Using the Qt Scene Graph from C++ with QSkinny

https://github.com/uwerat/qskinny

Who am I?

» working on Qt since 2008

» former QtNetwork maintainer

» @peha23 on Twitter

https://github.com/uwerat/qskinny

What is this talk about?

Using the Qt graphic stack from C++

https://github.com/uwerat/qskinny

Agenda

1. QML under the hood
2. The QML / C++ boundary
3. QSkinny
4. Outlook

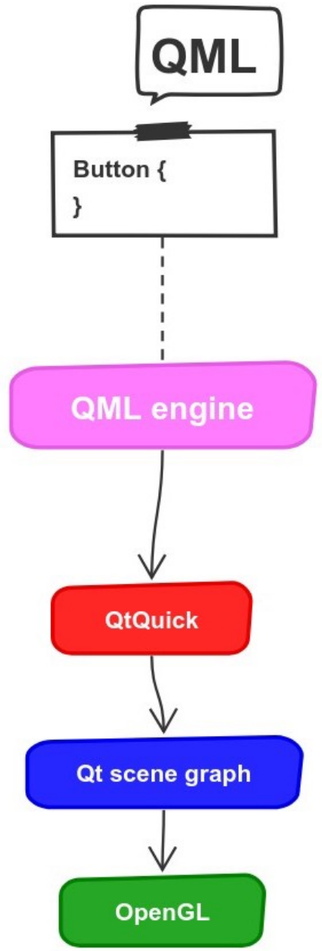
https://github.com/uwerat/qskinny

Agenda

1. QML under the hood
2. The QML / C++ boundary
3. QSkinny
4. Outlook

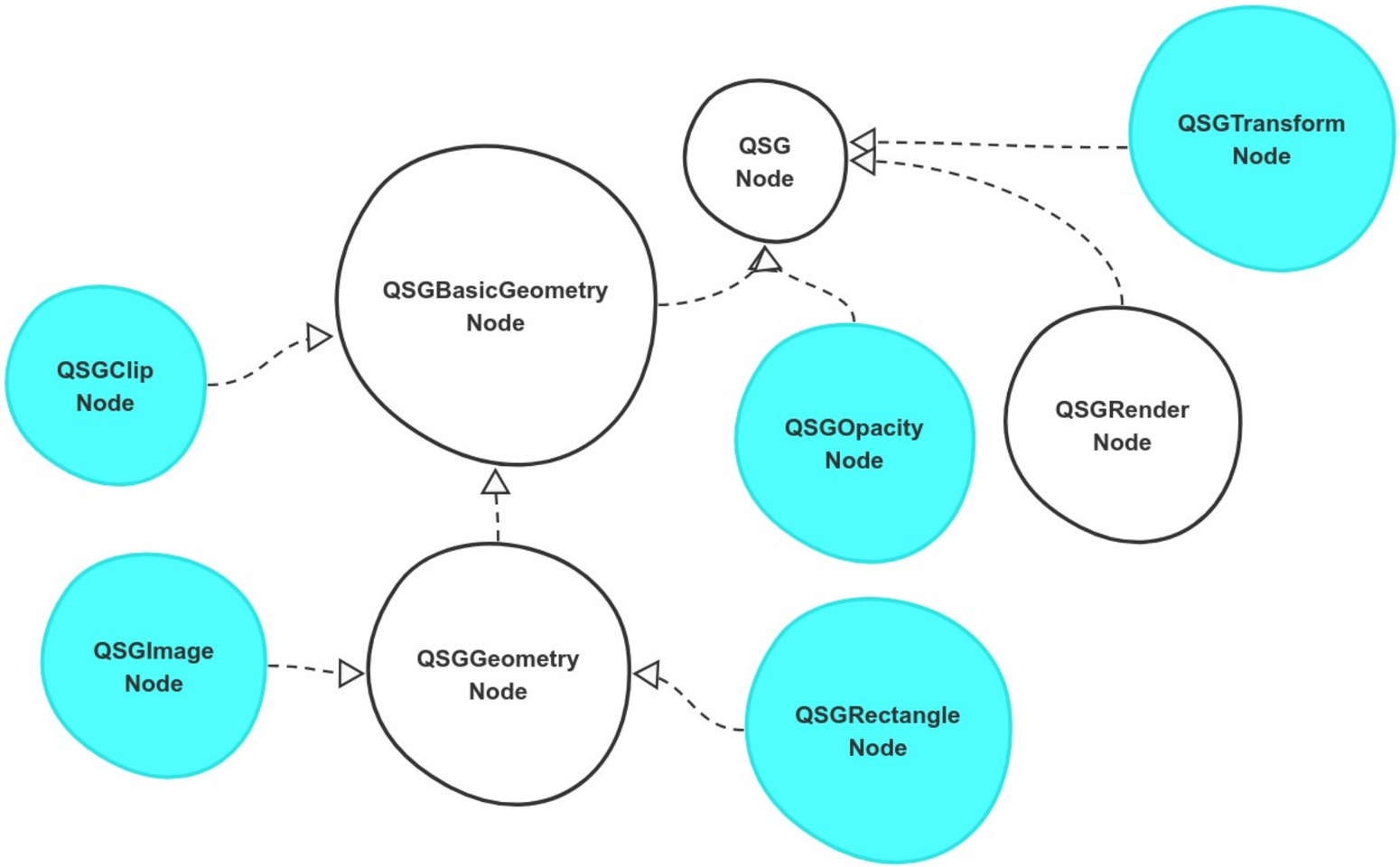
https://github.com/uwerat/qskinny

QML under the hood



https://github.com/uwerat/qskinny

Types of scene graph nodes

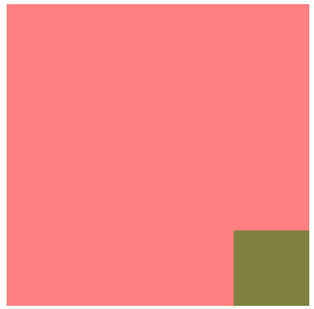


https://github.com/uwerat/qskinny

QML example

```
Rectangle {
    id: outterRectangle
    width: 200
    height: 200
    color: "red"
    opacity: 0.5

    Rectangle {
        id: innerRectangle
        width: 50
        height: 50
        clip: true
        anchors.bottom: parent.bottom
        anchors.right: parent.right
        color: "green"
    }
}
```



