basysKom

Entwicklerforum HMI 2017

Qt Quick Controls 2 für die Entwicklung von Embedded-HMIs

FRANK MEERKÖTTER

Einführung

Begriffsklärung

Qt
- C++ Framework
- Zur Umsetzung der Anwendung (Backend)

QML
- Declarative Beschreibung des HMI
- Javascript für HMI-Logik
- Ausgeführt durch die QML-Engine

Qt Quick
- Graphik-Stack und Primitive (Rectangle, Text, Image, Row, …)

Qt Quick Controls
- Abstraktionen und UI-Controls
- Menus, Buttons, Slider, StackView, ButtonGroup, ...

Einführung

Qt Quick Controls 2.x

- Neu in Qt5.7 (2016)
- Focus auf Embedded und Mobile
- Kein Nachfolger der Qt Quick Controls 1.x

Agenda

- Übersicht verfügbare Controls
- Styling
- Vergleich Architektur und Performance
- Zusammenfassung & Ausblick

basysKom

Entwicklerforum HMI 2017

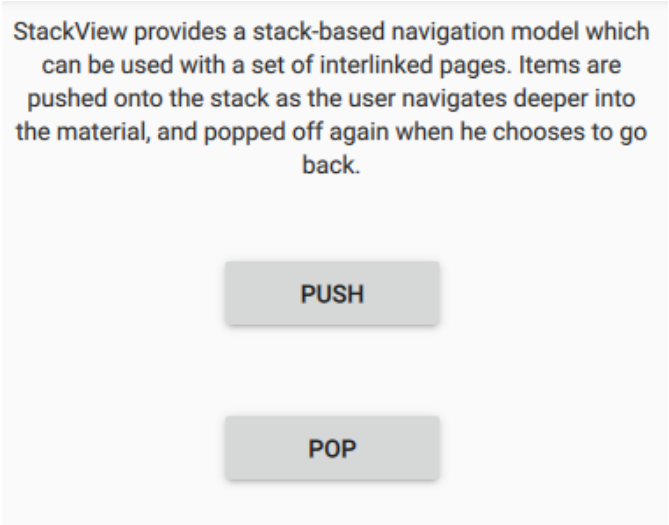
Frank Meerkötter

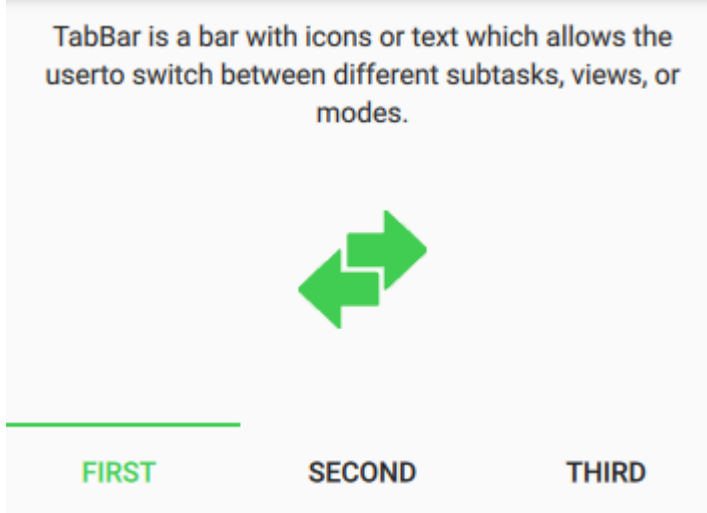
Verfügbare Controls

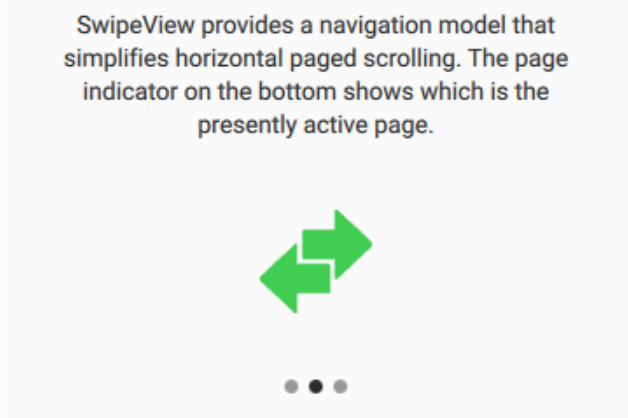
Navigation

Navigation Controls

- StackView
- SwipeView
- TabBar
- ...



