, feature support varies)

- Suitable for: From simple uses cases to complicated, larger footprint than with Qt Canvas 3D

- Developed by KDAB, a Qt Partner, together with The Qt Company

Qt WebEngine

- HTML5 is an important technology for various industries
- Qt WebEngine is a very powerful engine for HTML5 content
  - Based on Chromium project
  - Easy to use, high abstraction level
  - Cross-Platform
- Full integration to Qt graphics stack
  - Qt Widgets
  - Integration into the Qt Quick scenegraph
- Interface with native code through Qt WebChannel
- Qt WebEngine is fully supported for device creation and on all desktop platforms

Some numbers

HTML5Test.com (on Mac OS X)

- Qt WebEngine   506 points
- Google Chrome   512 points
- Safari 7.1   405 points

SunSpider

- Qt WebEngine: 139.0ms +/- 1.7%
- Google Chrome: 135.9ms +/- 1.4%
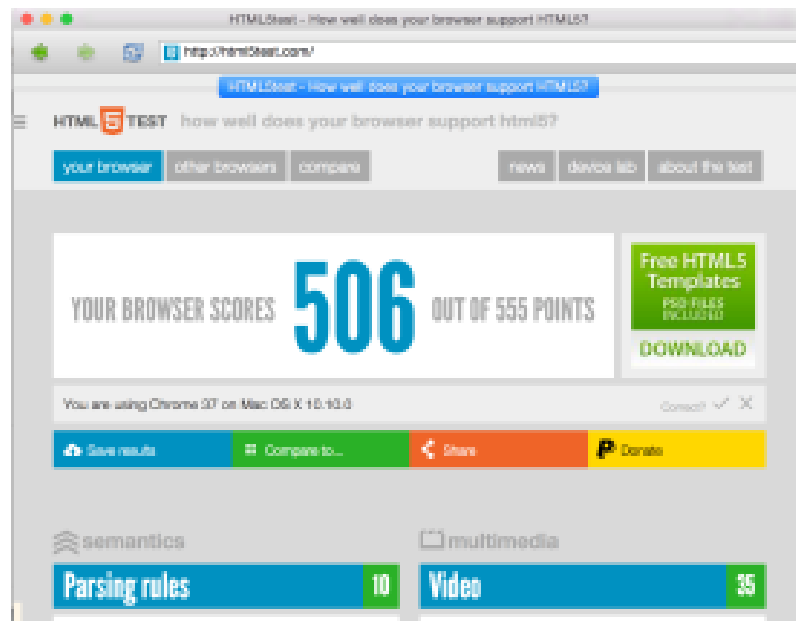
Improved Qt Multimedia

- Industries require high-performing UI with integrated multimedia or live camera
  - Machine vision, robotics, image and signal processing, bioinformatics etc.
- Makes it trivial to get a video stream from the camera or a video file rendered as part of your application
- Simple integration with external graphics libraries, like OpenCL or CUDA with Qt 5.5
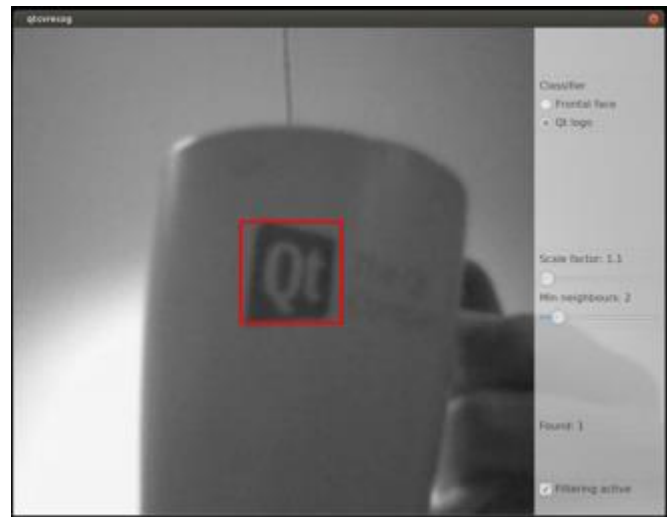
Qt Helps You Connect Applications and Devices

Bluetooth and Bluetooth Low Energy

- Qt Bluetooth provides cross-platform API for Bluetooth functionality

- Bluetooth Low Energy (BtLE) is an important way for IoT smart sensors and devices to communicate
  - Also used with wearables
  - Now fully supported through Qt Bluetooth, also on Android and iOS