https://github.com/uwerat/qskinny

QQuickRootItem — is a QQuickItem

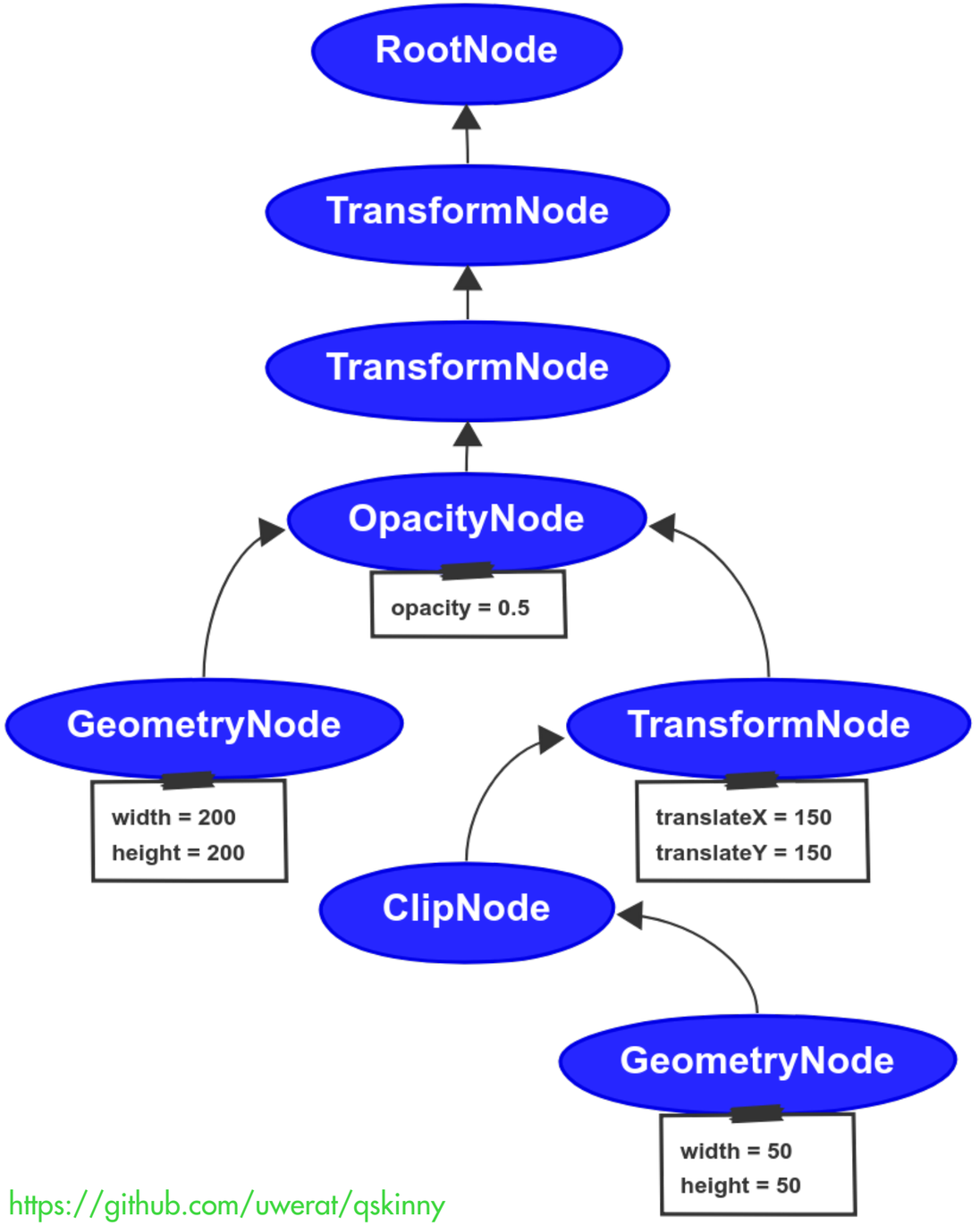
QQuickRectangle — is a QQuickItem

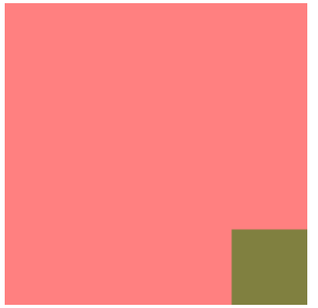
```
qreal x = 0
qreal y = 0
qreal width = 200
qreal height = 200
qreal opacity = 0.5
```

QQuickRectangle — is a QQuickItem

```
qreal x = 150
qreal y = 150
qreal width = 50
qreal height = 50
```

https://github.com/uwerat/qskinny



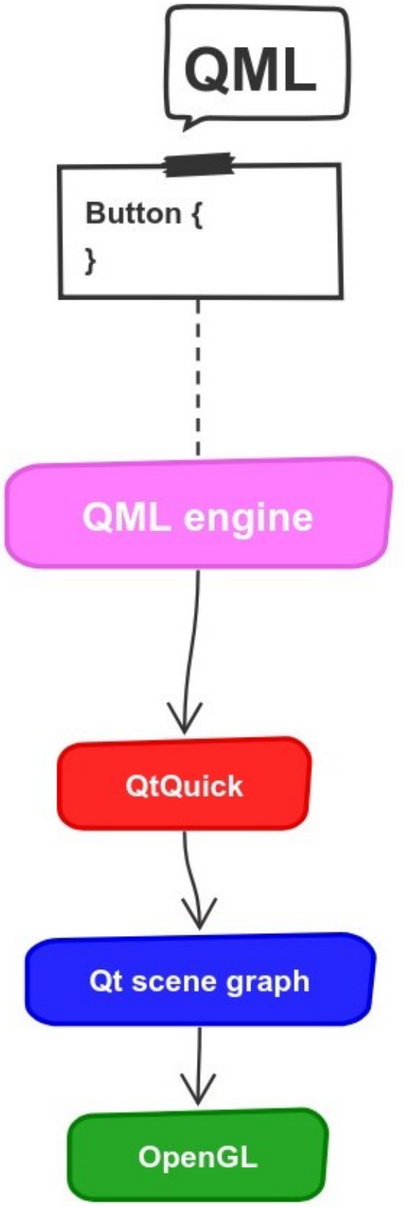


https://github.com/uwerat/qskinny

Agenda

1. QML under the hood
2. The QML / C++ boundary
3. QSkinny
4. Outlook

https://github.com/uwerat/qskinny



https://github.com/uwerat/qskinny

QtQuickControls 1

Write everything in QML

```
Control {
    id: slider
    (...)
    style: Settings.styleComponent(Settings.style, "SliderStyle.qml", slider)
    property Component tickmarks: Repeater {
    Rectangle {
        color: "#777"
        width: 1
        height: 3
        y: (...)
        x: (...)
        }
    }
}
```

https://github.com/uwerat/qskinny

QtQuickControls 2

some parts QML, some C++

qquickslider_p.h:

```
class Q_QUICKTEMPLATES2_PRIVATE_EXPORT QQuickSlider : public QQuickControl
{
    Q_OBJECT
    Q_PROPERTY(qreal from READ from WRITE setFrom NOTIFY fromChanged FINAL)
    Q_PROPERTY(qreal to READ to WRITE setTo NOTIFY toChanged FINAL)
    (...)
};
```