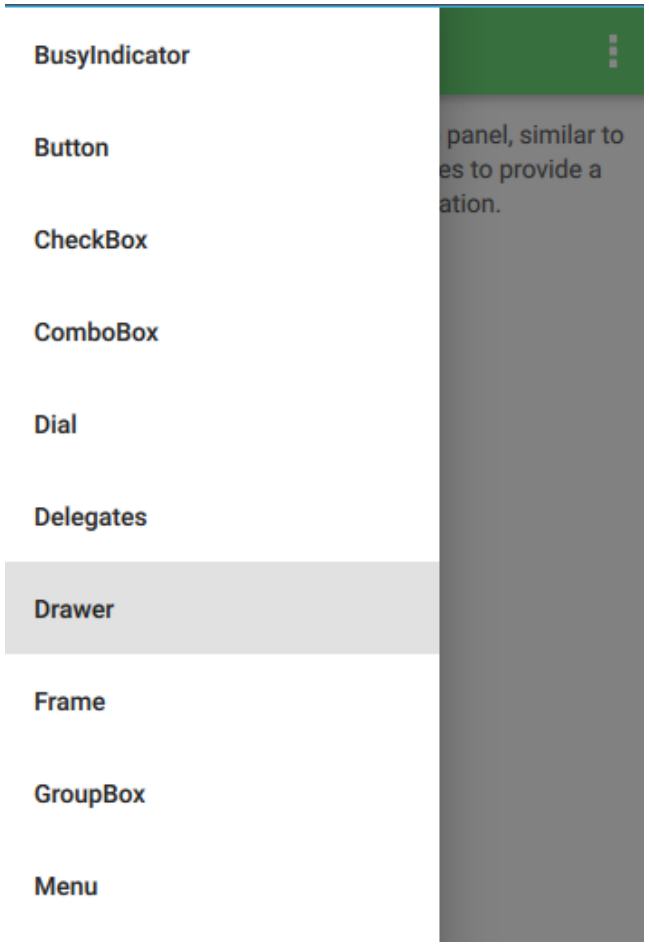


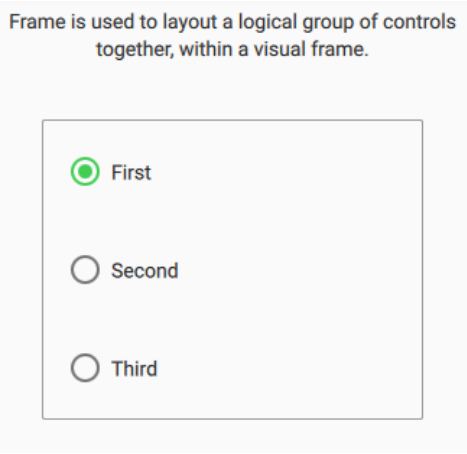


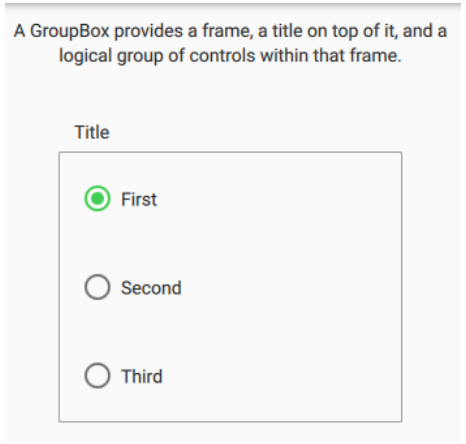
Container

Container Controls

- ApplicationWindow
- Frame
- Drawer
- Groupbox
- Page
- Toolbar
- ...







Indicator

Indicator Controls

- BusyIndicator
- Progressbar
- Scrollbar
- PageIndicator
- ...

ProgressBar indicates the progress of an operation. It can be set in an indeterminate mode to indicate that the length of the operation is unknown.

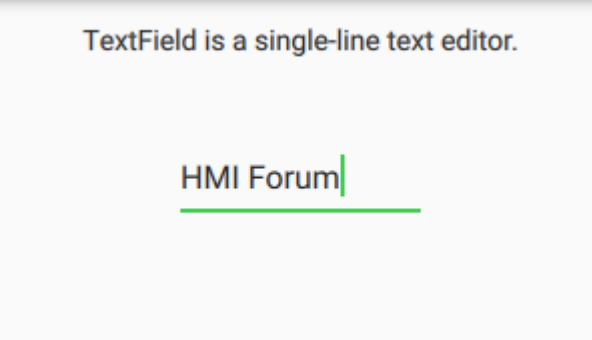
BusyIndicator is used to indicate activity while content is being loaded, or when the UI is blocked waiting for a resource to become available.

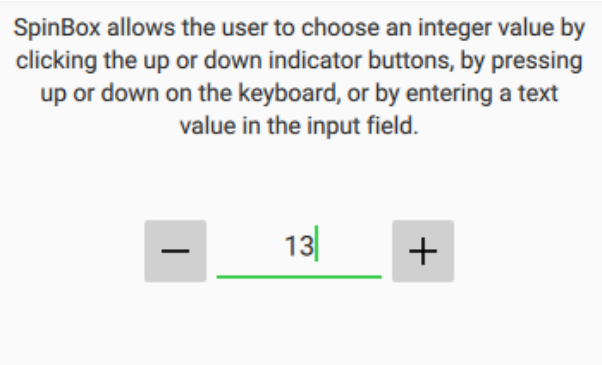
PageIndicator is used to indicate the currently active page in a container of pages.

