Meet Qt Framework

The Qt Company
June Joe
Business Development Lead
june.joe@qt.io

Sep 2021

- ✓ Growing and innovating from day one.
- ✓ Mature, Robust & Verified in the market over 25 Years

1995
First public version of Qt released by Trolltech (Norway)

2008
Trolltech acquired by Nokia

2012
Digia acquired "All rights to Qt" from Nokia

2014
The Qt Company becomes its own entity in Digia

2016
Qt Group Plc demerger from Digia, listed **independently** in NASDAQ Helsinki

2017
Qt Group Plc raised EUR 15 Million

2019
+1.5M developers have used Qt creating a large developer ecosystem

2020
the strategy; **year 2021 revenue 100M€**

Qt Releases

1.0	1995
2.0	1999
3.0	2001
4.0	2005
5.0	2012
6.0	**2021**

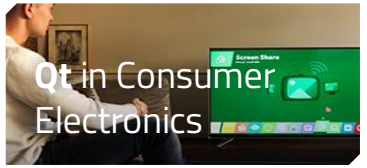
The Qt Company and Qt

Revenue	Growth YoY	Employees
90 M$	30 %	400

R&D Investment	Market Cap.
500+ M$	1.2+B$

Qt [kju:t] is a **Device Creation Framework** for designers and developers in creating next generation User Experiences with true hardware and operating system agnostic

- **Very Widely Used in 70+ industries / Global Ecosystem** – Millions of Downloads Annually



5000+ companies (>400 in Korea)
>3M Developers Worldwide
High-end to Low-end devices

- **Strong Partner Network** -100s+ of service companies around the world and all major SoC vendors have Qt readiness

QT FRAMEWORK VALUE PROPOSITION

What is Qt Framework ?

Application / Devices

Supported Languages

- C++
- Python
- Java Script
- HTML5
- Qt QML

Framework

- Qt Essential Libraries
- Qt Add-on Libraries

Target Platforms

- Desktop
- Embedded MCU/RTOS
- Mobile

Designer Tool

- Wireframe
- Interaction
- Prototyping
- Live preview on Target

Developer Tool

- Development
- Integrated Dev Environments
- Embedded SW Development

SW Testing & Verification Tool

- Squish
- CoCo

Qt

Qt Framework

“There is a screen, There is a Qt”

Full cross-platform application development framework with tools designed to streamline the creation of applications & UIs for desktop, embedded and mobile.

BUILT WITH QT

2019 CUSTOMER SURVEY

95%
ROI expectations exceeded

70%
find Qt easy to use

80%+
are more productive with Qt

Qt

NOT JUST A FANCY DESIGN BUT OFFERING VALUE

What does touch screen GUI offer ?

Scalability

Application

Brand Identity

Easy & fun to use

Connectivity

The Brave New Face brings an innovation of

- ✓ *Brand Identity*
- ✓ *User Friendly to MZ-generation*
- ✓ *Connectivity*
- ✓ *Application with Abundant Features & Update*
- ✓ *Easy, Fun & "Up-to-date" to use*
- ✓ *Better UI/UX*
- ✓ *& Value when all these combined together*

https://www.machinery.co.uk/machinery-features/new-machine-tool-controls

Embedded Product Planning Requirments

HW / SW Platform

- Affect / Consider BOM
- Development Maturity
- Asset & Knowhow
- Technical Enablement
- Component Availability

Reusability Scalability

- Multi generation propagation
- New Technology
  Plug-in
- Product Evolution

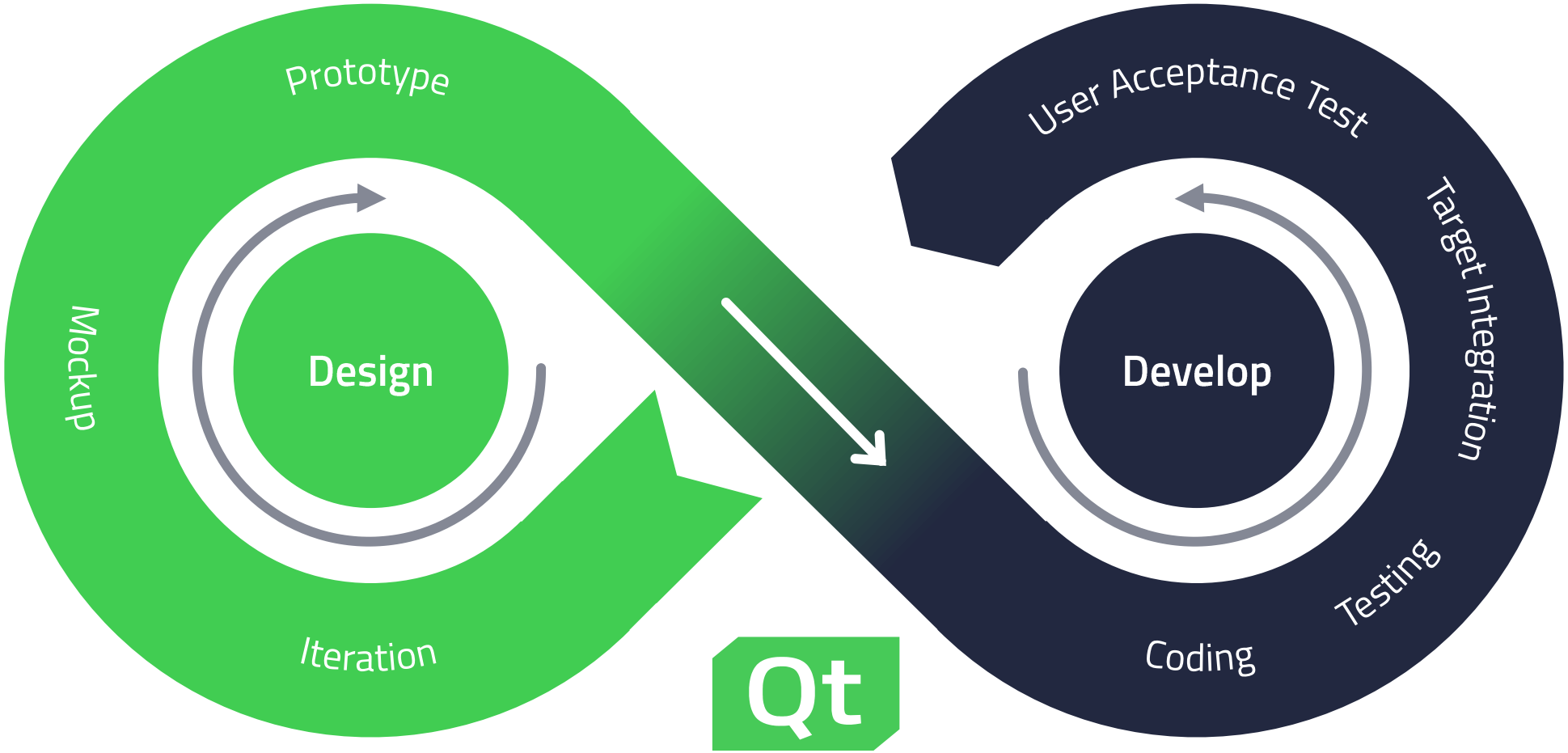
Design-Develop Work Flow

- Iteration time & GTM
- Product Level
- Real Brand Identity

Maintenance

- Avoid solution binding
- Focus on value creation

Enhanced Product Develpoment Workflow Needed



Code Once & Deploy Everywhere with same Qt source code

Embedded for Automotive & non-Automotive

- webOS, Embedded Linux, Android, Windows Embedded
- RTOS: QNX, VxWorks, INTEGRITY

Desktop

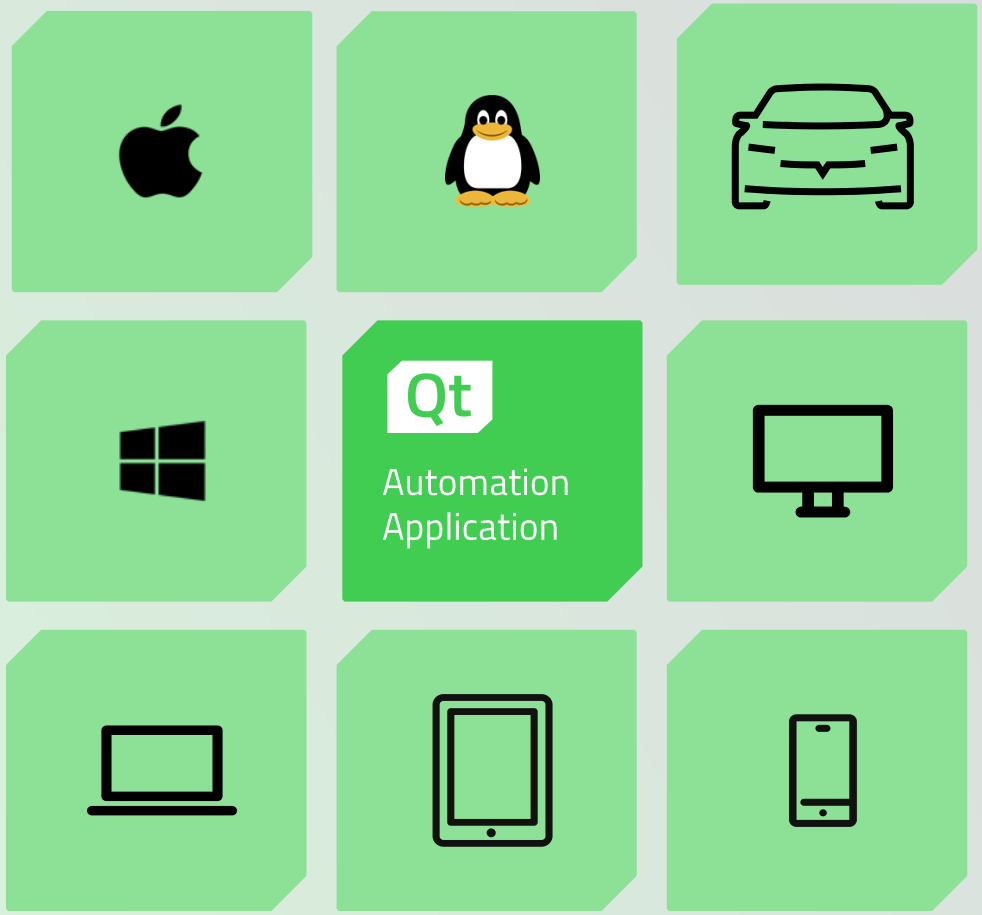
- Windows, Mac OS, Linux
- Linux Distributions(Suse, Redhat, Debian, Ubuntu & etc.), Enterprise UNIX

Mobile

- Android, iOS

Microcontroller(Automotive & non-Automotive)

- Ambiq, NXP, STM32, Renesas, Cypress
- Other Platforms on demand Demand



More Than a Collection of Libraries

- **Frameworks development projects**
  - Consistent APIs and documentation
  - Structure
  - Best practices – Frameworks provide proven solutions
  - Guides how to do things – can be extended

- **Frameworks come with a toolbox**
  - IDE, toolchains, etc.
  - Graphical and design tooling
  - Make it easy to apply best-practices

- **A good framework drives structure and consistency when thousands of engineers working with millions lines of code**

"Collection of libraries"

Kanzi, EB-Guide,
Crank & studio tools

VS

Framework

Collaborative Automotive Development Platform

- Whole development platform offering including design component creation tool to development including connectivity.
- Massive computing platform evolution with autonomous driving + ADAS enabler requires enormous investment, which Qt already offers as of today.

