

# Input Handling Update

## What's Coming Up in Qt 6

Shawn Rutledge

[shawn.rutledge@qt.io](mailto:shawn.rutledge@qt.io)

ecloud on #qt-labs, #qt-quick etc.



## About me

- Qt user since ~2004
- The Qt Company - Oslo since 2011
- Pointing devices: touch, Wacom tablets
- Linux/X11 and macOS
- QtPDF
- Qt Quick, Controls and Dialogs



# Agenda

- Goals
- API changes
- Qt Quick
- Demo
- Remaining work
- Q&A



# Goals



4 - 11 September 2020

- Every QInputEvent carries a QInputDevice\*
- Qt::MouseEventSource was not enough
- Widgets, Qt Quick items and handlers keep working the same
- Common event delivery code for all QPointerEvents
- Qt Quick delivery simplified: no more wrappers
- Flickable handles touch (including replay: QTBUG-85607)
- Event objects as lightweight as possible
- Unblock fixing some old bugs
- Wacom tablets supported better
- Multi-seat

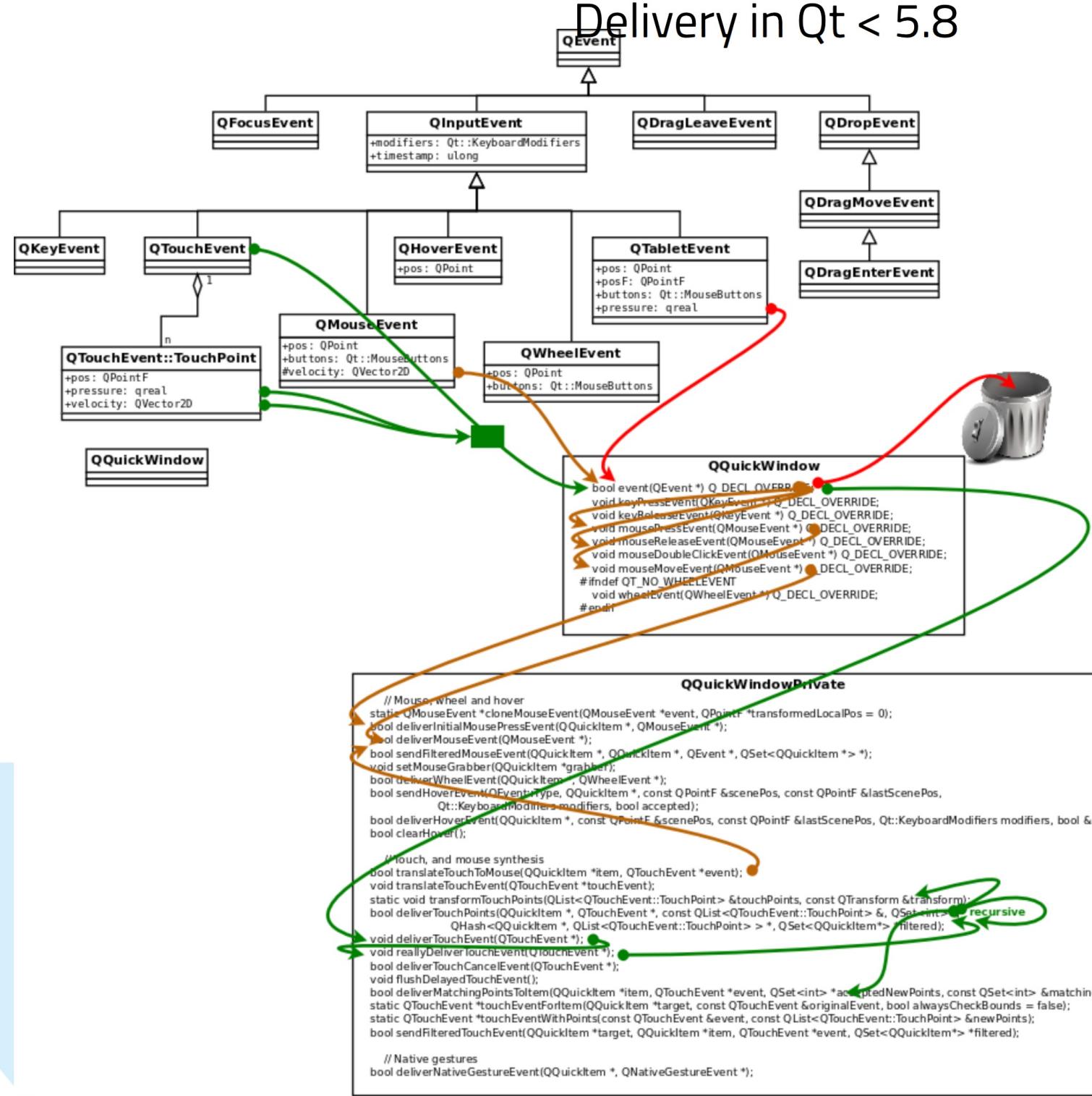
# Initial conditions - Qt Quick



4 - 11 September 2020

- QQuickPointerEvent is a QObject wrapper for QTouchEvent, QMouseEvent etc.
- QQuickEventPoint is a wrapper for QTouchEvent::TouchPoint
  - but every QQuickSinglePointEvent has one too
- QQuickEventPoint stores state between events: exclusive grabber, passive grabbers
- QQuickEventPoint has a parent pointer (QQuickPointerEvent \*event) which is stable
  - So it can be passed alone to any function, C++ or QML, and knows its context
- Delivery code is simplified by these wrappers
- Wrappers were meant as a prototype for QInputEvent refactoring