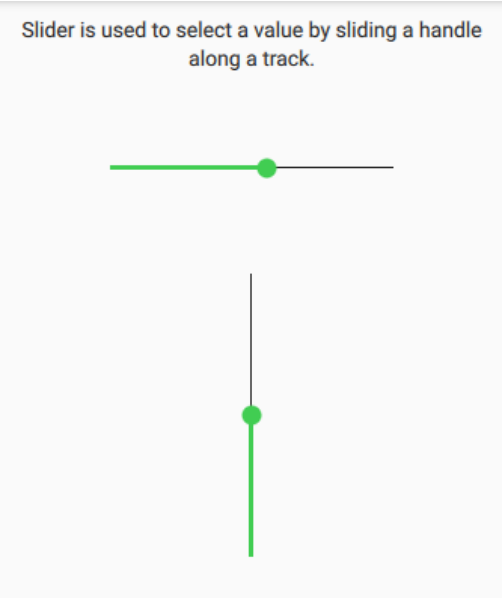
Input

Input Controls

- Slider
- SpinBox
- TextArea
- TextField
- ComboBox
- ...



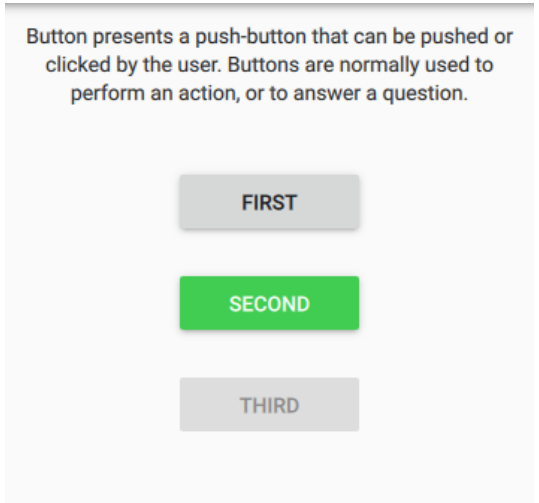


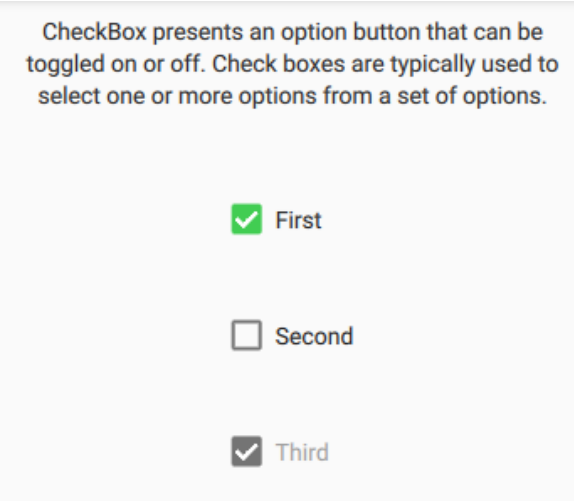


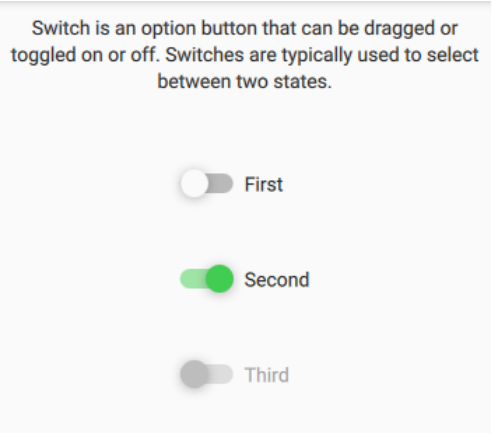
Buttons

Button Controls

- Button
- CheckBox
- RadioButton
- Switch
- ...