Qt Location – Technology Preview available with Qt 5.5

- With connectivity, location-based services and mash-ups are becoming a vital part of many IoT systems
- Qt Location module allows integration of
  - Maps
  - (Reverse) geocoding
  - Navigation
- Can be used with different backends, through plugins
  - Comes with Nokia HERE and Open Street Map plugins
  - Can be extended to your own
- Together with Qt Positioning API (GPS/wifi/text location) provides all location-related services through Qt APIs

Qt Map Viewer Example

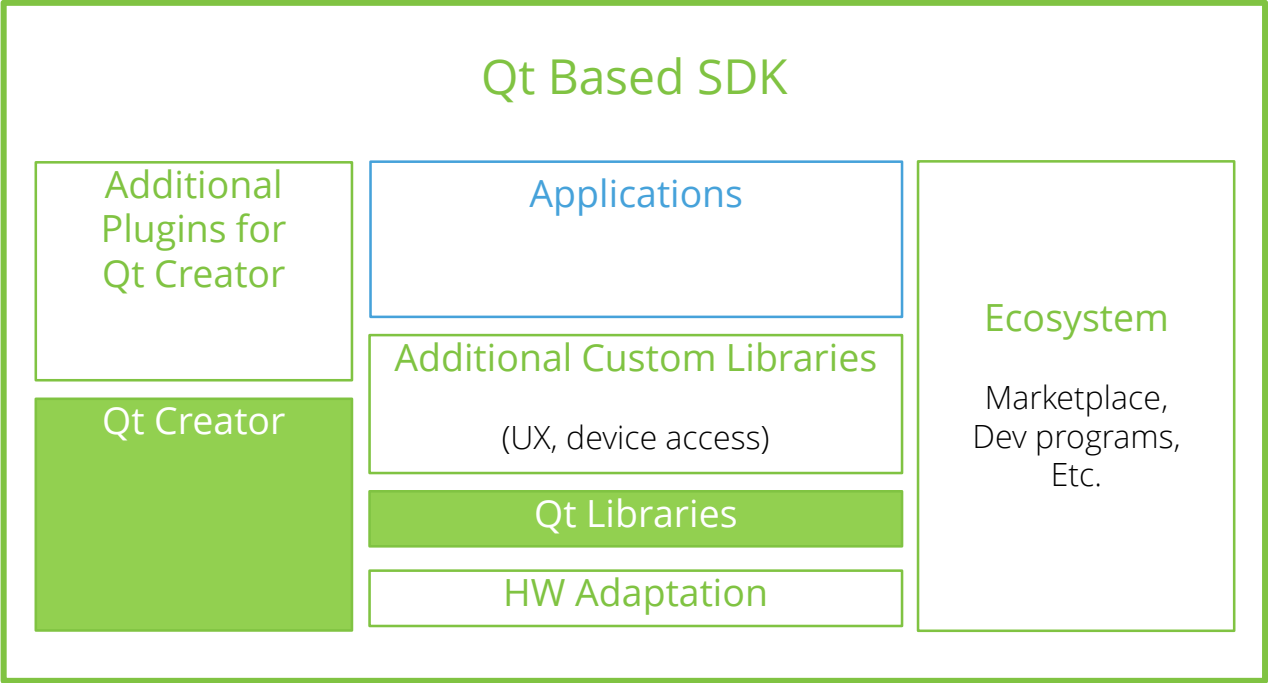# Ultimate Performance, Reliability and Stability

Ultimate Performance, Reliability and Stability

- Core power of Qt comes from the C++ foundation
  - Modern UX without compromising the native performance
  - Full control over the code
  - Mature code APIs, prooven-to-work over the years

- Qt 5.5 brings even better standards of quality for the core Qt libraries

- Improved feature parity between platforms ensuring full cross-platform compliance

- Qt 5.5 is a solid foundation for long-lasting software projects

Create Your Unique Platform

Qt as Complete Technology Platform

Qt Based SDK

Additional Plugins for Qt Creator	Applications	Ecosystem
Qt Creator	Additional Custom Libraries (UX, device access)	Marketplace, Dev programs, Etc.
	Qt Libraries	
	HW Adaptation	