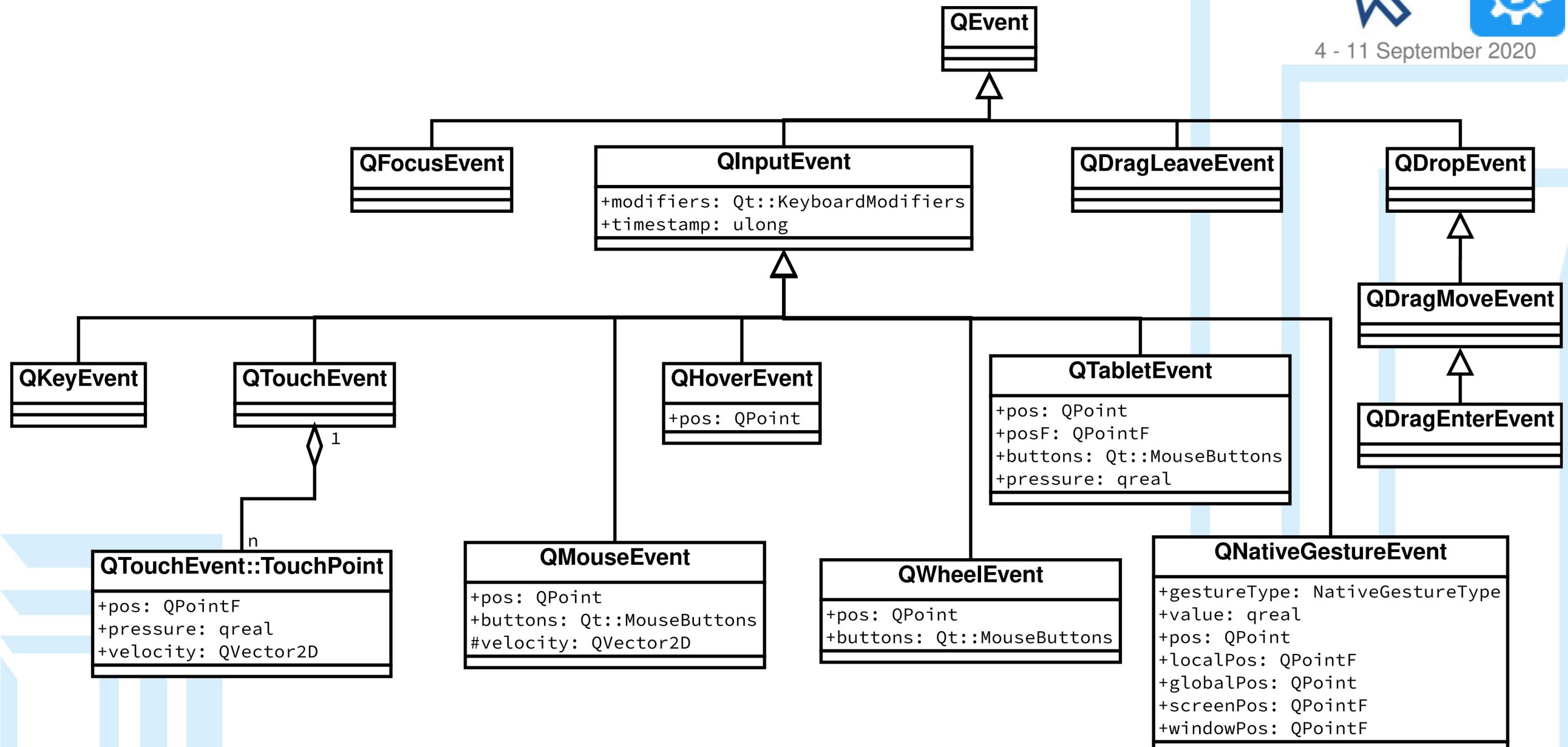
# Delivery in Qt < 5.8



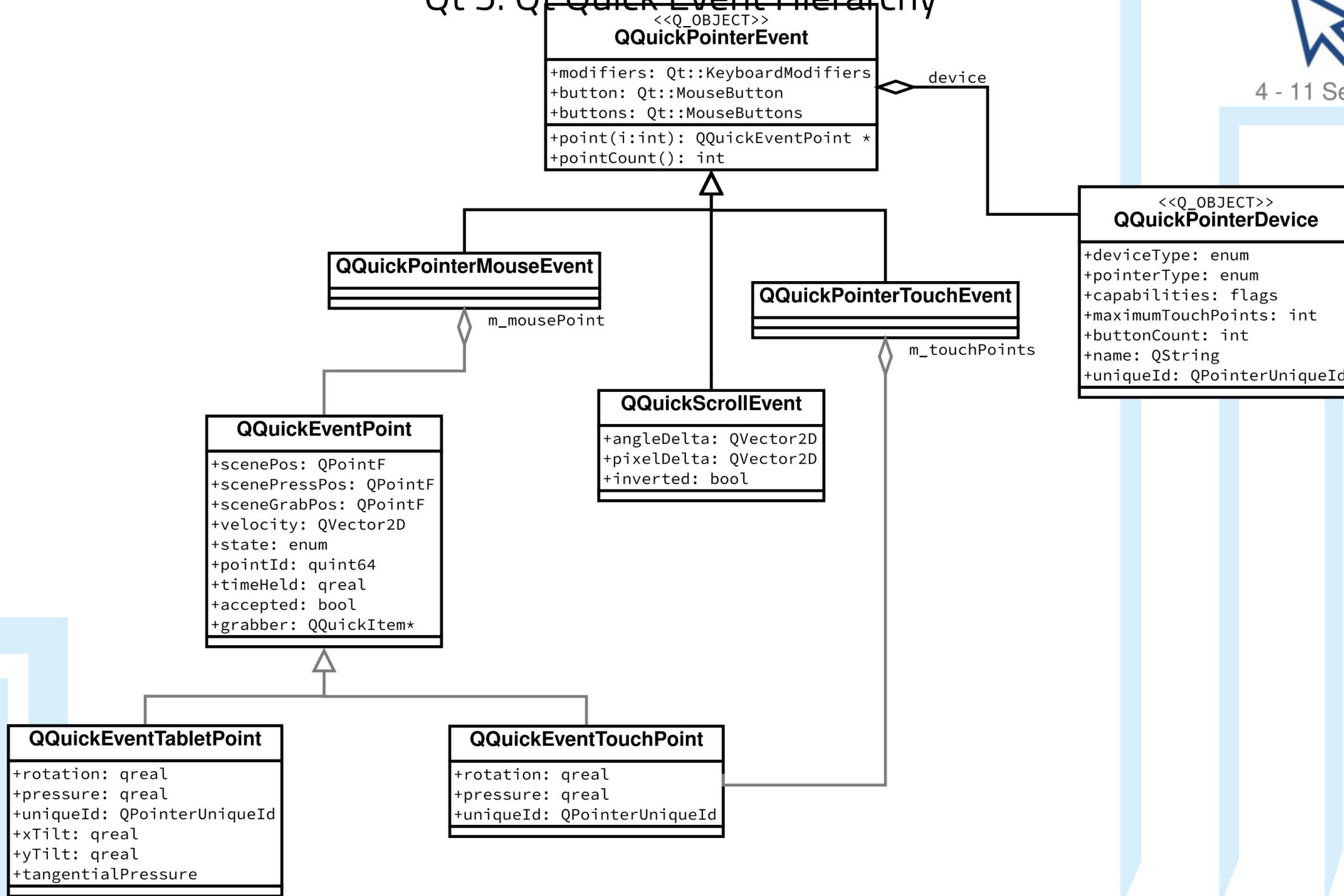
# Qt 5: QEvent hierarchy



4 - 11 September 2020



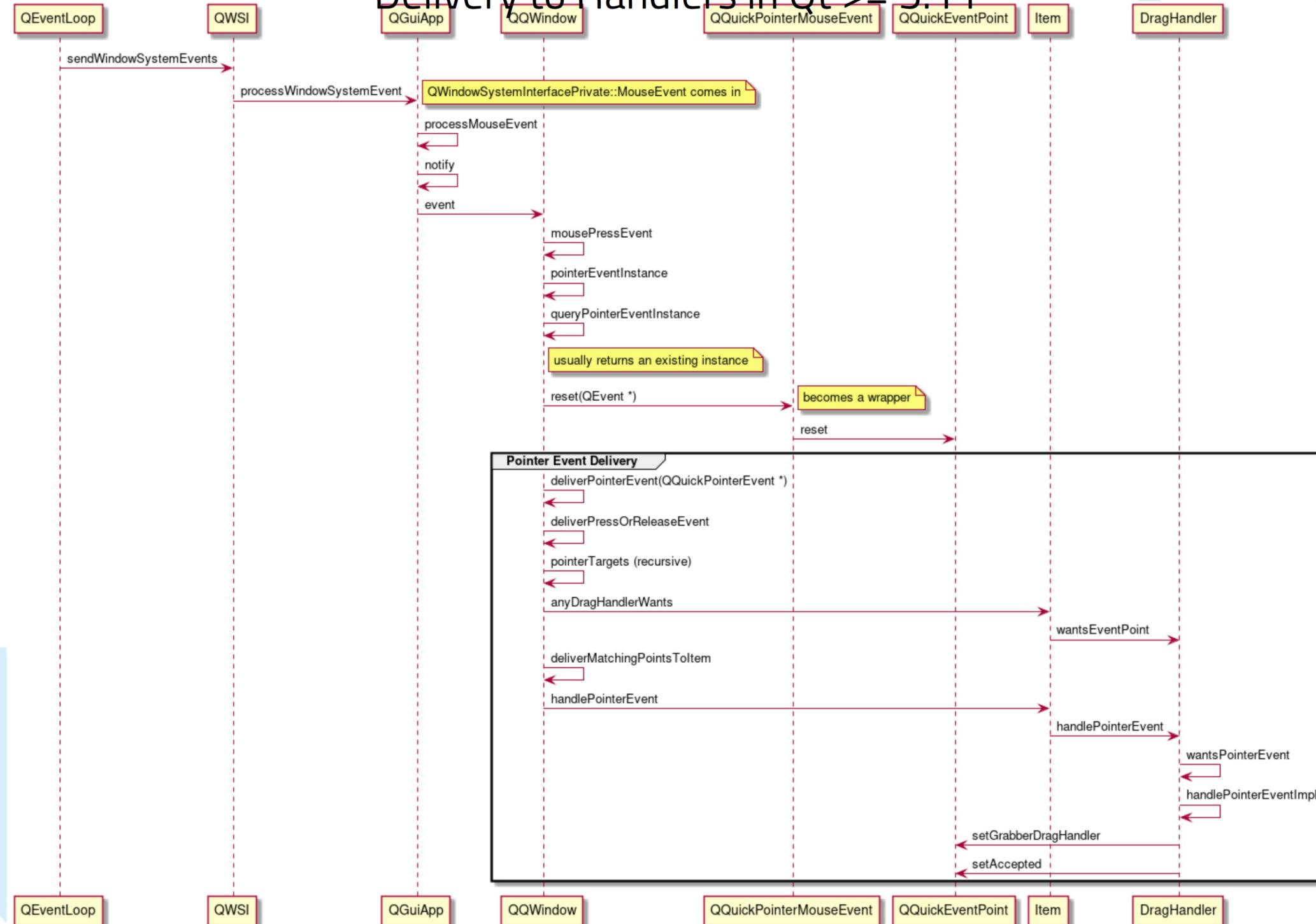
# Qt 5: Qt Quick Event Hierarchy



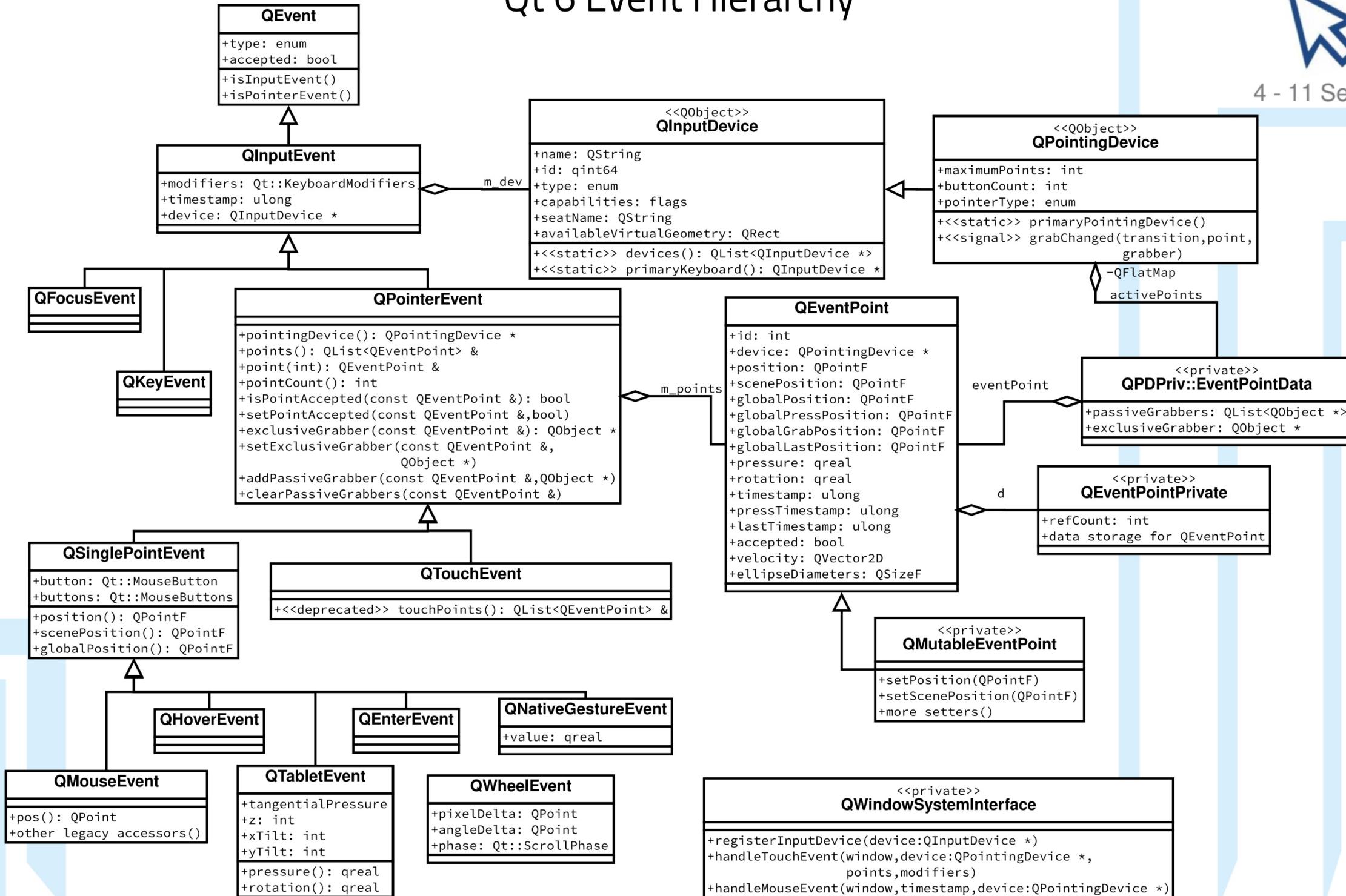
# Delivery to Handlers in Qt >= 5.11



September 2020



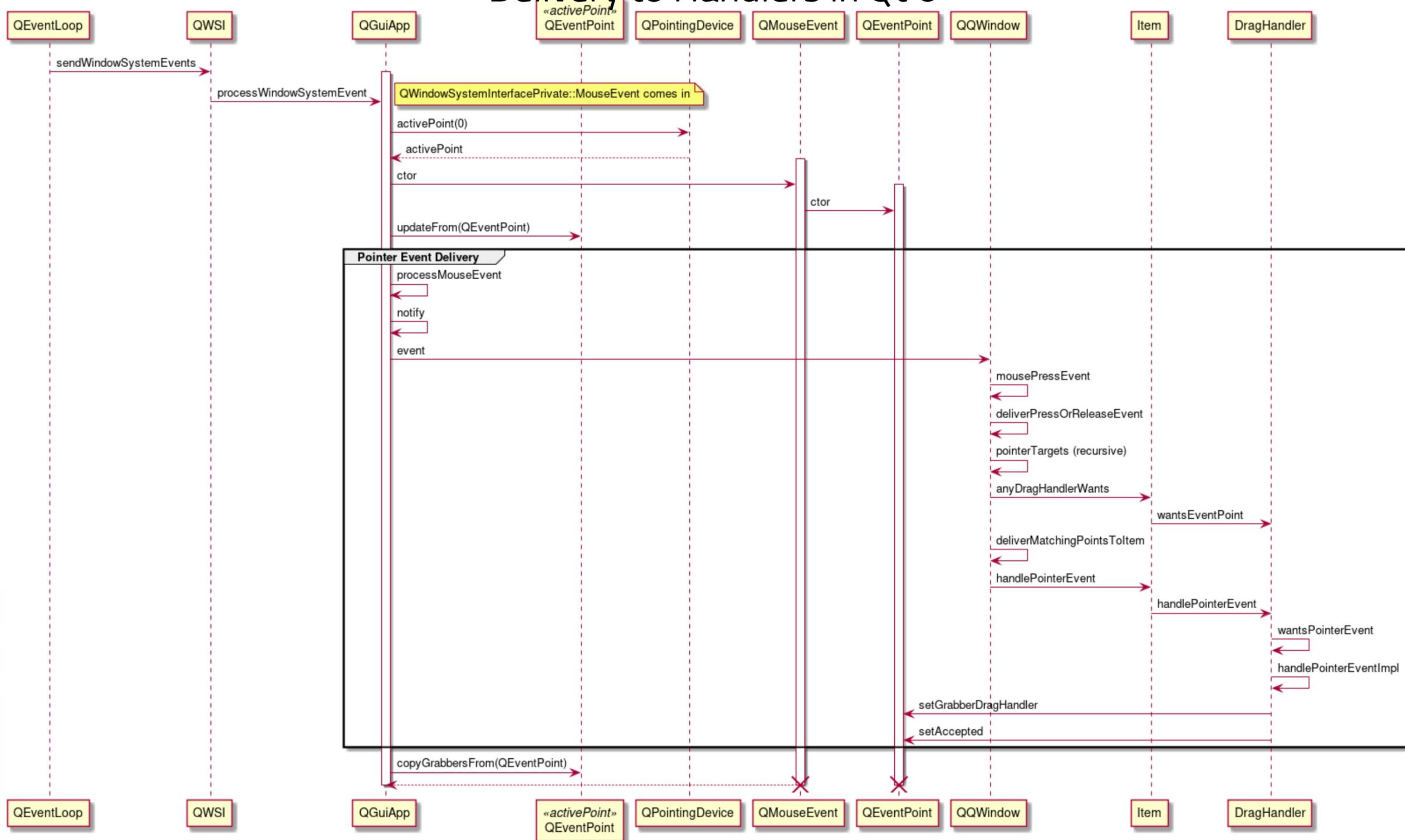
# Qt 6 Event Hierarchy





er 2020

# Delivery to Handlers in Qt 6

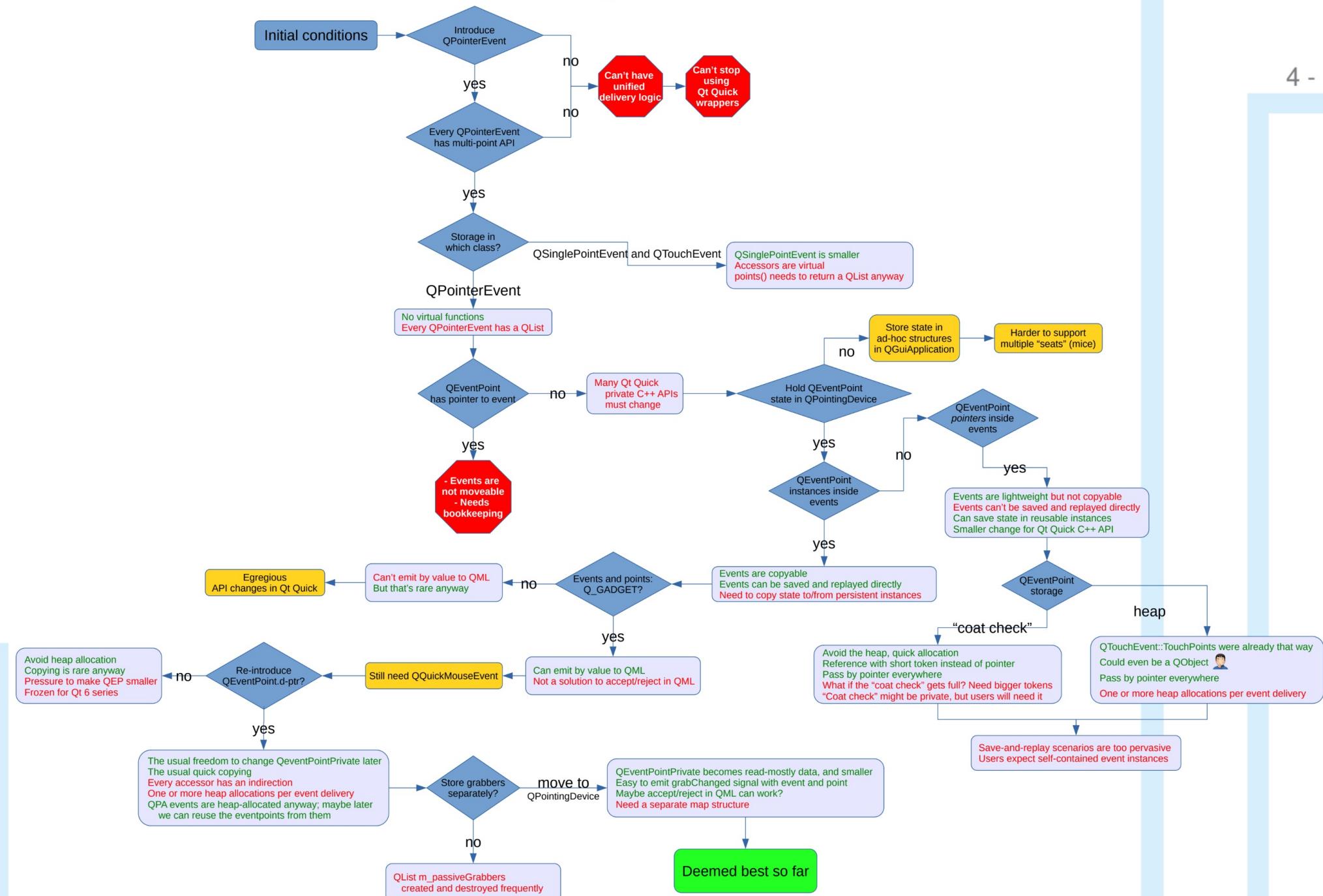




4 - 11 September 2020