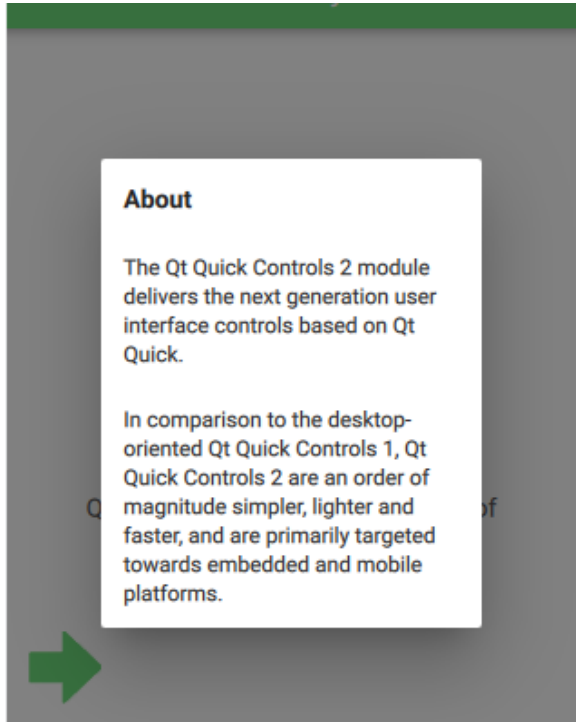


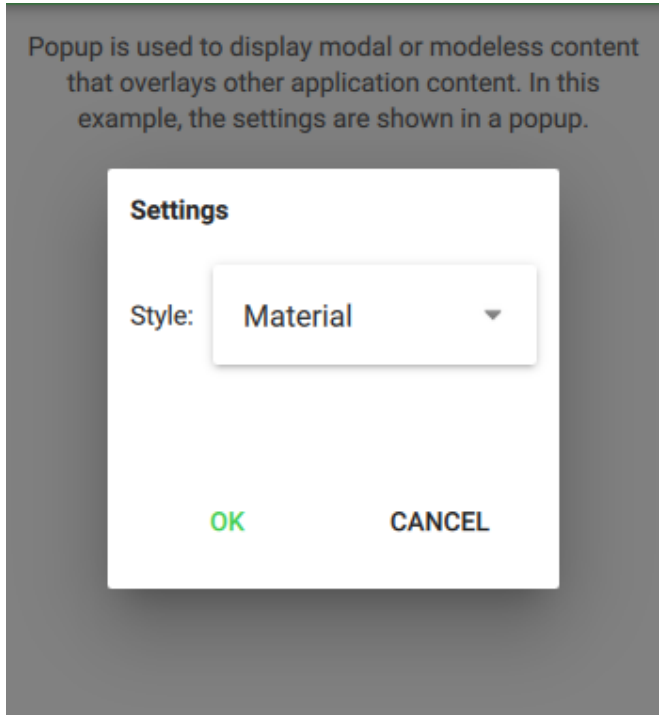


Popup

Popup Controls

- Popup
- Dialog
- Menu
- Drawer
- ...





basysKom

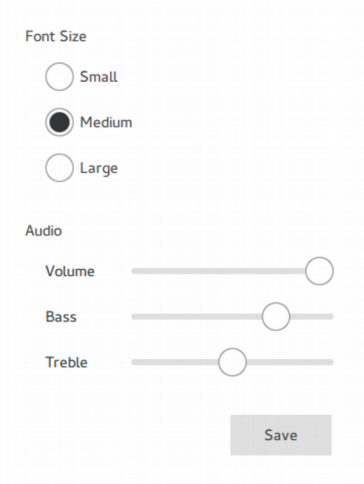
Frank Meerkötter

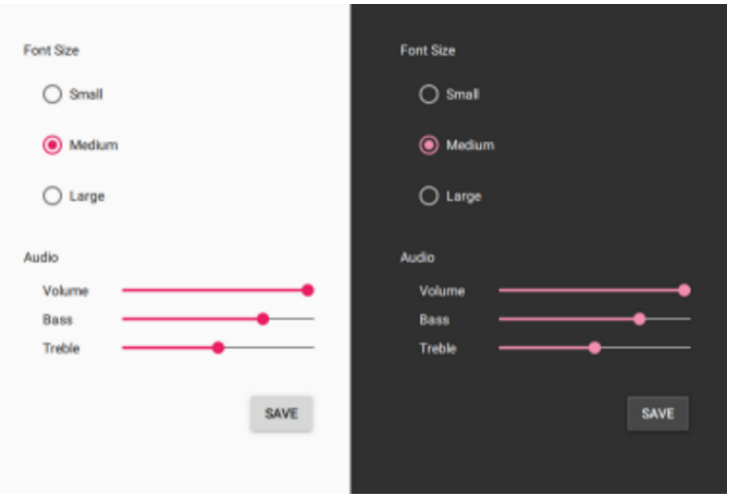
Styling

Styling

Verfügbare Styles

- Default
- Material (Android)
- Universal (Microsoft)