Slider.qml:

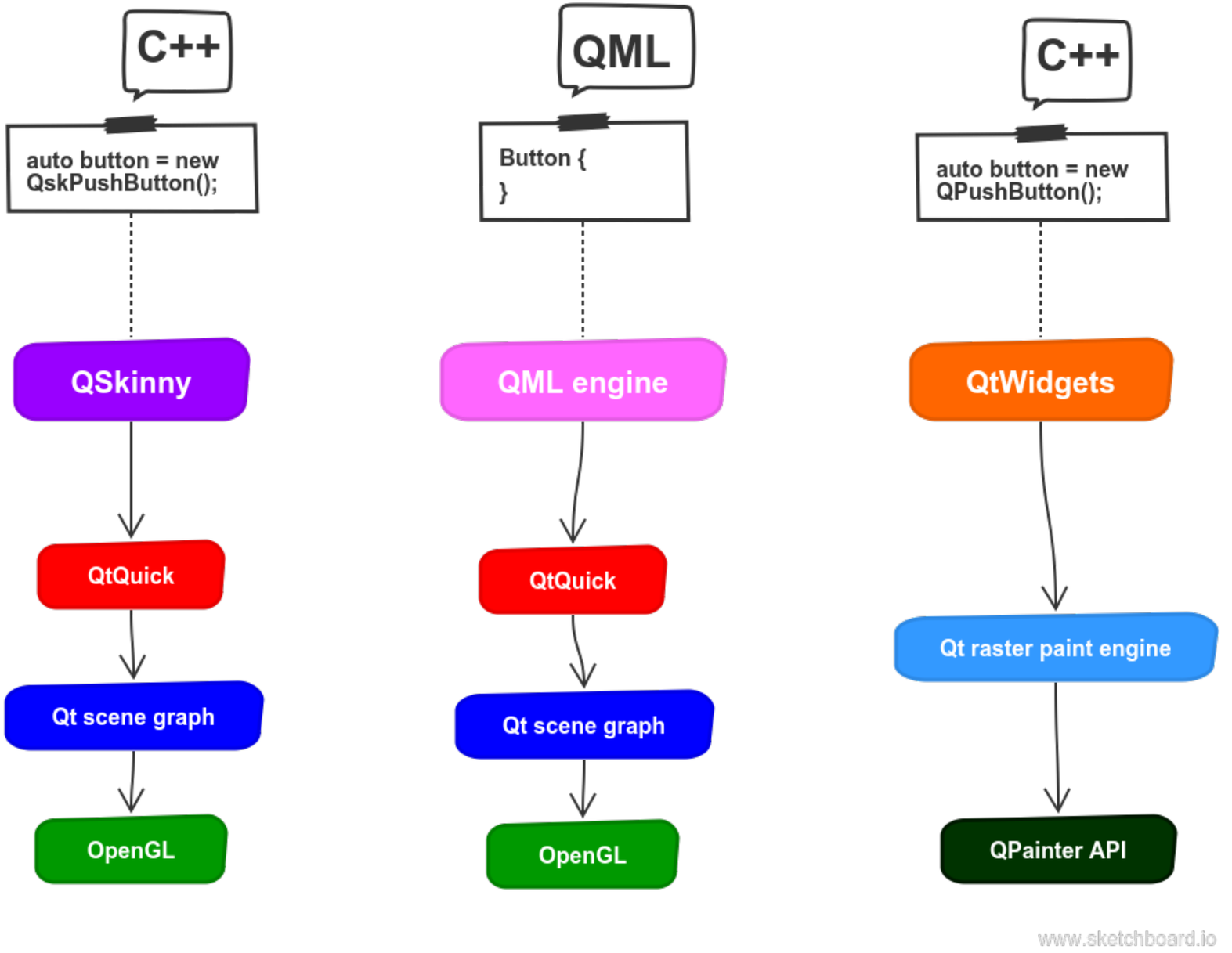
```
T.Slider {
    id: control
}
```

https://github.com/uwerat/qskinny

Agenda

1. QML under the hood
2. The QML / C++ boundary
3. QSkinny
4. Outlook

https://github.com/uwerat/qskinny



https://github.com/uwerat/qskinny

QSkinny design goals

» lightweight

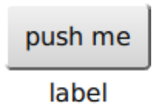
» flexible theming

» dynamic sizing

https://github.com/uwerat/qskinny

QSkinny API

```
QskWindow window;
auto box = new QskLinearBox(Qt::Vertical);
auto button = new QskPushButton("push me", box);
auto label = new QskTextLabel("label", box);
window.addItem(box);
window.show();
```