- QTouchEvent::TouchPoint is heap-allocated and has a PIMPL in spite of being temporary
- QPA events (and touchpoints) are special unrelated mostly-duplicate classes
- QPA events are heap-allocated in spite of being temporary
- All QInputEvents are stack-allocated in QGuiApplication, and go out of scope after delivery
- QInputEvent subclasses have duplicated but incompatible API
- QTouchEvent can have multiple points

# Design decisions



4 - 11 September 2020

# Agnosticism

```
1:  bool event(QEvent *ev) override
2: {
3:     if (ev->isPointerEvent() && static_cast<QPointerEvent *>(event)->isPressEvent()) {
4:         for (QEventPoint &point : event->points()) {
5:             if (reactToPress(point.position()))
6:                 point.setExclusiveGrabber(this);
7:         }
8:     }
9:     return true;
10: }
11: return false;
12: }
13:
```



4 - 11 September 2020

# Is it a synthesized mouse event?



4 - 11 September 2020

```
1: 
2: void mousePressEvent(QMouseEvent *event) override
3: {
4:     qDebug()
5:         // The oldest API - unusable
6:         << "spontaneous" << event->spontaneous()
7: 
8:         // Qt::MouseEventSource : since Qt 5.4
9:         << "source" << event->source()
10:        << "actual mouse?" << (event->source() == Qt::MouseEventNotSynthesized)
11:        << "perhaps from touch or tablet?" << (event->source() == Qt::MouseEventSynthesizedByQt)
12: 
13:        // Similar to the QML Pointer Handlers acceptedDevices API (since 5.10)
14:        << "device type" << event->pointingDevice()->type()
15:        << "from touchscreen?" << (event->pointingDevice()->type() == QInputDevice::DeviceType::TouchScreen)