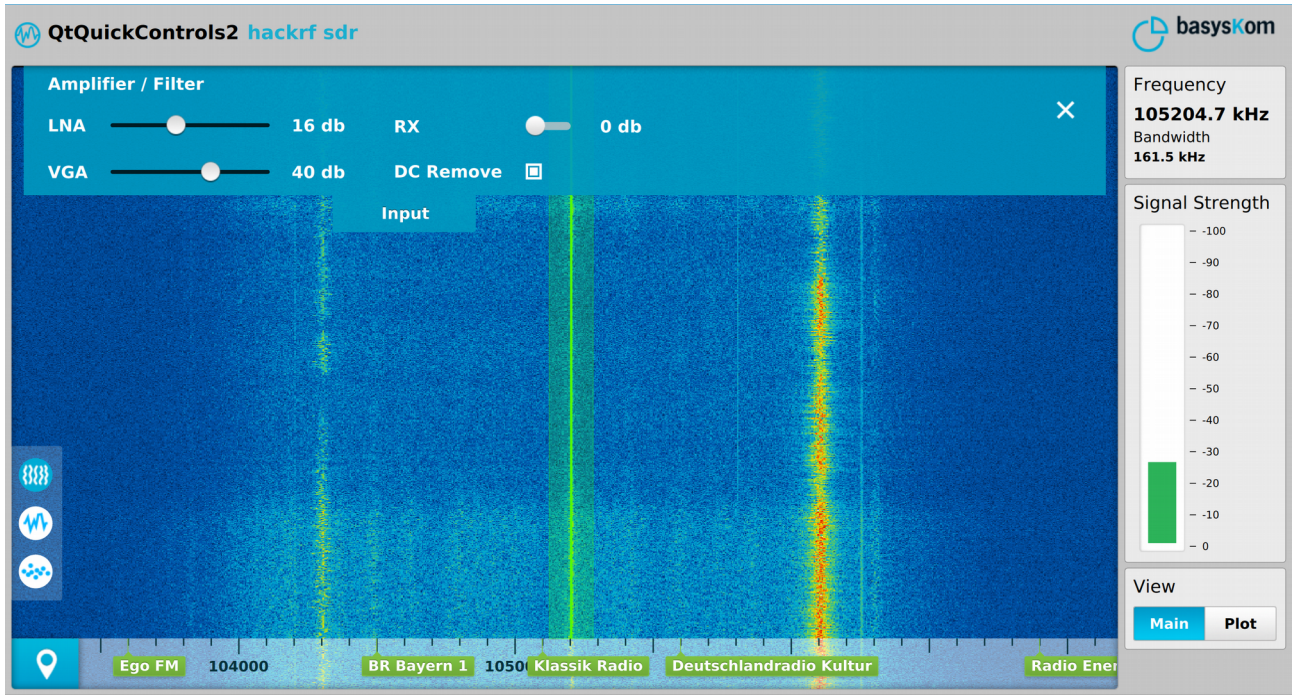




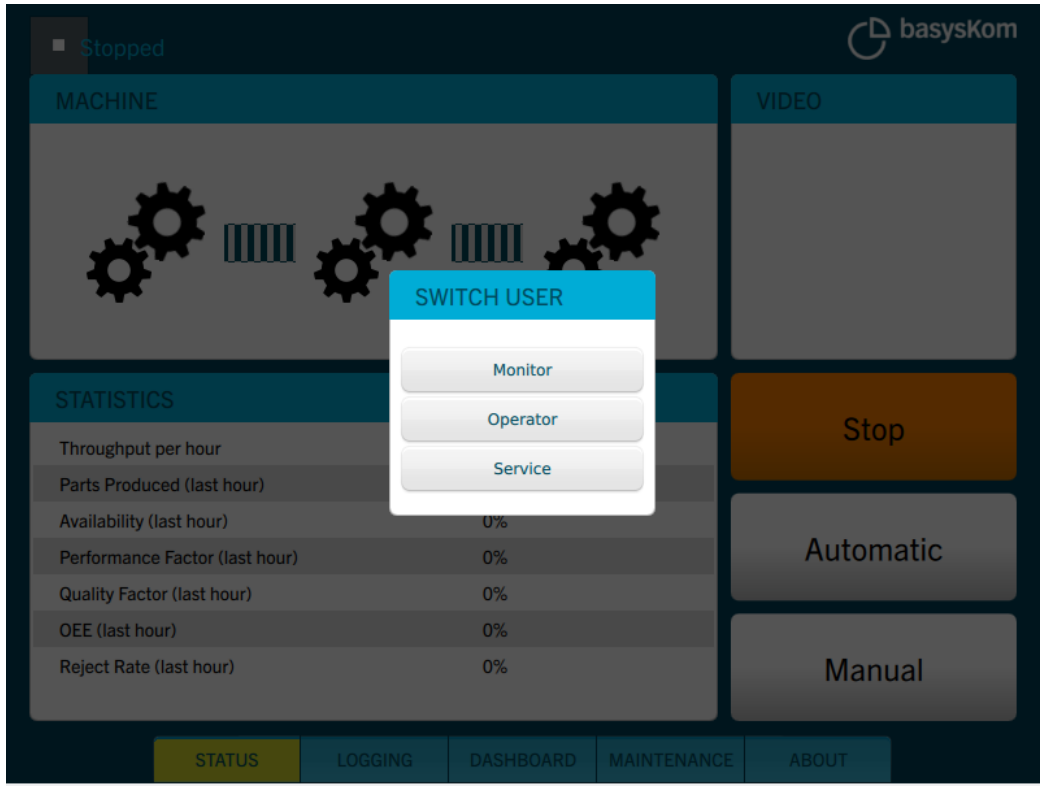
Styling

Hintergründe und Nutzung

- Material und Universal sind keine nativen Styles
- Umsetzungen der jeweiligen Design-Spezifikationen mit den Mitteln der Qt Quick Controls 2
- Testbett für die Entwicklung der QQC2
- Beispiel für die Umsetzung eigener Styles