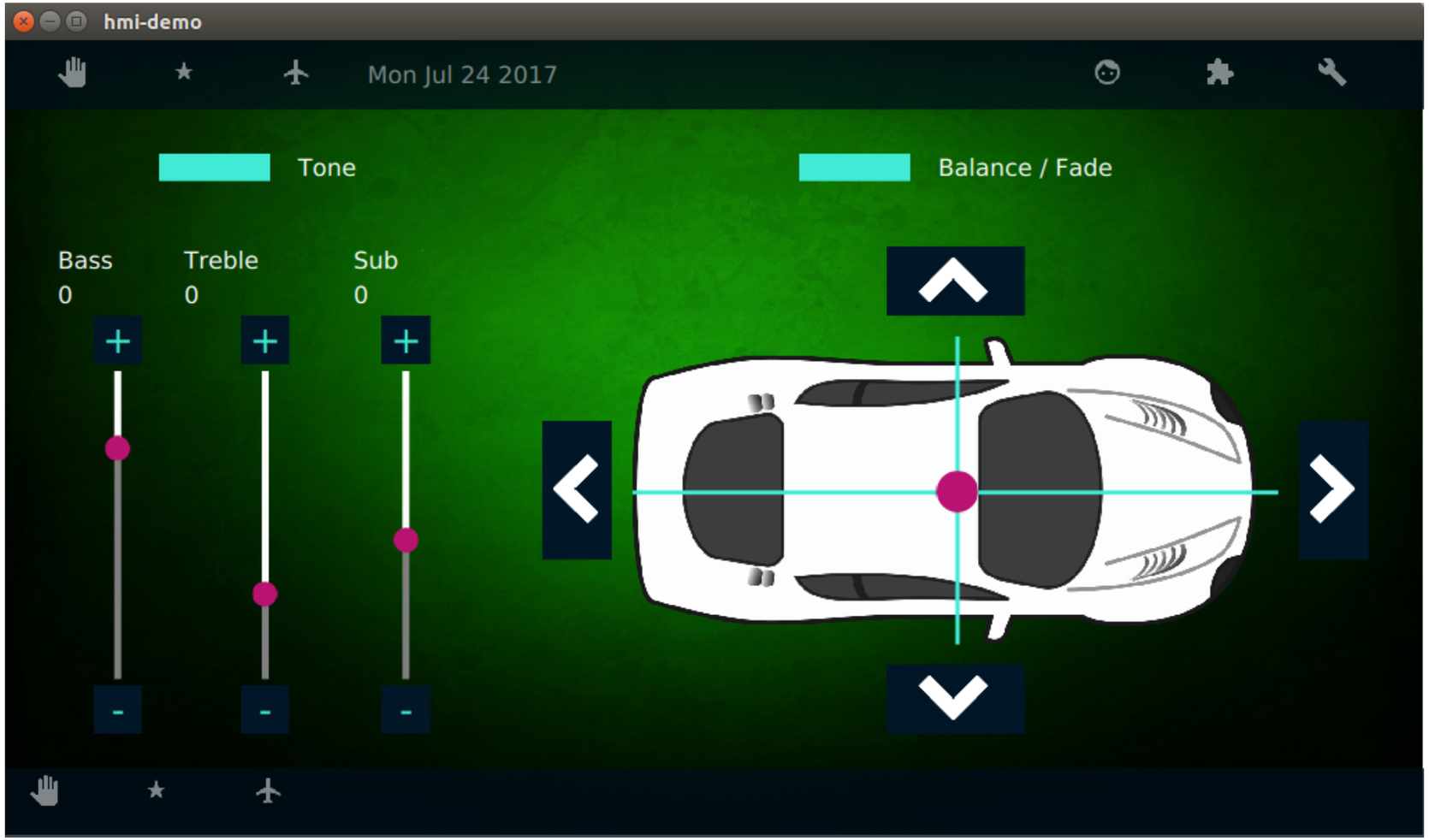
https://github.com/uwerat/qskinny

separation of content and style

(here diagram of skinlet etc.)