OEM Design Asset Creation

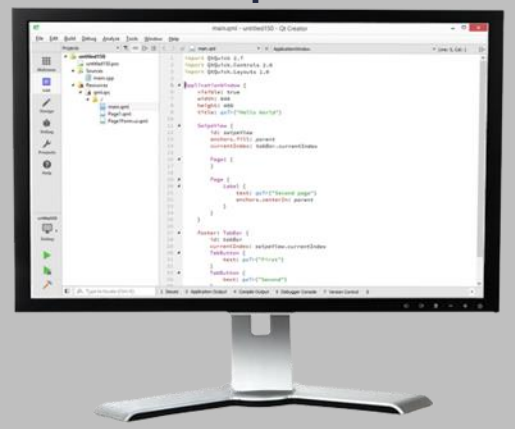
Interaction Designers

UI flow & navigation
Wireframes

Sensorial Designers

Visual assets
Motion designs
Audio assets

Application Development



Application Developers

Custom UI components
Data connections
Back-end logic

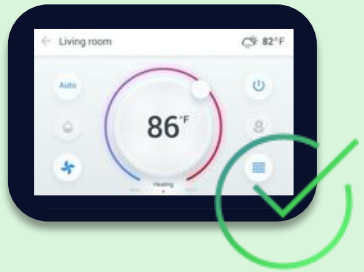
Interactive

Deploy

Testers & Management

Rapid iterations
Visuals review
Performance / UX testing

Maintain

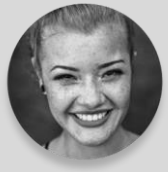


Cross Platform Maintenance

Desktop / Embedded / Mobile
MCU up to High Spec HW

Qt style workflow

Design



Visual Designer

Interaction Designer

DS

Design and implement pixel-perfect UIs immediately usable for developers

Validate designs
Prototype

Develop

Developer

QC

Integrate up-to-date designs and focus on back-end and application logic development

Test
Deploy

Deploy

Enhanced the work-flow and rapid development

"

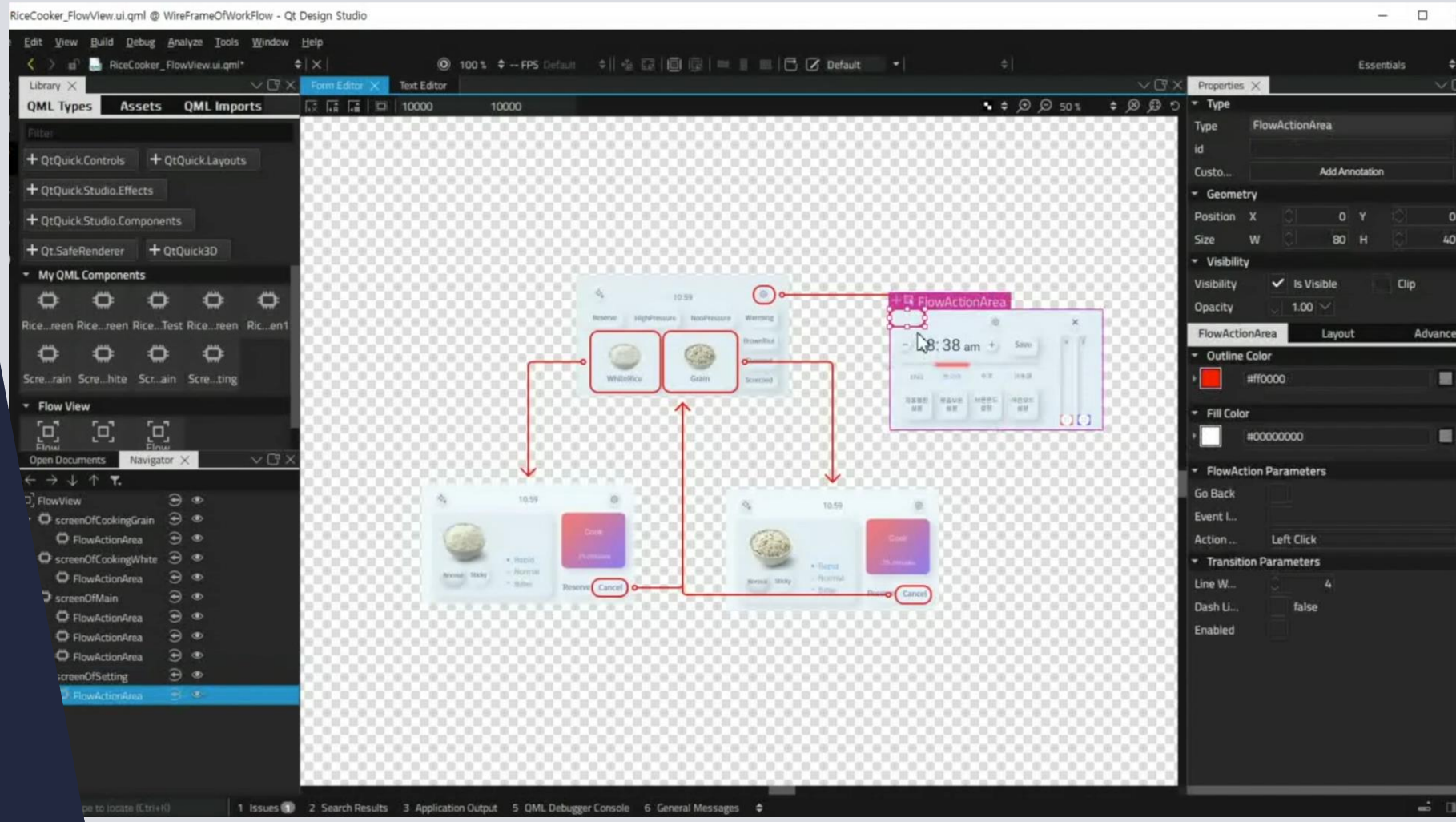
'I don't think everyone knows how powerful Qt's libraries are.

Our development process is much easier than before.

It can easily take 1000 to 1300 hours to develop a big automation system. Thanks to Qt, this project has been reduced to 50 hours.'

"

Rune Volden, R&D Manager, Ulstein



KEY VALUES

Qt for Device Creation

– The design/developer accelerator

Target for Embedded Development

- Yocto Support
- Webengine & virtual Keyboard supported
- Design Tools & IDE
- +1600 C++ library

Cross Platform Support

- Multi Platform & HW Support: Cmake Build
- Scalable solution from high-end MPU to low-end MCU

Graphics Toolkit for premium GUI

- Productive GUI creation using QML
- 2D / 3D Supported
- Performant ready-made Controls
- Latest Graphics Backend support Vulkan, Metal, Direct3D

Productive Rapid Prototyping

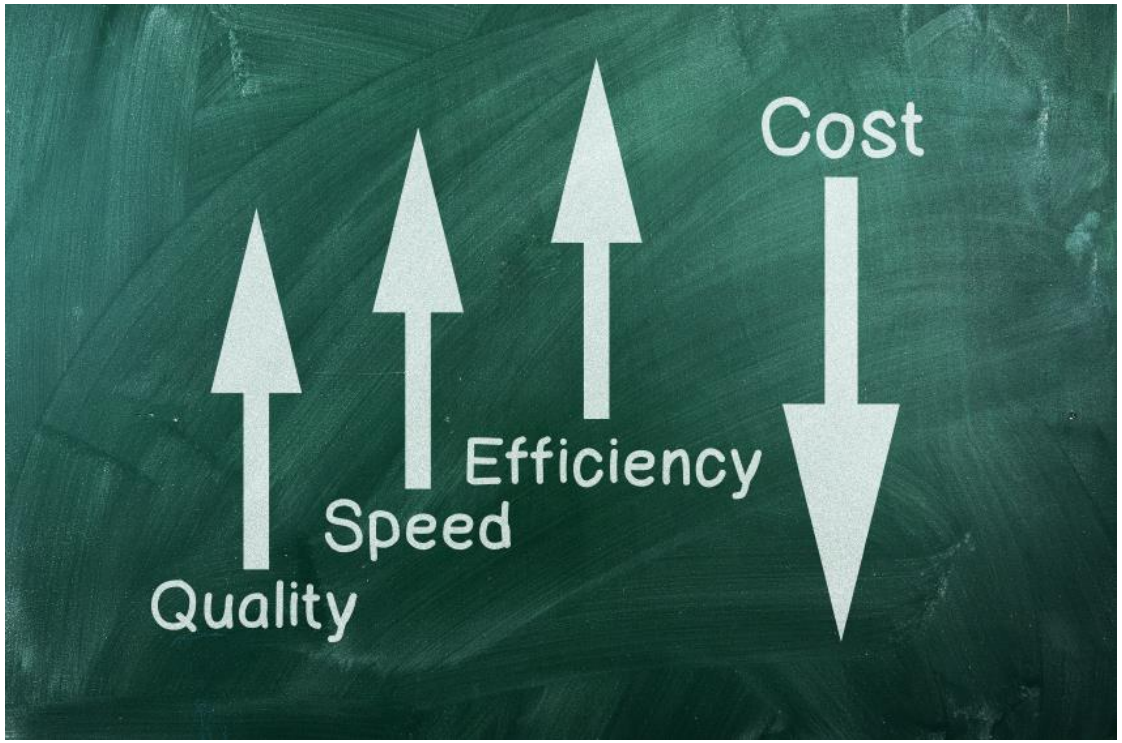
- Reference HW board with software stack (Qt+Embedded Linux)
- Desktop Emulation
- Live preview on the device
- Design Tools & IDE

Matured, Proven, & Scalable to support entire product generation

- Strong ecosystem with Millions of Qt developers
- Support All major SoC, MPU & MCU
- Rich documentation / tutorials
- +1600 C++ Libraries
- Support C++,Python, Javascript, HTML5
- Yocto, Buildroot, Cmake, Qmake

A Standalone Offering

- Enables a smartphone-like UX on a more cost effective, purpose-built platform.
- Freedom to move across hardware vendors without redoing GUI application
- Design tools for shrinking the iterations between Designer and Developer teams means faster GTM.
- Leverages hardware acceleration where available for GPU or 2D Graphics Accelerator
- Multiple build tool and OS support to fit your in-house expertise, not dictate them.