Styling – Beispiel (Live Demo)



Styling – Beispiel

Ansätze

- Inline
- Eigene Komponenten
- Templates

```
import QtQuick 2.0
import QtQuick.Controls 2.0
import com.basyskom.industrialhmi.style 1.0

Button {
    id: root

    background: BorderImage {
        source: root.down || root.checked ?
                    "qrc:/assets/images/buttonPressed.png" :
                    "qrc:/assets/images/button.png"
        anchors.fill: parent
        border.left: 10; border.top: 10
        border.right: 10; border.bottom: 10
    }

    contentItem: Text {
        text: root.text
        horizontalAlignment: Text.AlignHCenter
        verticalAlignment: Text.AlignVCenter
        color: root.down || root.checked ?
            BkStyle.bkButtonTextHighlight:
            BkStyle.bkButtonTextDefault
    }
}
```

basysKom

Entwicklerforum HMI 2017

Frank Meerkötter

Vergleich Architektur und Performance

Architektur

QtQuick Controls 1

- In QML umgesetzt
- Alle Eventualitäten abdecken zu können als Leitgedanke
  - Der Desktop als primäres Ziel
- Platform-natives als auch Custom-Styling

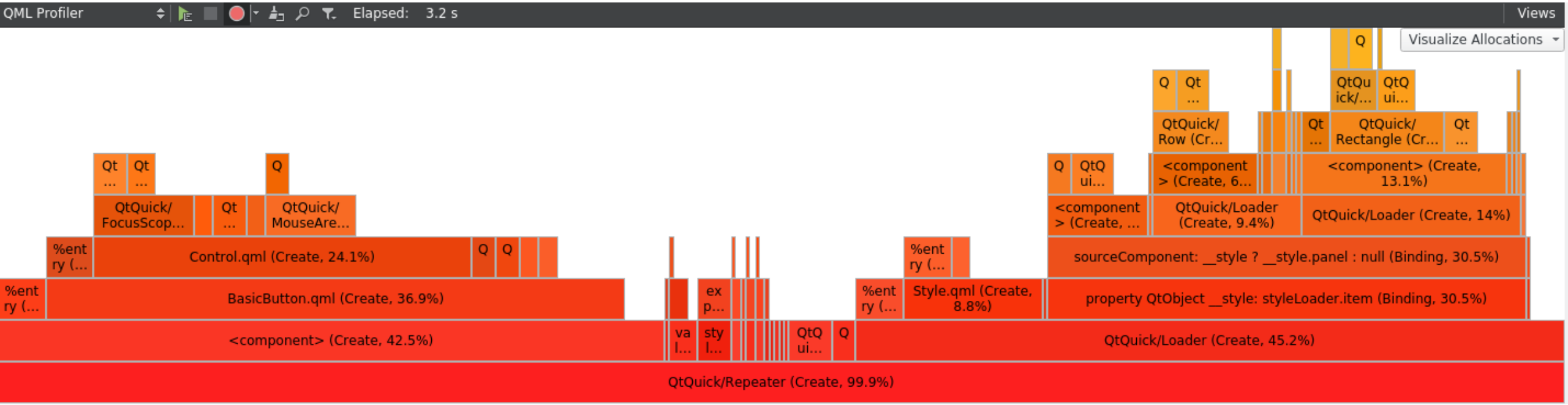
QtQuick Controls 2

- Bessere Trennung zwischen Darstellung und Logik
  - Darstellung ist in QML umgesetzt
  - UI-Logik/State sind in C++ umsetzt
- Einfachheit und Performance als Leitgedanken
  - Mobile und Embedded als (primäre) Ziele
- Kein Platform-natives Styling

