# QML Next

## Qt Quick UI with non-C++ languages

Qt Contributor Summit 2024

2024 Sept, by vladimir.minenko@qt.io

Qt Group

# Motivation

Why QML Next

- Enlarge the footprint of the use of Qt through
  - to use cases where C++ is not used (that much)
  - enabling the use of Qt without writing a Qt app
- Enlarge the Qt user community through
  - Keeping being modern and appealing
  - Stay in the loop even if users (have to) develop in a non-C++ language
  - Flatten the learning curve
- In all of the above, we focus on QML (language) and Qt Quick (framework)

The focus of this talk

# Where

we could work on this

- In the Qt Framework software

The focus of this talk

- Other aspects aside touching the code of Qt

# In the Qt Framework Software

If there, so what would be that then?

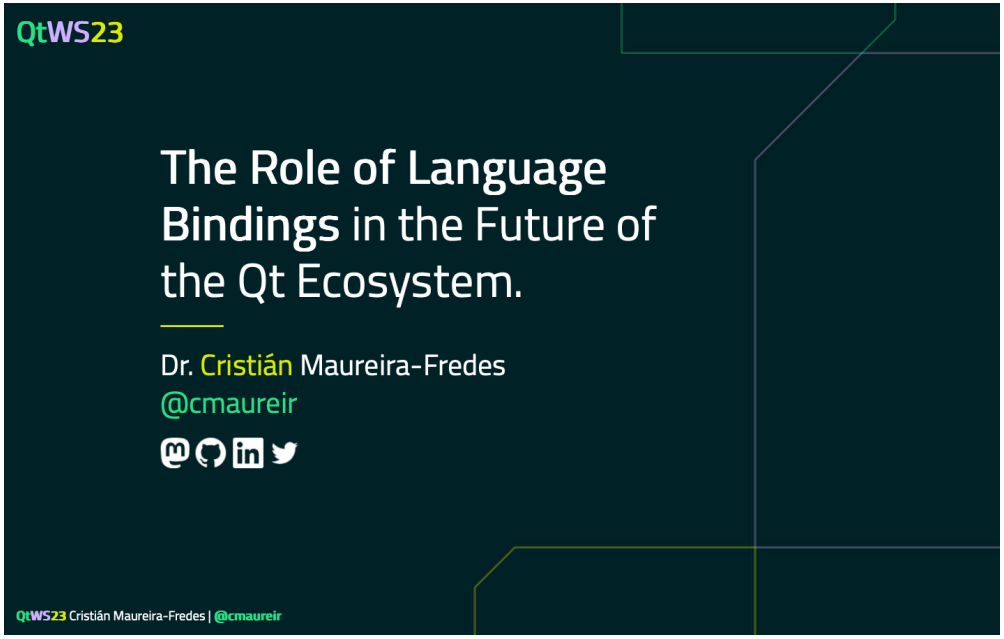
The focus of this talk

- QML and Qt Quick bring UI, other languages - data models / business logic
- Enabling development of Qt Quick application without needing any C++ glue
- Enabling development of foreign language applications that integrate Qt Quick into a larger non-Qt UI

# It is serious

QML Next as “Qt Quick UI with non-C++ languages”

Potentially, some of you were in that talk in the last year:




The importance of this for Qt is a good reason why this talk is a bit similar

# The concept

QML Next as “Qt Quick UI with non-C++ languages”

**An extension interface for Qt Quick UI to use backends written in other languages**

- The persona:
  - Some code base and knowledge in language XYZ
  - Wants to get a UI for that code base
  - Interested in Qt Quick / QML as technology, but does not want to touch C++ to just get that UI done

- What can we do for this user beyond sending to read https://wiki.qt.io/Language_Bindings ?
  - Started a self-explorational exercise
  - Defined requirements ”minimal app”
  - Got it done with backends in C++ , C#, and Python
  - Missed to cover Rust and Swift, and did not get to JS
  - Looked in the mirror and asked how is it?

# Expectations

QML Next as “Qt Quick UI with non-C++ languages”

**An extension interface for Qt Quick UI to use backends written in other languages**

An MVP should:

- Zero effort to start with some minimal functionality
- Advanced use requires advanced techniques
- A final app still uses Qt to run: the main() and the event loop with Qt (C++)
- A reference integration for most appreciated ones: C#, Swift, Rust, <see the next 3 lines>
- <get our head around if and how we handle JS, and which “JS” that actually is>
- <see how we plug the new QML for Android>
- Explain or extend the use of Python with Qt in other ways than PySide (Qt for Python) does today
- Provide QML-internal interfaces for more languages with C/C++ interoperability

# Keep going

QML Next as “Qt Quick UI with non-C++ languages”

- Reduce Qt exposure on the backend side:
  - Exploring how to handle custom types
  - and register functions
  - how that could look like for users
- Exploring what is needed in the QML Engine to work well with other languages
- Getting one developer for Swift now and one for Rust in the Fall
- See <> brackets on my other slides

# Now, it is your turn

QML Next as “Qt Quick UI with non-C++ languages”

**Tell us:**

- “This ‘persona’ is like me at X%”
- “You, guys, actually missed XYZ whereas this is very important”
- “I want to join that!”