Make the best of Qt

- Scale seamlessly across your product line from higher end MPU devices running embedded Linux to low end MCU devices.
- Match your user experience across all products including your companion app.
- Consistent tools and language for all your platforms. Promotes Agile development
- Further accelerate your time to market through code re-use and ability to rapid prototype

More than 1600+ c++ classes

Qt Module offerings for faster & reliable development

Add-Ons

- Canvas 3D
- Charts
- Qt Quick 2D renderer
- Data Visualization
- Purchasing
- Active Qt
- Graphical Effects
- NFC
- Location
- Qt Quick 3D
- X11, Windows Extras
- Print Support
- Sensors
- Concurrent
- WebEngine
- Android, Mac Extras
- Image Formats
- Positioning
- Qt Lottie
- WebSockets
- XML
- SVG
- Bluetooth
- D-Bus
- WebChannel

Essentials

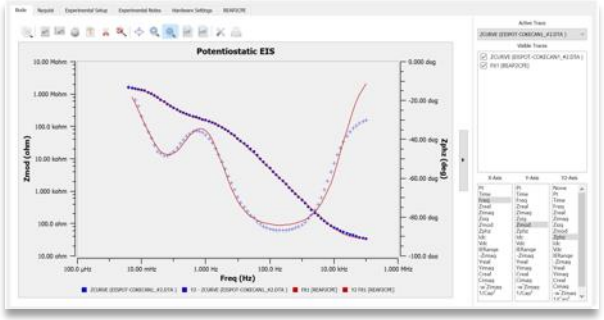
- Multimedia Widgets
- Quick Dialogs
- Quick Controls
- GUI
- Widgets
- Multimedia
- Quick Layouts
- Quick
- Core
- Network
- SQL
- Test
- QML

Industrial Automation Protocols

- Qt Serial Bus
- Qt VNC
- Qt WebGL Streaming
- Qt WebAssembly
- Qt MQTT
- Qt KNX
- Qt OPC UA
- Qt CoAP

QT FRAMEWORK DETAILS

Optimal UI solutions for each use case

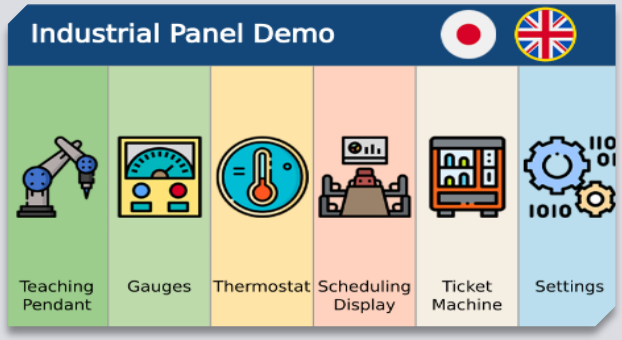


2D/ 3D UIs

Qt Quick declarative UI design (QML) for fluid, modern touch-based User Experiences

Web / Hybrid

Use HTML5 for dynamic web documents, Qt Quick for native interaction



Remote UIs

Run headless device UIs remotely in the browser using WebGL or WebAssembly

Qt Widgets

Customizable C++ UI controls for traditional desktop look-and-feel or more static embedded UIs for more limited devices

2D / 3D UIs with QML

Nice modern, phone like **UIs for all targets**

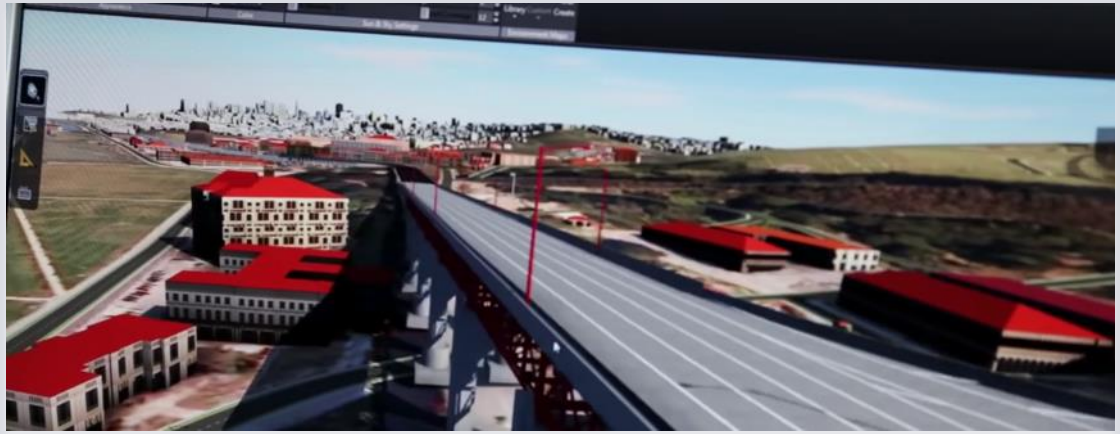
- Especially for embedded and mobile

WYSISYG UI design tool - **Qt Design Studio**

- Generates UI implementation in QML

QML declarative language for creating UIs

- Easy to learn
- Quick to prototype even in the target HW – no compilation needed
- Can be compiled to the native code to get the best possible performance
- Great tooling to find rendering bottlenecks (custom Engine with profilers)
- HW accelerated on targets with the GPU



QT FRAMEWORK DETAILS

Widgets

Easy to use, easy to extend, easy to style (Pute C++ APIs)

Most suitable for desktop

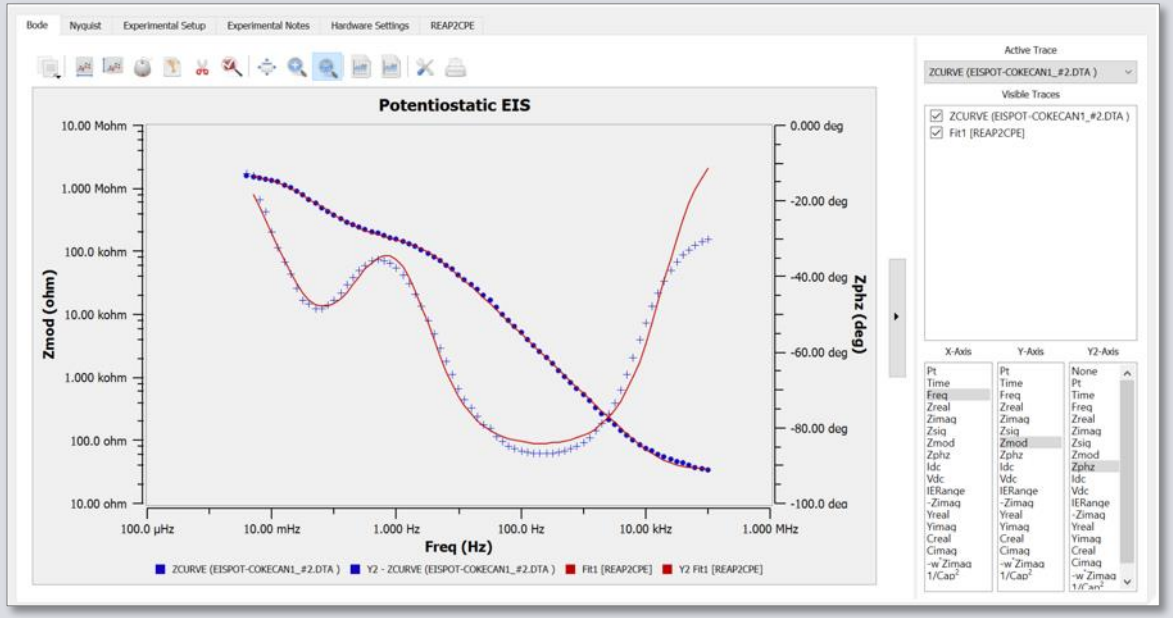
Native desktop look'n'feel – easily stylable

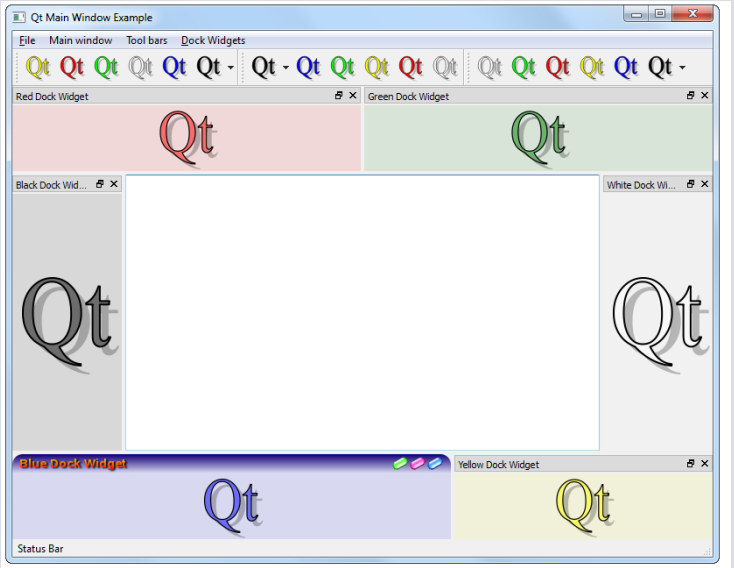
Easy to scale to any display size and orientation

WYSISYG UI design tool

- Create the UI sketch with a custom style in minutes
- Plenty of controls available: buttons, sliders, LCD number, tree view, dock widget