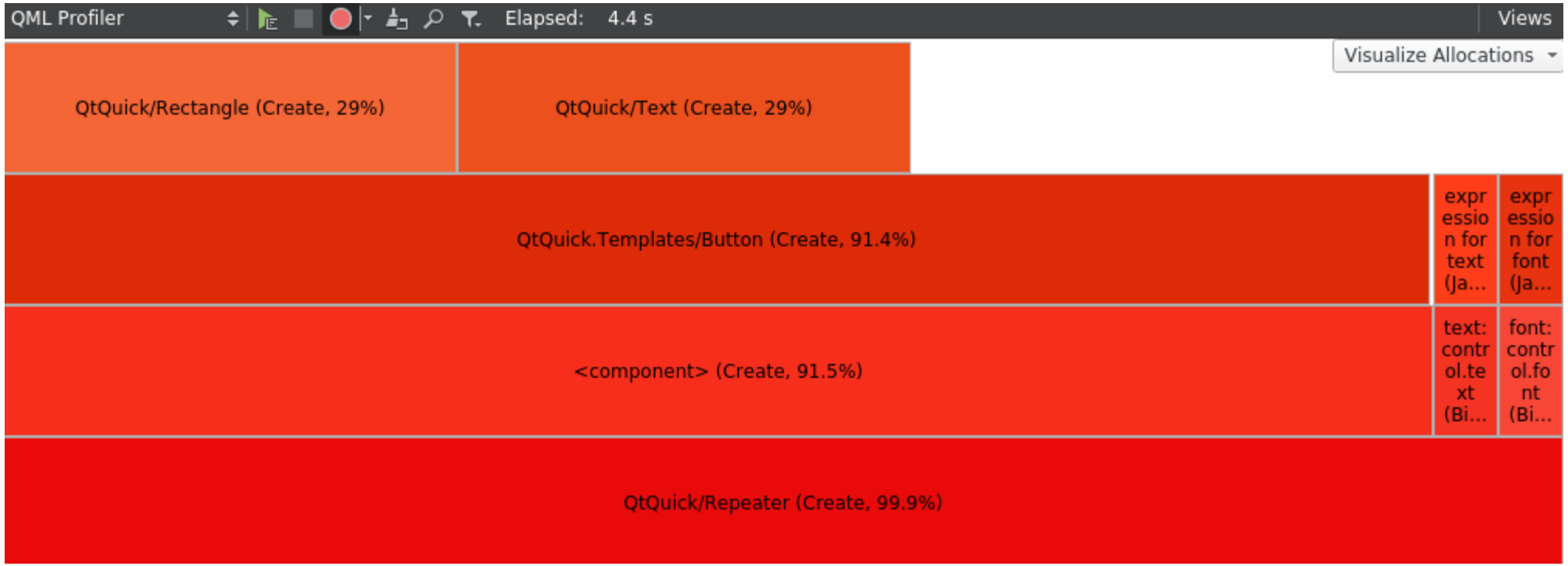
Anwendungsbeispiel

```
import QtQuick 2.7
import QtQuick.Window 2.2
import QtQuick.Controls 2.0

Window {
    visible: true; width: 640; height: 480

    Repeater {
        model: 1000
        Button {
            text: "HMIForum 2017"
        }
    }
}
```