Other new Features in Qt 5.5

Windows Store Apps and Windows 10

- ## Full support for Windows store apps
  - Windows 8.1 and newer
  - Windows Phone 8.1 and newer
- ## Full support for Windows 10
  - Classical desktop apps work out of the box
  - Windows store app support coming with Windows 10 release
- ## Integration into Qt Creator
  - Package creation, deployment and on device testing

Qt Quick

- Typed array support
- Improved C++ JS integration bridge
  - Improve QML/JS ←→ C++ integration
- Performance improvements
  - Faster startup
  - Reduced memory consumption
- Many bugfixing and other improvements
- TreeView for Qt Quick Controls
- Enterprise Controls folded into Qt Quick Controls
  - Now available in the open source version
- Larger improvements for Qt Quick Controls planned for Qt 5.6
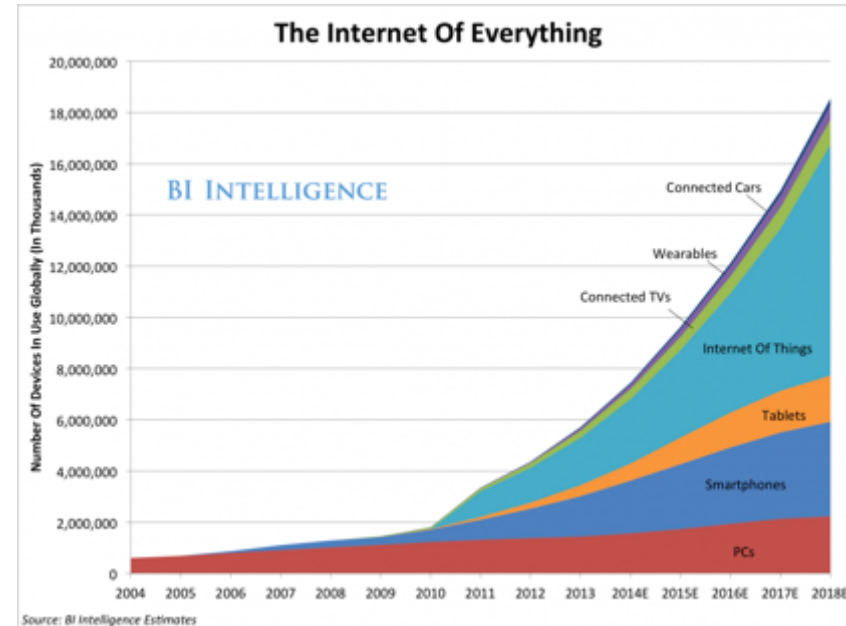 - Optimized for embedded