No graphics processor needed => extends the HW base

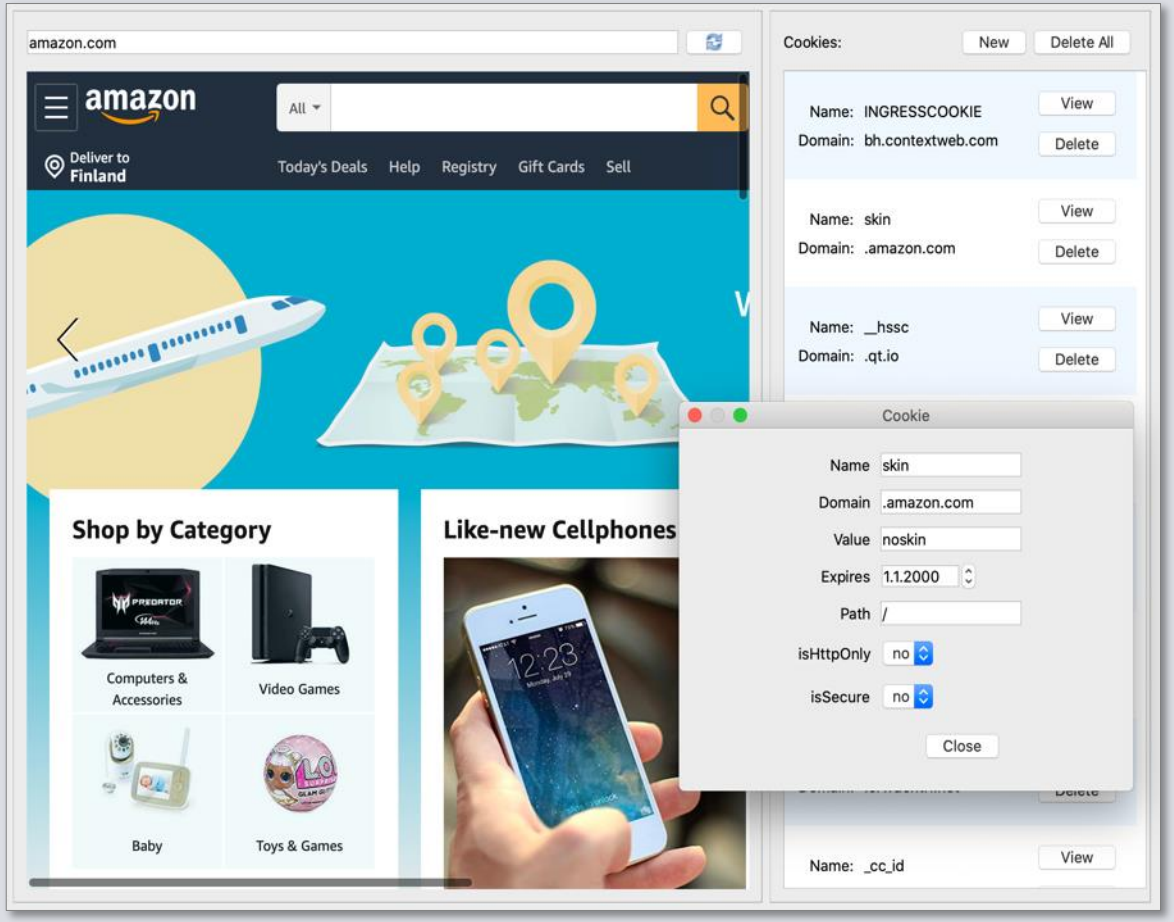




QT FRAMEWORK DETAILS

Web / Hybrid

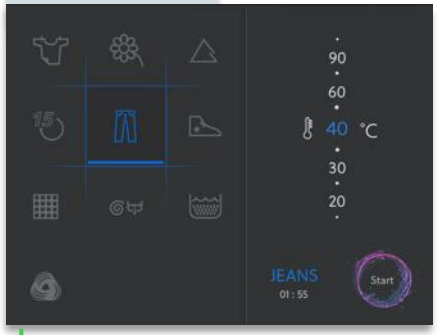
- Reuse and enhance web UIs in desktop or embedded targets
- No need to migrate the UI to Qt (or Vice-versa)
  - Shorter time-to-market
  - No need to test in several browsers
  - No surprises in rendering or performance
- Mix web UI with Qt UI controls
  - Both in Qt Quick and Widgets apps
- Qt allows exposing any platform API
  - Allows native calls in JavaScript
  - Extremely powerful



SINGLE CODEBASE

Cross product-line development

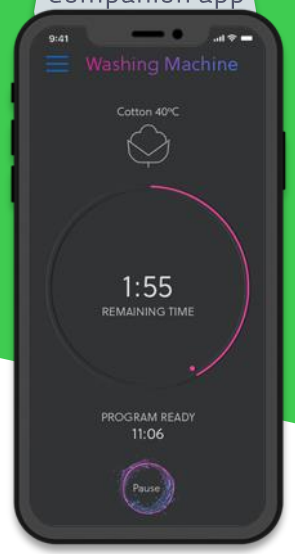
Low-end



Cortex-M4 MCU (<10 EUR BOM) – 640x480

- ✓ Qt for MCUs
- ✓ Smartphone-like UX
- ✓ Basic animations
- ✓ Bare metal or freeRTOS

Companion app



- ✓ Complex/ simple apps
- ✓ Win, Mac, Linux, Android, iOS
- ✓ WEBASM

Mid-range

ARMv7A 32bitlow end MPU (<30 EUR BOM) – 854x480

- ✓ Higher resolution
- ✓ 2.5D Graphics
- ✓ Full Qt Framework
- ✓ Advanced animations
- ✓ Linux or RTOS

High-end

ARM-v8A 64bit Quad Core high end MPU (<100 EUR BOM) – 960x480

- ✓ Highest resolution
- ✓ Dual screen support
- ✓ 2D/3D Graphics
- ✓ Full Qt Framework
- ✓ Linux or RTOS

TARGETS

Qt on microcontroller hardware

Declarative UI and OOP – the best of both words

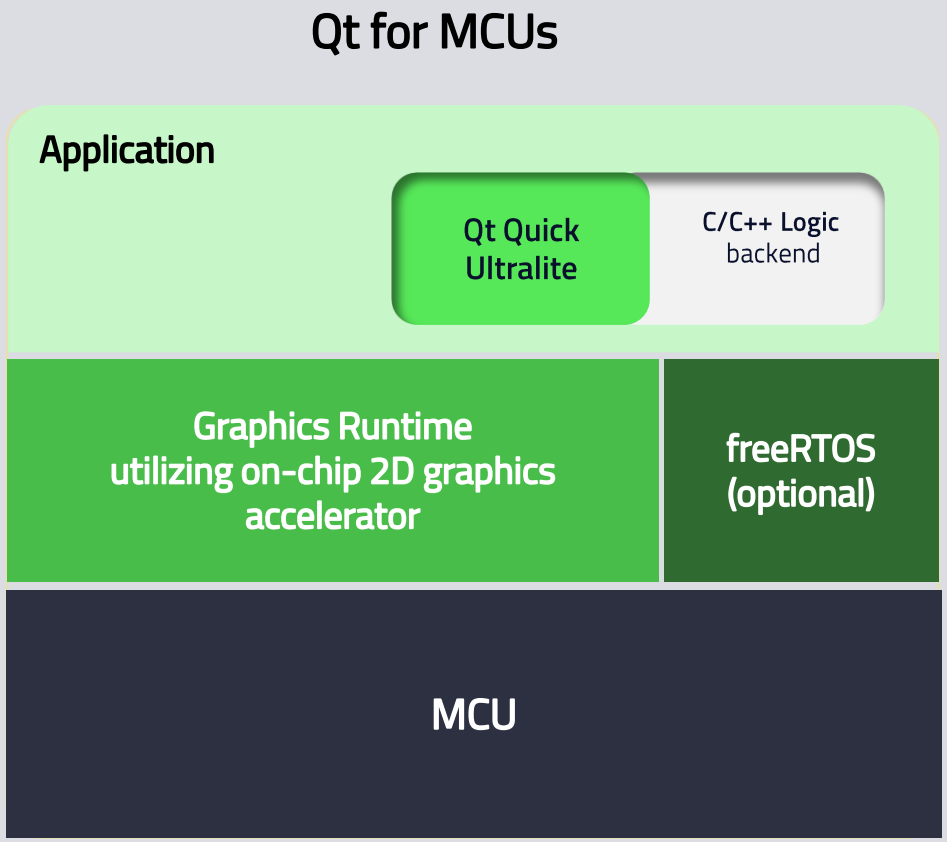
- Re-use and deploy same QML based UI while implementing Application logic in standard C/C++

Ultimate Performance. Tiny Footprint.

- A new rendering engine uses HW 2D accelerators to achieve good graphical performance. The runtime itself has a very small footprint (starting from ~80KB)

Supports on a wide range of MCUs and RTOSs

- MCUs from ST, NXP, Renesas, Cypress/Infineon, Xilinx UltraScale+ and many more on custom porting
- Bare Metal or FreeRTOS (on selected boards)