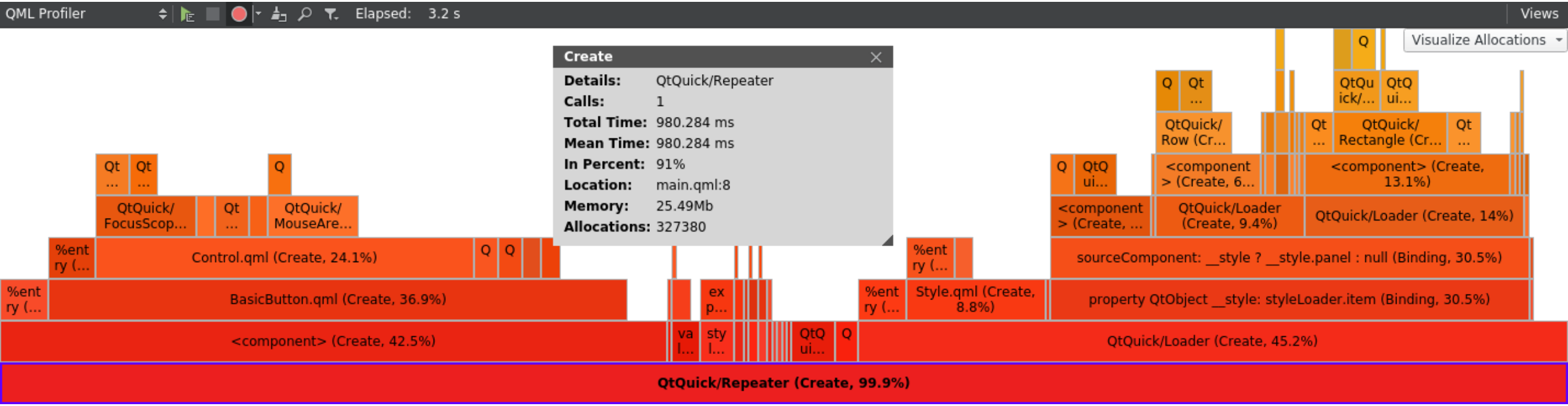
Architektur – QtQuick Controls 1



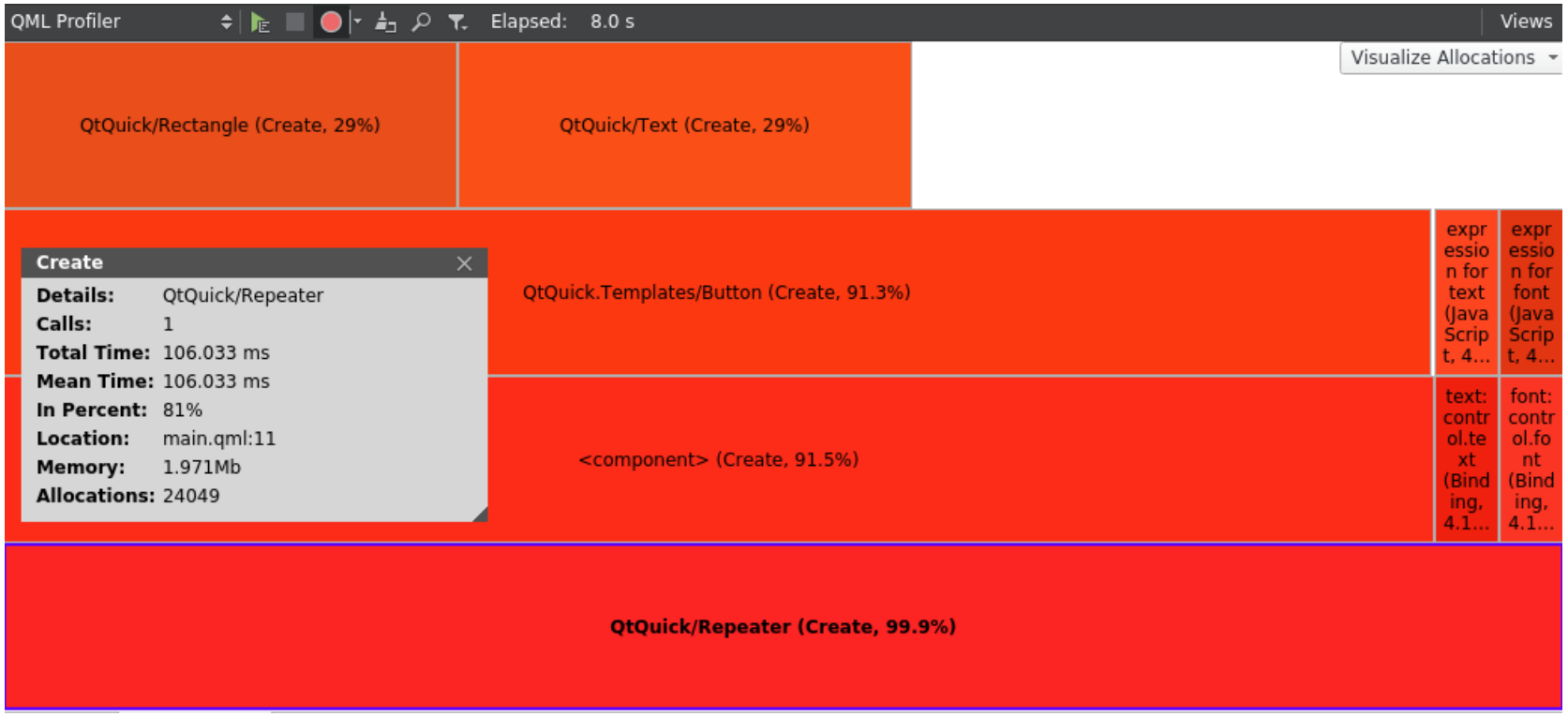
Architektur – QtQuick Controls 2



Performance – QtQuick Controls 1



Performance – QtQuick Controls 2



Fallstricke

(Noch) nicht vorhanden

- TableView
- TreeView
- SplitView

Mix von QQC1 und 2 ist möglich, aber

- Kein High-DPI Support in den Qt Quick Controls 1
- Vorsicht bzgl. Name-Clashes

Zusammenfassung & Ausblick

Roadmap

QtQuick Controls 2.0

- Qt5.7 (2016)
- Initialer Release

QtQuick Controls 2.1

- Qt5.8 (2016)
- Neue Controls: Dialog, DialogButtonBox & Menu/ToolSeparator

QtQuick Controls 2.2

- Qt 5.9 (Mai 2017)
- Neue Controls: ScrollView & DelayButton
- Multitouch-Support
- Actions

Zusammenfassung

Embedded HMIs/Mobile

- Speicherverbrauch und Performance sind deutlich besser als bei den QQC1

Styling

- Saubere Trennung von Darstellung und Logik
- Gut zu Stylen

basysKom

EMBEDDED SOFTWARE ENGINEERING

Consulting, Development & Coaching