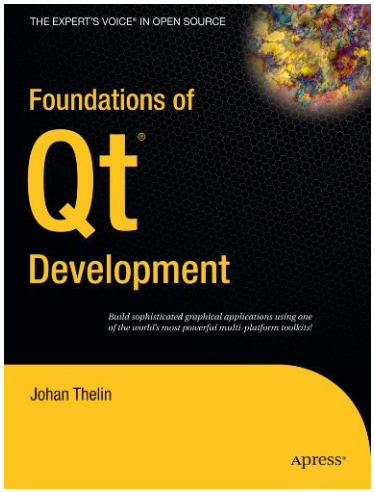
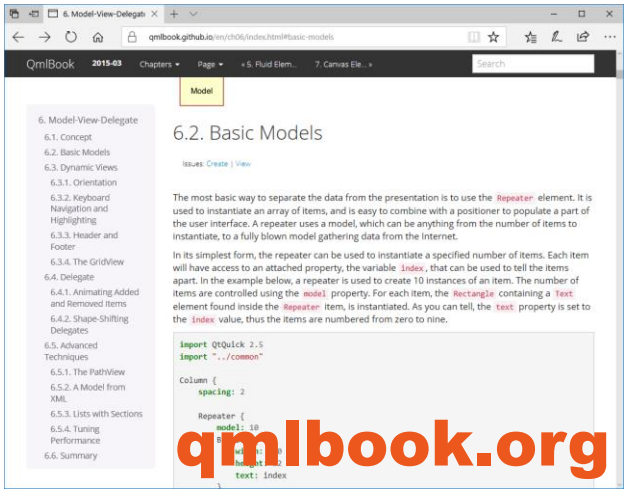
Qt in Automotive

FOSDEM // Brussels 2018

Introduction





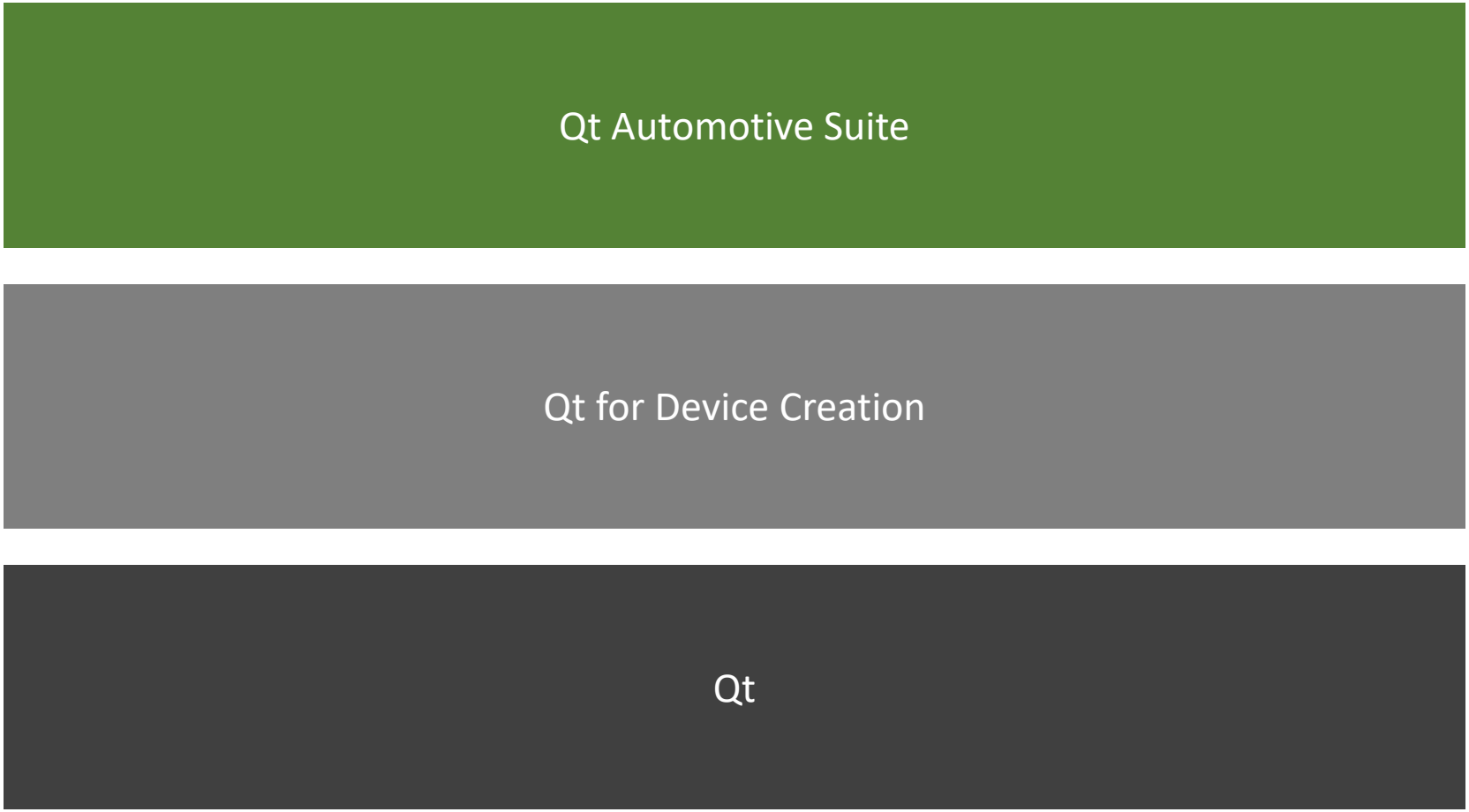
- Johan Thelin
- Qt, Embedded Linux, Luxoft, Pelagicore, Nokia Qt