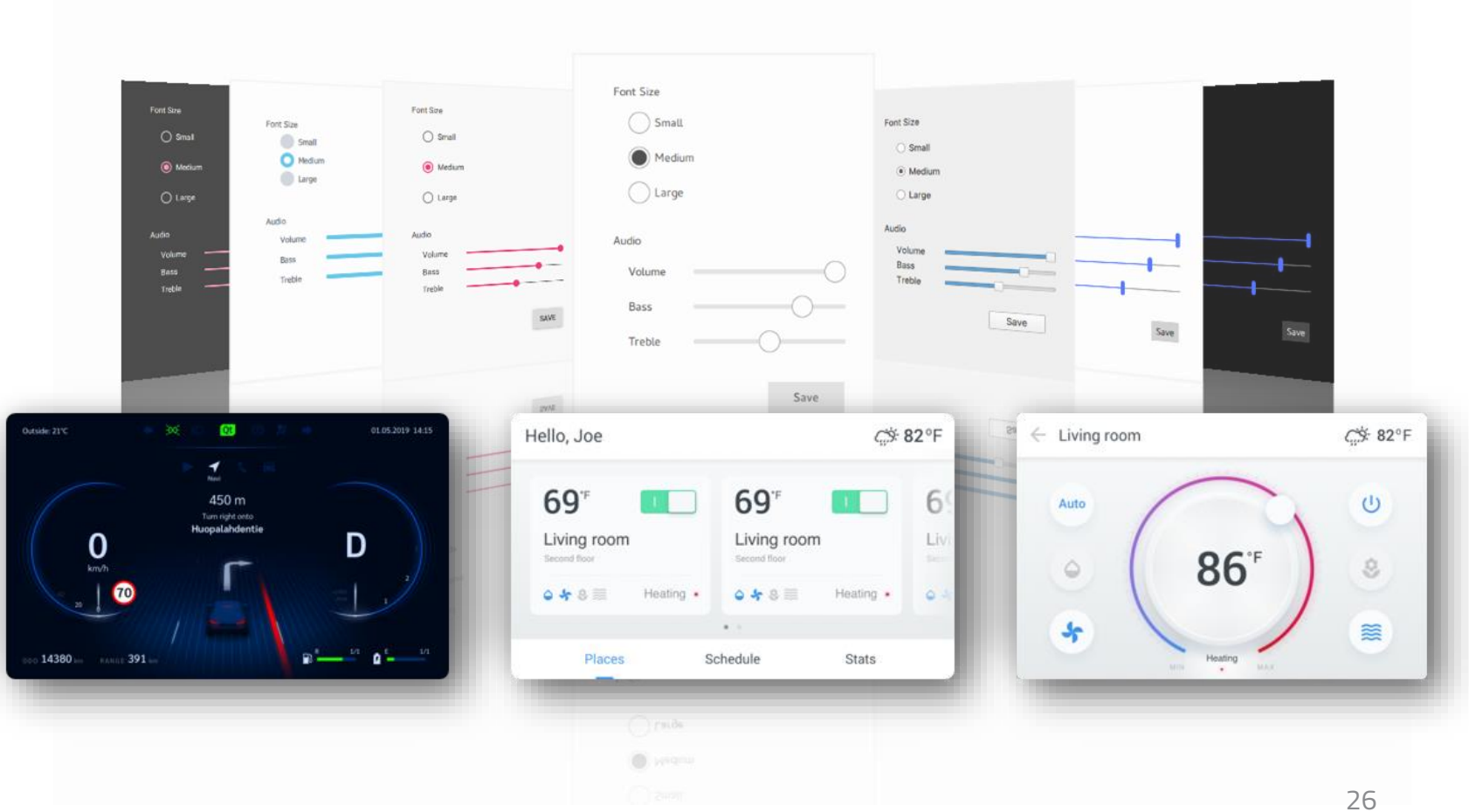


UX UI

Topnotch
GUI Framework

World Class GUI Framework

Various kinds of well-structured, mature controls allow you to develop directly. *Button, Popup, Slide, Switch, Swipview, Stackview etc.* All the gestures & input method could be implemented with single line of code in Qt.



Boost Your Process by QML and Qt Tools

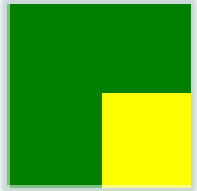
Easy, intuitive QML and straightforward tools simplify development process.

Boost by QML

QML is declarative language, objective code, compiled to a binary machine code

```
Rectangle {
    x: 50
    y: 50
    width: 100
    height: 100
    color: "green"
}
```

```
Rectangle {
    x: 100
    y: 100
    width: 50
    height: 50
    color: "yellow"
}
```



QML allows easy development process – JSON-like syntax

Boost by Tool

Designers produce QML based "UI Specification" directly usable by developers

Client Designers
UI flow & navigation
Wireframes
Visual assets

Client Developers
Custom UI components
Data bindings
Application logic

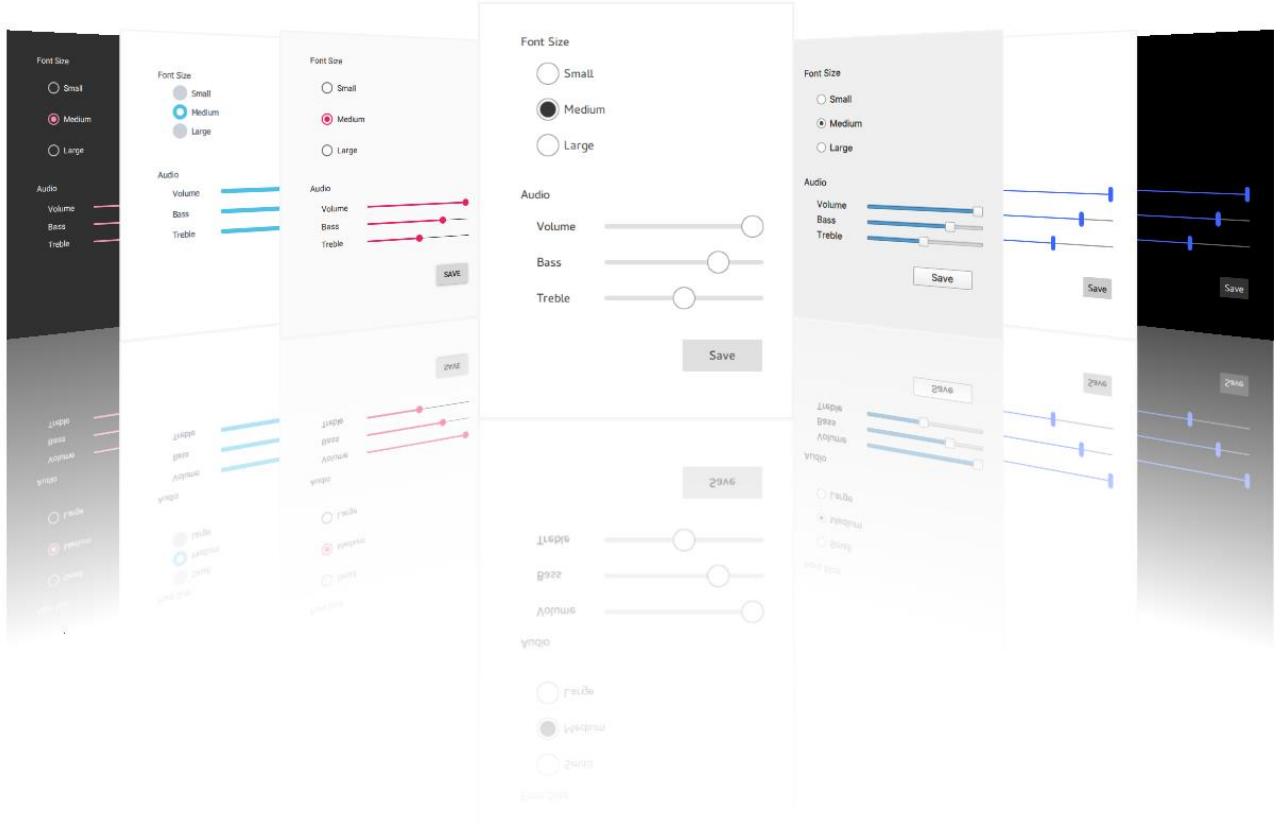
QML syntax

- The **import** statement imports a module in a specific version.
- Comments can be made using **//** for single line comments or **/* */** for multi-line comments. Just like in C/C++ and JavaScript
- Every QML file needs to have exactly one root element, like HTML
- An element is declared by its type followed by { }
- Elements can have properties; they are in the form `` **name: value** ``
- Arbitrary elements inside a QML document can be accessed by using their id (an unquoted identifier)
- Elements can be nested, meaning a parent element can have child elements. The parent element can be accessed using the **parent** keyword

```
// RectangleExample.qml

import QtQuick 2.5

// The root element is the Rectangle
Rectangle {
    // name this element root
    id: root

    // properties: <name>: <value>
    width: 120; height: 240

    // color property
    color: "#4A4A4A"

    // Declare a nested element (child of root)
    Image {
        id: triangle

        // reference the parent
        x: (parent.width - width)/2; y: 40

        source: 'assets/triangle_red.png'
    }

    // Another child of root
    Text {
        // un-named element

        // reference element by id
        y: triangle.y + triangle.height + 20

        // reference root element
        width: root.width

        color: 'white'
        horizontalAlignment: Text.AlignHCenter
        text: 'Triangle'
    }
}
```

Qt Quick Controls: Ready-made QML UI building blocks

- Qt Quick Controls 2 provides 35 functions and 5 style templates.
- Rendered via the Qt Quick scene graph
- Saving developing time by avoiding duplicated works makes advanced work performance
- Easily customizable Style and keep brand consistency
- More details are on our documentations or press **F1** then you can find explanation on it

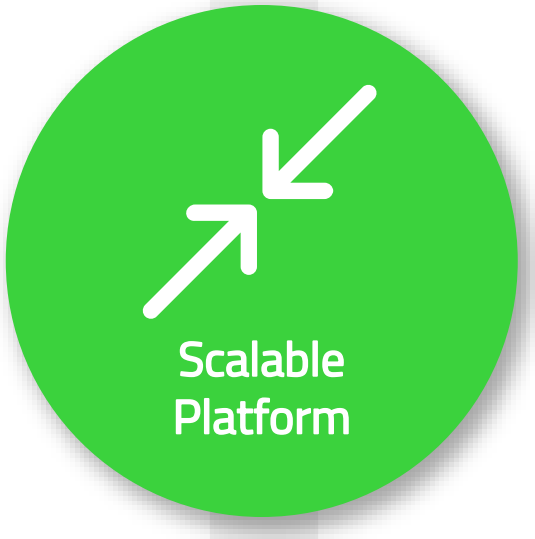


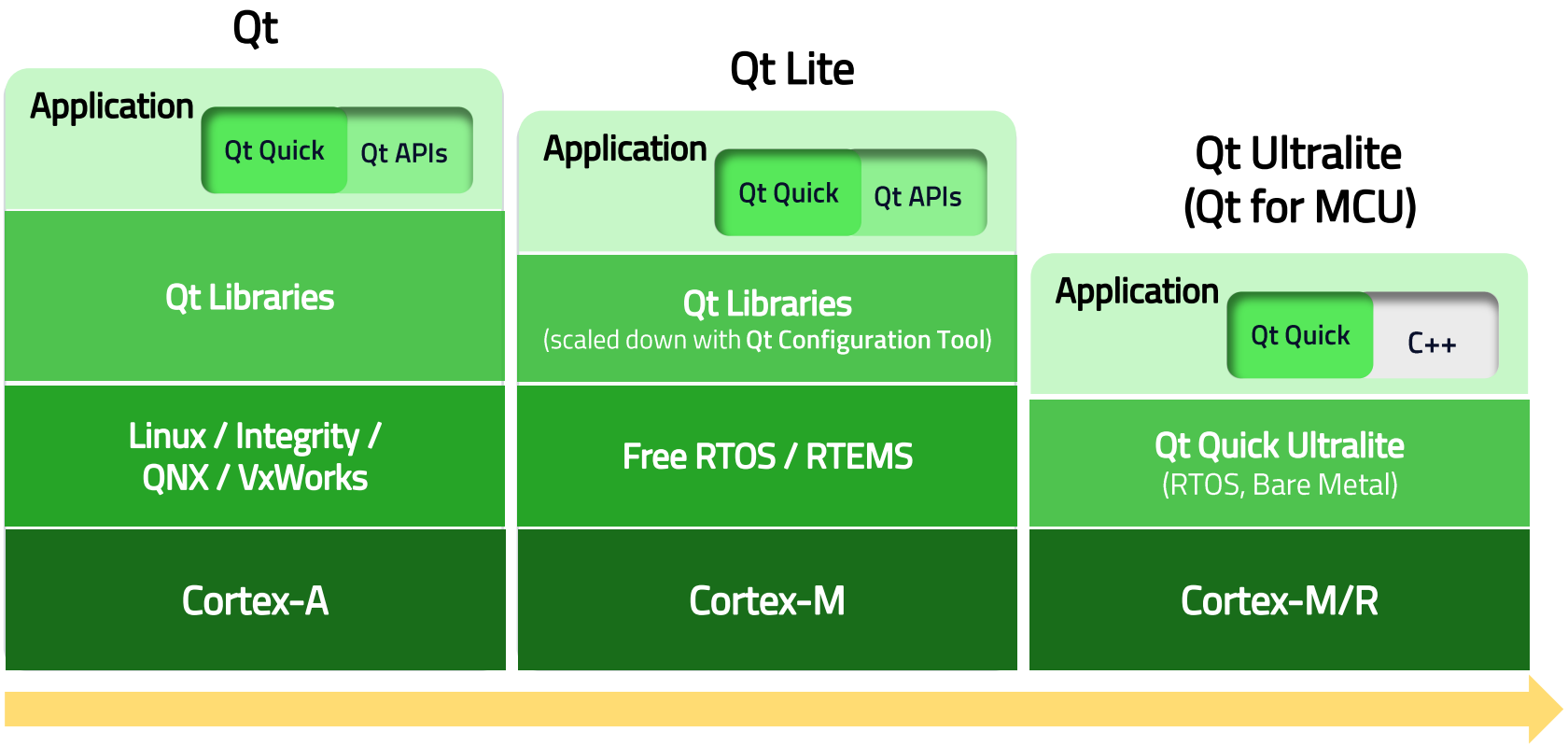
Qt Quick Controls: Ready-made QML UI building blocks