Qt Creator 3.4

- More refactoring features
- Easy use of automated tests, built with Qt Test and Qt Quick Test modules
- Improved debugging possibilities, eg. Valgrind memory analyzer
- Improved C++11 support
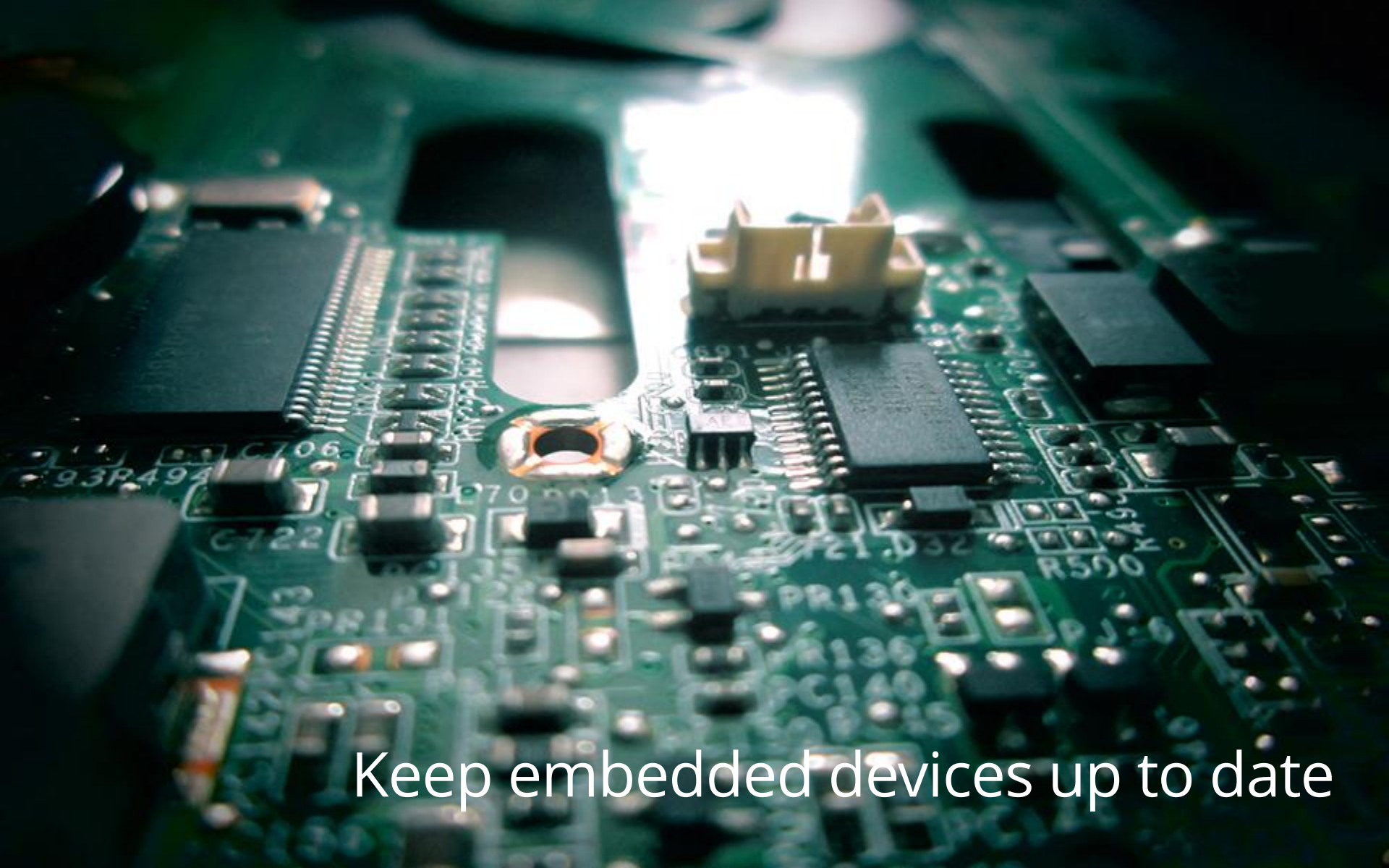- Support for 64bit Android

Outlook

Connected Devices

- More and more devices get touch screens
- Need a simple to configure, flexible stack
- Machine to Machine (M2M)
  - Common data protocols
  - Discovery
- Issues
  - Data processing and analytics
  - Security, access control
  - Software updates
  - Data control and sharing

The Internet Of Everything

BI INTELLIGENCE

Connected Cars
Wearables
Connected TVs
Internet Of Things
Tablets
Smartphones
PCs

Number Of Devices In Use Globally (In Thousands)

20,000,000
18,000,000
16,000,000
14,000,000
12,000,000
10,000,000
8,000,000
6,000,000
4,000,000
2,000,000
0

2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014E 2015E 2016E 2017E 2018E

Source: BI Intelligence Estimates

Make device creation
faster and easier

Keep embedded devices up to date

Integrate UX designers

How to build your solution with Qt?

Example: Logistics

Available products

Qt for application development

- Comprehensive set of C++ frameworks
- Qt Quick user interface definition language
- Integrated tooling
- Cross platform

Qt for device creation

- Everything you get with Qt for application development
- Reference stacks based on Yocto
- Build your own stack tooling
- Device simulator
- Integrated device deployment and debugging

Dual licensing

- Dual licensed
    - Open Source (LGPLv2.1 and/or LGPLv3)
    - Commercial licensing available from **The Qt Company**
    - Ensures future availability of the asset and thus secures your investment

- Large part developed as OSS project
    - **But:** 80% of development done by The Qt Company
    - You can contribute your favorite feature to Qt

Schedule

Latest stable releases:

Qt 5.4.1 & Qt Creator 3.4

Qt 5.5 beta   expected next week

Qt 5.5 final   June 2015

Qt 5.6   November 2015

Thank You!

lars.knoll@theqtcompany.com

www.qt.io