16:        << "from touchpad?" << (event->pointingDevice()->type() == QInputDevice::DeviceType::TouchPad)
17: 
18:        // Similar to the QML Pointer Handlers acceptedPointerTypes API (since 5.10)
19:        << "pointer type" << event->pointingDevice()->pointerType()
20:        << "from finger?" << (event->pointingDevice()->pointerType() == QPointingDevice::PointerType::Finger)
21:        << "from eraser?" << (event->pointingDevice()->pointerType() == QPointingDevice::PointerType::Eraser);
22: }
23:
```

# PointHandler



4 - 11 September 2020

```
import QtQuick 6.0

// a simplified version of qtdeclarative/tests/manual/pointer/singlePointHandlerProperties.qml
Item {
    id: root
    width: 480
    height: 480

    PointHandler {
        id: pointHandler
        target: null
        acceptedButtons: Qt.AllButtons
    }

    Item {
        id: crosshairs
        x: pointHandler.point.position.x - width / 2
        y: pointHandler.point.position.y - height / 2
        width: parent.width / 2; height: parent.height / 2
        visible: pointHandler.active
        rotation: pointHandler.point.rotation

        Rectangle {
            color: "tomato"
            anchors.centerIn: parent
            width: 2; height: parent.height
            antialiasing: true
        }
        Rectangle {
            color: "tomato"
        }
    }
}
```

## Stuff left to work on

- Flickable: make it flick nicer
- Flickable: event replay
- rename QPointingDevice to QPointerDevice?
- get rid of QPointingDevice::pointerType?





4 - 11 September 2020

# Input Handling Update

## What's Coming Up in Qt 6

Shawn Rutledge

`shawn.rutledge@qt.io`

ecloud on #qt-labs, #qt-quick etc.