More details : https://doc.qt.io/archives/qt-5.10/qtquickcontrols2-guidelines.html

From High-end to Low-end

Qt framework is provided scalably following your needs.
Full stack for high-end, minimal stack for low-end hardware.





Qt for MCUs – *Ultimate performance, Tiny footprint*

Qt for MCUs uses a new graphic runtime, Qt Quick Ultralite, that delivers high performance with low memory consumption, which is achieved by a new translation of QML to C++.

Inside Qt for MCUs

GUI Application

QML UI
frontend

C/C++ logic
backend

Qt Quick Ultralite
Graphics Runtime
utilizing on-chip 2D graphics accelerator

NXP ST RENESAS CYPRESS EMBEDDED IN TOMORROW™ infineon

.....More to come.....

Thermostat Demo

› Showcases different user interface controls, user interactions and list models

To see the full demo clip, please visit https://youtu.be/p9_Qy3kw1wc

Thermostat Demo

Key Metrics		NXP RT1050/1064	STM32F769i	STM32F7508	STM32H750B	STM32F469i
Display	Resolution	**480x272**	**800x480**	**480x272**	**480x272**	**800x480**
	Pixel Depth	16-bit color	32-bit color	32-bit color	32-bit color	24-bit color
RAM Usage	Qt runtime	**230 kB**	**230 kB**	**210 kB**	**225 kB**	**230 kB**
	Assets	1.4 MB*	3 MB*	1.4 MB*	1.4 MB*	3 MB*
	Framebuffer	522 kB (single buffering)	3.1 MB (double buffering)	1 MB (double buffering)	1 MB (double buffering)	1.1 MB (single buffering)
	Total	2.2 MB	6.3 MB	2.6 MB	2.6 MB	4.3 MB
Flash Usage	Qt Application	**404 kB**	**419 kB**	**404 kB**	**401 kB**	**354 kB**
	Assets	1.4 MB	3 MB	1.4 MB	1.4 MB	3 MB
	Total	1.8 MB	3.4 MB	1.8 MB	1.8 MB	3.4 MB
Frame rate	Max	**60 fps**	**60 fps**	**60 fps**	**60 fps**	**35 fps**
	Min	60 fps	40 fps	40 fps	45 fps	22 fps

*Assets can also be read directly from flash and never copied to RAM

Connectivity


```
{
  "id": "961b276c-40f7-11ea",
  "location": "b77f-2e728ce88125",
  "rpm": 6200,
  "temp": 27.4,
  ...
}
```


Attach to peripherals

Control external hardware via any protocol.
CANbus, Modbus, Serial Port, Bluetooth, BTLE,...

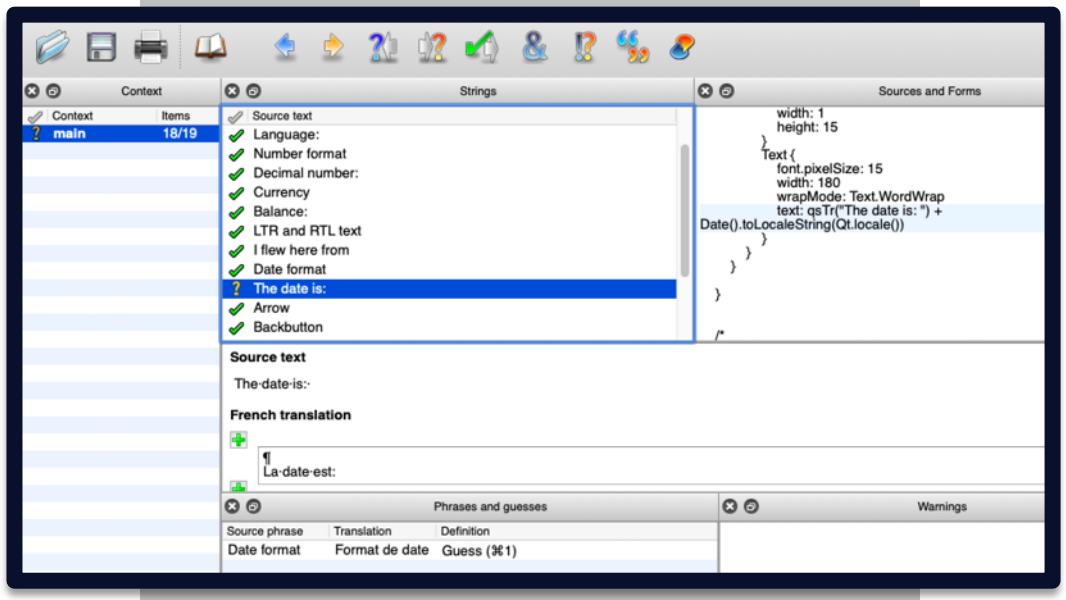
Data Serialization

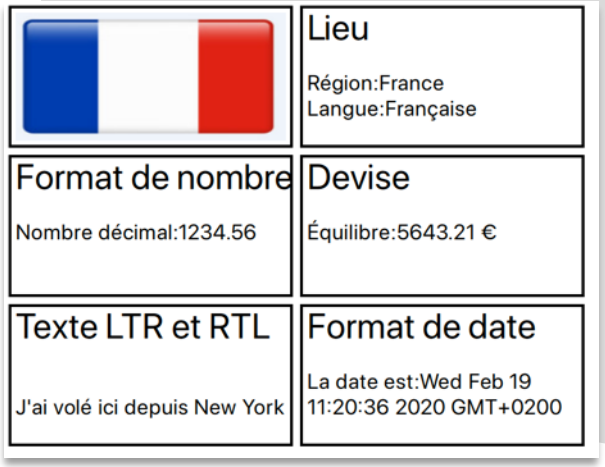
Store and export data to industry standard formats.
JSON, CBOR, XML,...

Cloud synchronization

Publish telemetry data, visualize health status, database storage.
Protocol layer: MQTT, CoAP, OpcUA, KNX, HTTP,...
Transport layer: TCP, UDP, Websockets, Local sockets,...

Internationalization: Efficient product adaptation to any target market





Application development with engineering language

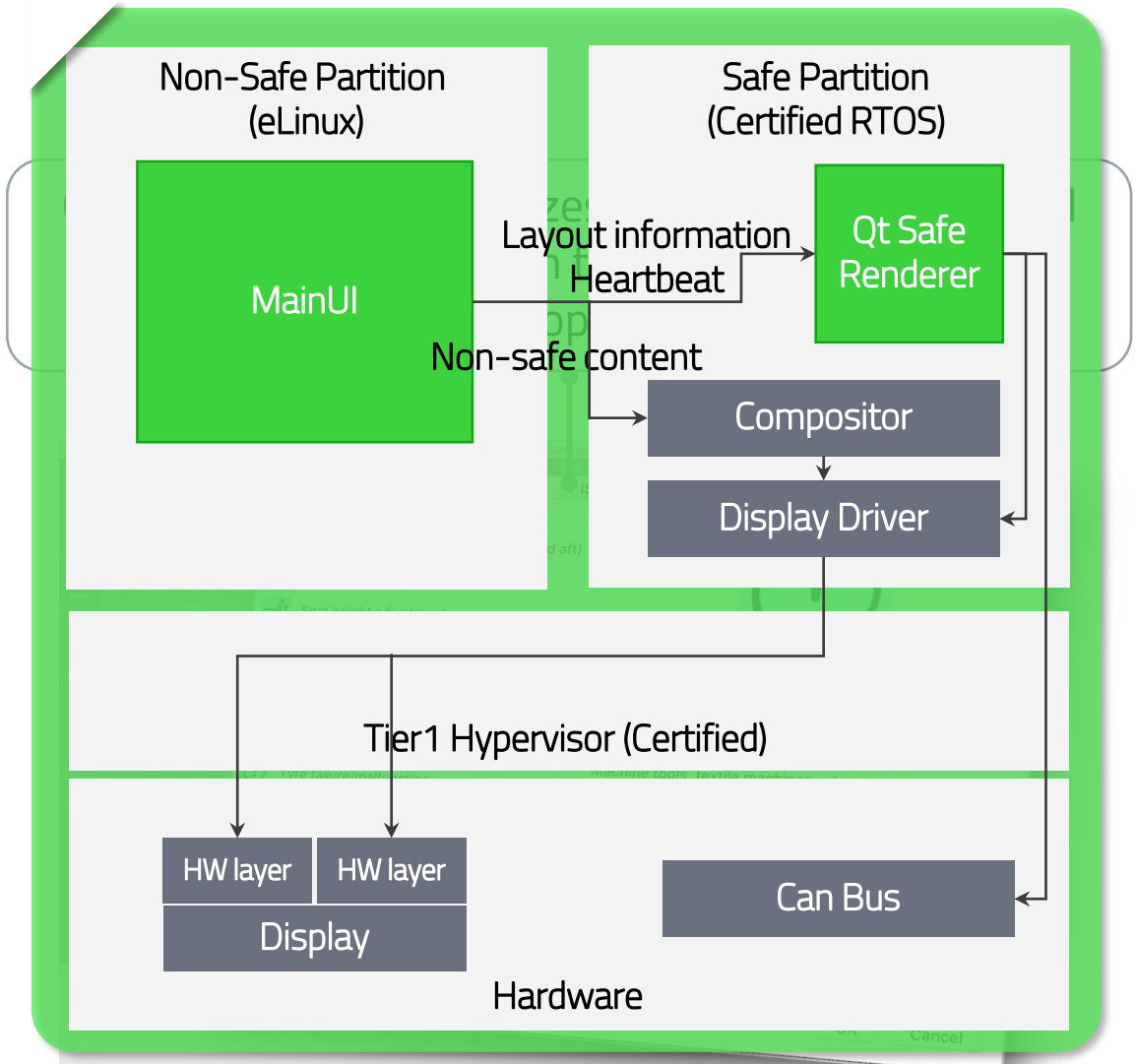
Translations provided by professional translators