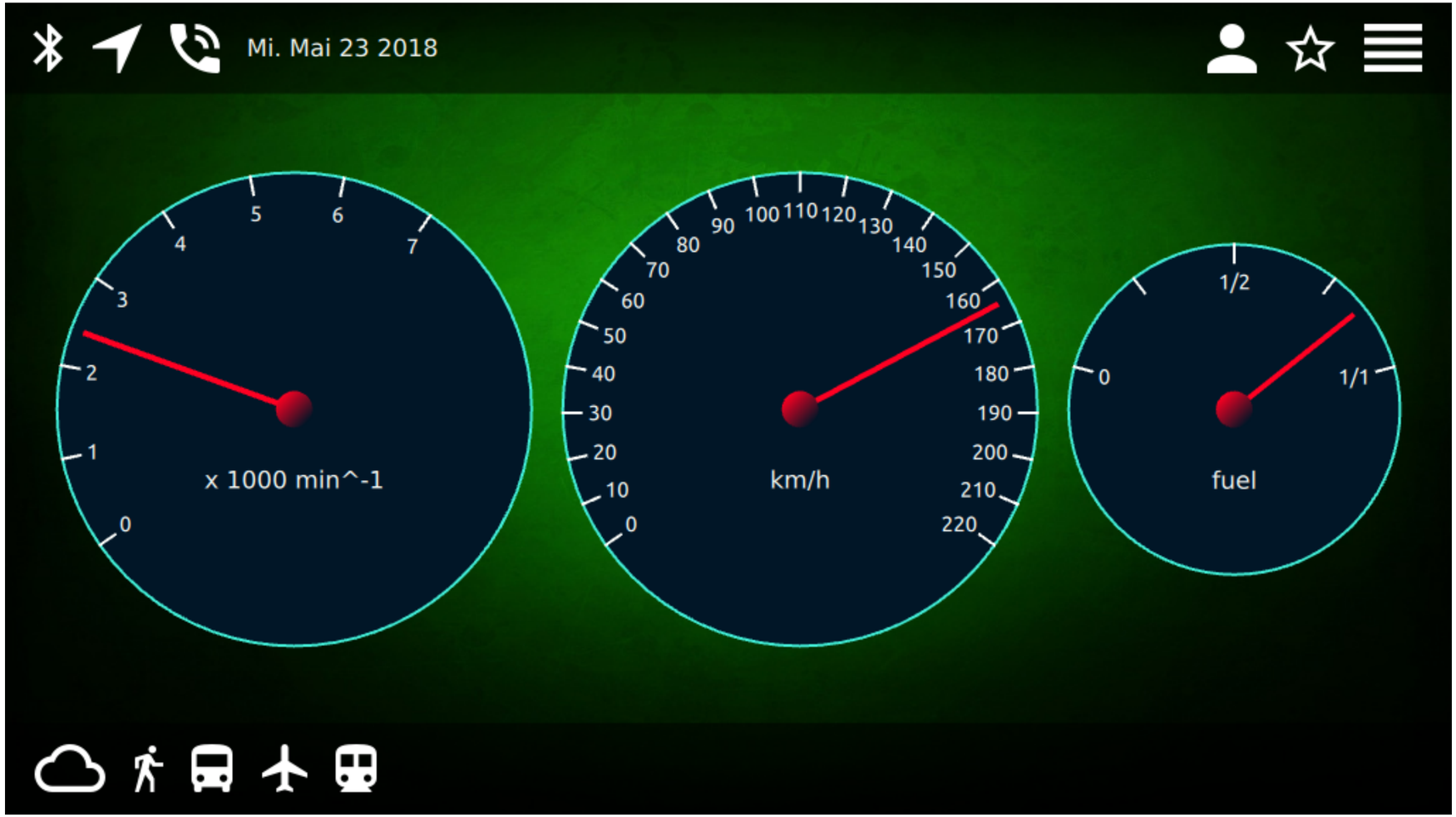
https://github.com/uwerat/qskinny

example



https://github.com/uwerat/qskinny

example



https://github.com/uwerat/qskinny

Outlook

QSkinny

polishing / documentation

Qt 6

(maybe) new styling / opening up QtQuickControls 2?

https://github.com/uwerat/qskinny

Discussion

@peha23 on Twitter

https://github.com/uwerat/qskinny