Dynamic locale selection in the target

Qt Safe Renderer

Easier way to safety critical systems with a rich GUI

- Tool for rendering the safety-critical information in functional safety applications based on Qt.
  - *Renders* the safety critical UI
  - *Controls* how the graphics planes are
  - *Monitors* correct operation of non-safety critical UI
  - *Disables* the non-safety UI if error detected
  - *Will try* to restart non-safety UI if failure detected



Easy Defining Safety Critical UI Parts

Flexibility – No need of modification

Functional Safety

› The objective of functional safety is *freedom from unacceptable risk* of physical injury or of damage to the health of people either directly or indirectly.

Functional Safety Standards

› We have certification for 4 standards:

ISO 26262:2018-6 **ISO 26262**:2018-8	**ASIL-D**	Road vehicles
IEC 61508:2010-3 7.4.4	**SIL-3**	Electrical/electronic programmable safety related systems
EN 50128:2011 6.7.4	**SIL-4**	Railway applications
IEC 62304:2015	**2006+A1**	Medical device software

TÜV NORD

DAkkS Deutsche Akkreditierungsstelle D-ZE-11074-01-00

Certificate

No. SEBS-A.113256/17, V2.0

TÜV NORD Systems GmbH & Co. KG hereby certifies to

The Qt Company
Bertel Jungin aukio D3A
02600 Espoo, Finland

that the

Qt Safe Renderer

meets the tool requirements listed in the below mentioned standards

- IEC 61508:2010; Part 3; Section 7.4.4; Qualified up to SIL 3
- ISO 26262:2018; Part 8; Section 11; Part 6; Qualified up to ASIL D
- EN 50128:2011; 6.7.4; Qualified up to SIL 4

Certification Program Leittechnik (SEB-ZE-SEECERT-VA-320-20, Rev. 3/9.15)

This tool can also be used as a software supporting tool in a software safety lifecycle according to IEC 62304:2015(2006+ A1)

Base of certification is the report SEBS-A.113256/17AR and the tracking list in the valid version. This certificate entitles the holder to use the pictured Trusted Tool mark.

Valid until: 2024-04-17
File reference 8114839486

Hamburg, 2019-04-17

Bianca Pfuff

Certification Body SEECERT
TÜV NORD Systems GmbH & Co. KG
Große Bahnstraße 31, 22525 Hamburg, Germany

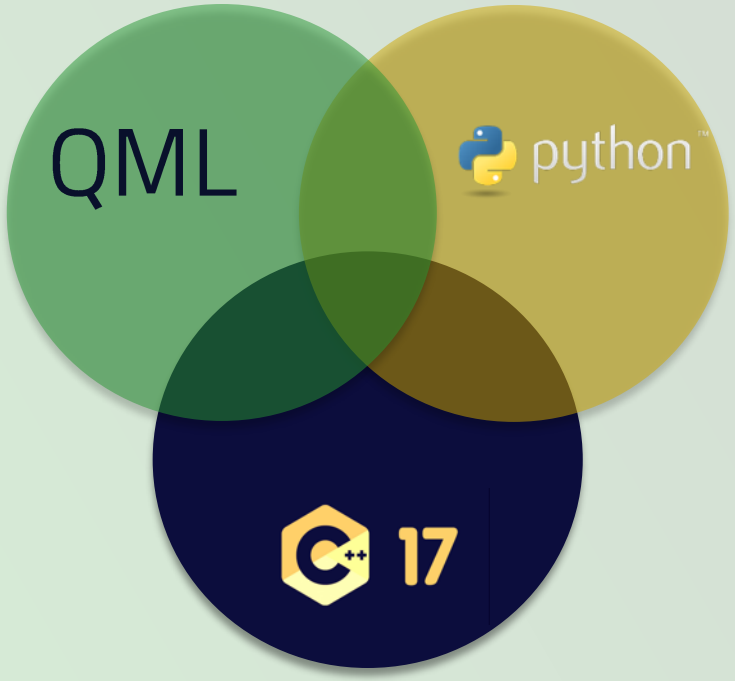
Voluntary Certification
TÜV NORD
TÜV NORD Systems GmbH & Co.KG
Trusted Tool
QT Safe Renderer
Qualified up to
IEC 61508:2010 SIL3
ISO 26262:2018 ASIL D
IEC 50128:2011 SIL 4
SEBS-A.113256/17

Qt for Python (PYSIDE)

- **Easy to extend applications**
  - Cross platform plugins without recompilation
  - Scripting support
  - Full control of runtime environment
- **Integrate Machine Learning**
  - Python most used language for ML
  - Easily accessible via Qt for Python
- **Remove programming language barrier**
  - 4th most popular language (StackOverflow Insights 2019)
  - 2nd most loved language
  - 1st most wanted language

Best of all worlds

- Fast UI iterations
- Design Tooling
- Declarative



- Rapid prototyping
- Integrate Machine Learning
- Extensibility (plugins, scripts)

- Best performance
- Scalable

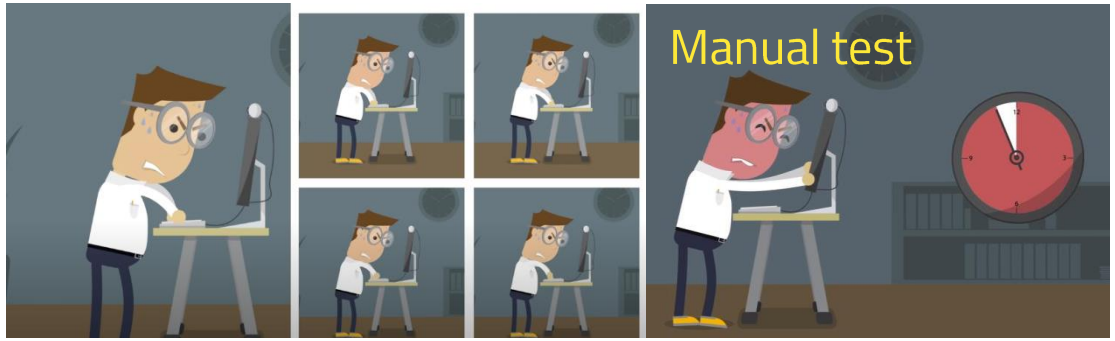
Challenges with GUI Test and how Squish + Coco can help

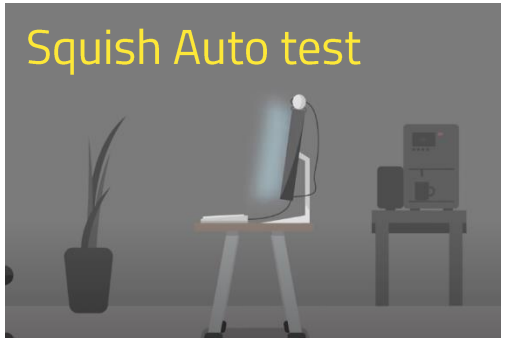
Manual GUI test challenges

- Manual test or GUI Test equipment
- Difficult to check/confirm the test results
- Impossible to automate the regression test
- Diverse input/output environment and test environment setup
- Get quantified test result and share.
- Time and cost
- Waste of Human resource
- Dependency to testers
- None-quantifiable test process and result

Automated GUI test advantages

- 24 hours / 7 days a week testing. Reduce testing hours.
- Removes "Human errors"
- Identical repeat of test. (Quantifies test hours)
- Diverse test scenarios are available to test.
- Diverse inputs can be tested
- Multiple tests can go in parallel
- Reproduce test and share test result.





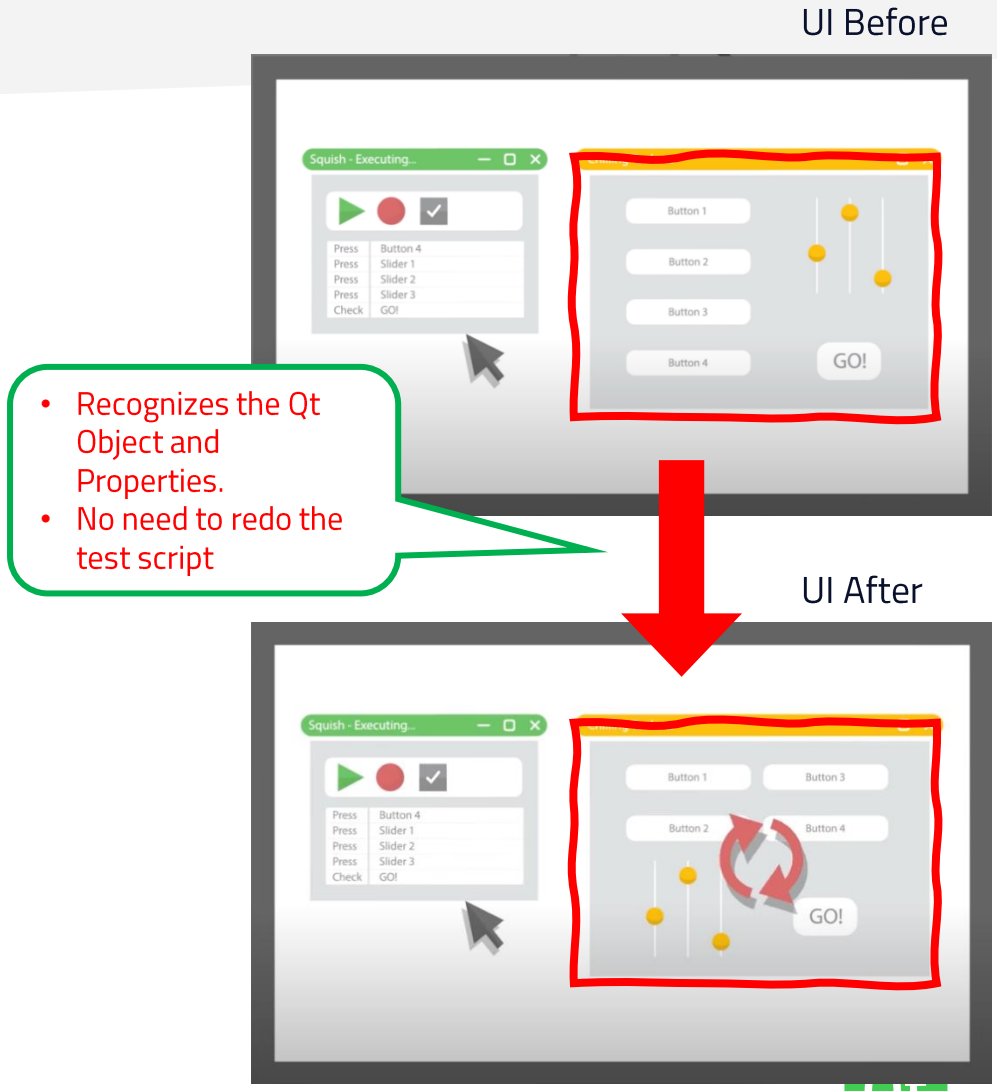
Why Squish & Coco for Qt project?

Squish GUI Tester

- Dynamically/automatically recognizes **Qt object and property**.
  - provides simplified workflow for test case creation.
  - provides informative reports to developers, QA personals.
  - enables view-logic verification, as well as simple pixel comparison.
  - no need to modify AUT (application under tests) in most cases.
- Embedded device
  - provides seamless workflow for all devices, platforms
  - serves as remote controller during test case creation (upcoming version)

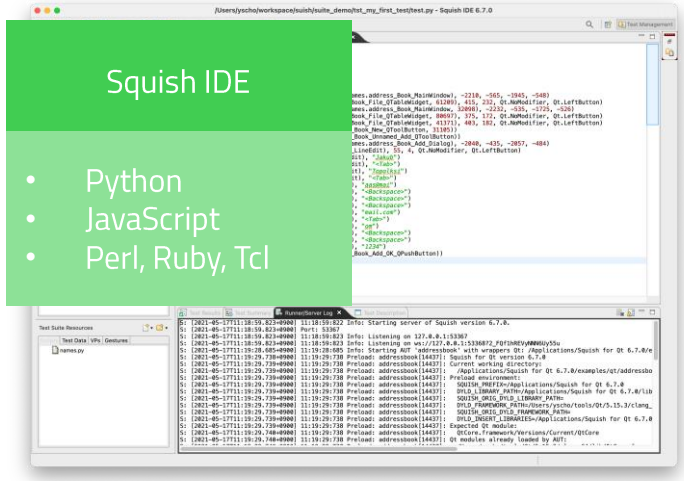
Coco

- Perfect match for modern GUI project, supports QML as well as C++
- Makes testing more effective and productive.
  - Identifies untested code, dead code, redundant test
  - Provides various coverage levels
  - Provides easy-to-use GUI for result browsing and analysis



Qt Tools for Automated GUI Test

Squish

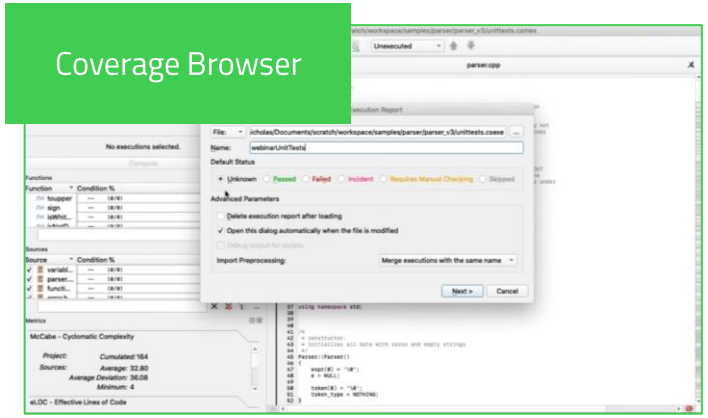


Squish IDE

- Python
- JavaScript
- Perl, Ruby, Tcl

Squish Runner | Squish Server | Hook, APIs, …

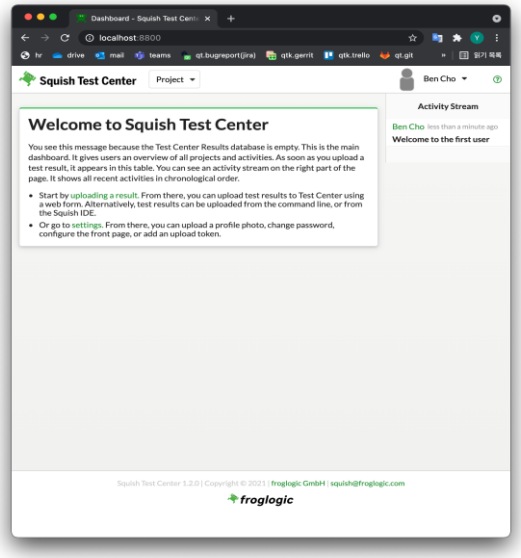
Coco



Coverage Browser

Coverage Scanner | Coco QML Scanner | Wrappers

Test Center



WEB APP | Server

Design-Development-Deployment Process in Embedded Development

Design/Graphics Authoring

Ps Figma Xd Adobe XD

UI/UX Scenario Implementation

DS

Brand UX Asset
Graphics + Custom Component

Custom component
Brand Specific color, controls, buttons, UI flow, animation effects, Theme

Custom component utilization by Qt Design Studio for optimal design-development process

Device Creation

QC Qt Application Manager

Qt Wayland Compositor

Qt Middle ware

OS

HW TARGETS

- ✓ 2D & 3D Unified UI with seamless look & feel
- ✓ Ready made 1,600+ C++ Libraries for Automotive Development
- ✓ Strong Middleware
- ✓ Chipset Agnostic
- ✓ Qt Wayland Compositor
- ✓ HTML5/JS/C++/QML/Python Support

Deployment Testing & Validation

Qt Test

froglogic

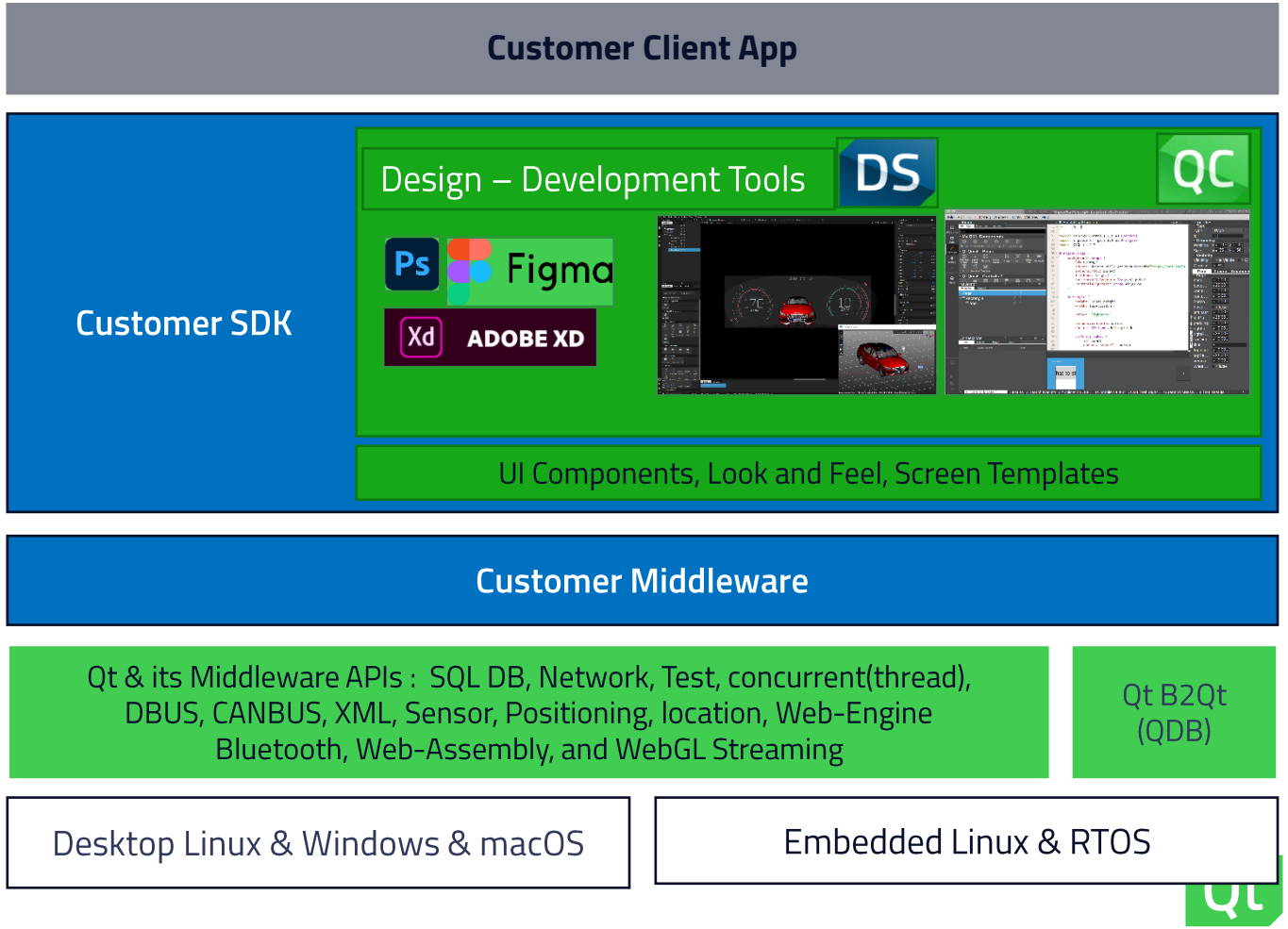
Squish Coco Code Coverage

Rich Qt Middleware Contribution on Customer Platform SDK

- Middleware Based on Qt
  - Allows to develop & test on Desktop(Host)

- Qt framework provides
  - HMI toolchain, 1600+ Rich C++ Libraries
  - Networking, Database, Browser, Connectivity Libraries

- Customer UX Components
  - 2D/3D Branded specific colors, controls, buttons, UI flows, animation effects, Themes



Thank you !
감사합니다.

조용준 부장
june.joe@qt.io 010-5745-6744
APAC Business Development Lead
The Qt Company