122

What problems do we share?

What solutions do we share?

What is the Qt Auto Suite?

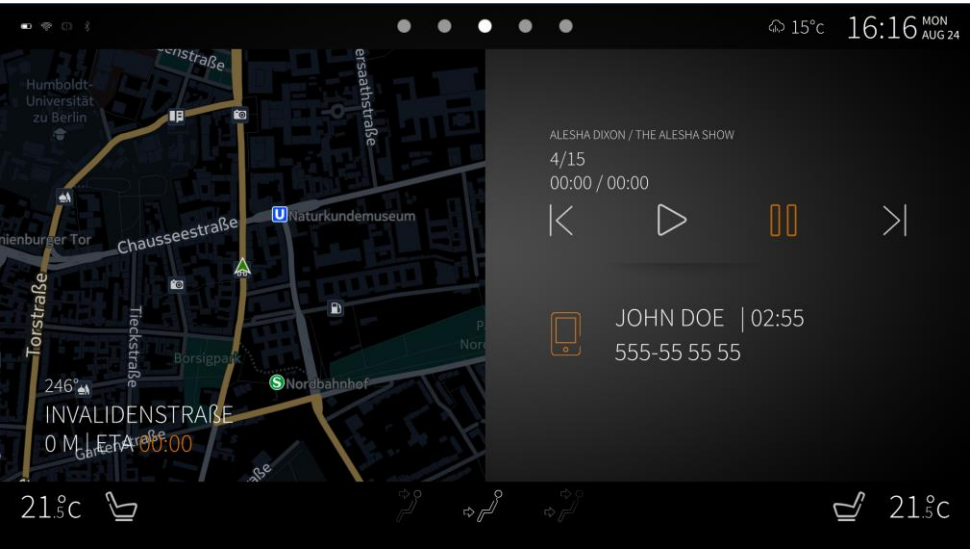


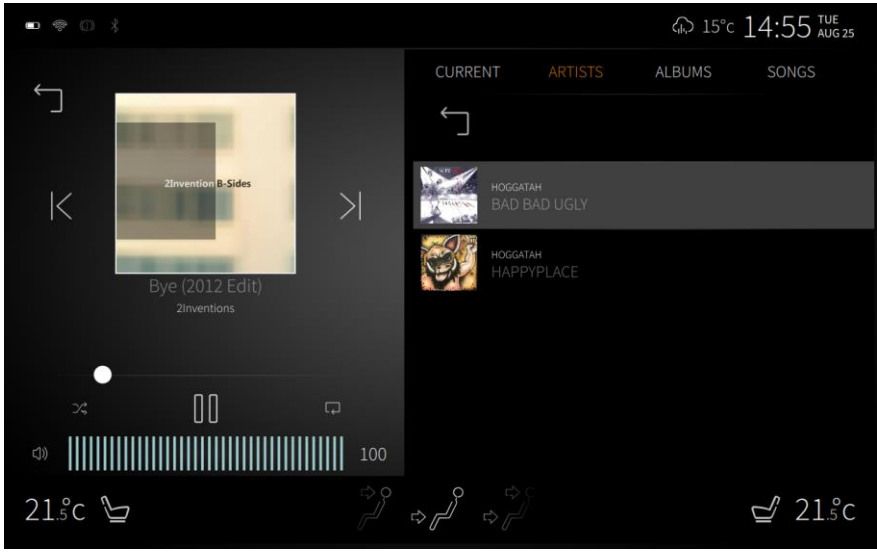
What is in the Box?

- Qt Modules
  - Qt Application Manager
  - Qt IVI
  - Qt GENIVI Extras
- Tools and best practices
  - QFace
  - QmlLive
  - GammaRay
  - QtCreator integrations of QtAppMan

More stuff

- Neptune – An Automotive Reference UI
  - Center Stack
  - Instrument Cluster
  - AppStore
- Solutions to key automotive issues
  - Notifications
  - Performance measurements
  - Application life-cycle
  - Chain of trust from app bundle to running processes


15°c
16:16
MON
AUG 24
Humboldt-
Universität
zu Berlin
Chausseestraße
Naturkundemuseum
Torstraße
Tieckstraße
Borsigpark
Nordbahnhof
246°
INVALIDENSTRAßE
0 M | ETA 00:00
ALESHA DIXON / THE ALESHA SHOW
4/15
00:00 / 00:00
JOHN DOE | 02:55
555-55 55 55
21.5c
21.5c


15°c
14:55
TUE
AUG 25
CURRENT
ARTISTS
ALBUMS
SONGS
2Invention B-Sides
Bye (2012 Edit)
2Inventions
HOGGATAH
BAD BAD UGLY
HOGGATAH
HAPPYPLACE
100
21.5c
21.5c


21:37
NAVIGATOR
15°c
P R N D
Brunnenstraße
Invalidenstraße
75
km/h
1591 km
800m
SHELL
1.5€/L
E
F
60 M
NEXT: Invalidenstraße
00:03

So, what do you have to do?

- Qt Auto provides a reference

- You still have to...
  - ... vehicle integration
  - ... OEM specific features
  - ... your own look and feel
  - ... your own app distribution infrastructure


Qt Application Manager
Qt IVI
Qt GENIVI Extras

QML 101

```
import QtQuick 2.5

Rectangle {
    width: 360
    height: 360
    Text {
        anchors.centerIn: parent
        text: "Hello World"
    }
    MouseArea {
        anchors.fill: parent
        onClicked: {
            Qt.quit();
        }
    }
}
```

- Instantiation
- Bindings
- Events

Qt App Man and the System UI

- Application Manager provides the mechanisms, System UI the behavior

- Application Manager is the QML run-time environment in which the System UI is executed.
- Control APIs:
  - **ApplicationManager**, for launching, stopping and controlling applications
  - **ApplicationInstaller**, for installing, updating and removing applications
  - **WindowManager**, for implementing a Wayland compositor
  - **NotificationManager**, for implementing org.freedesktop.Notification

Starting Apps

```
import QtQuick 2.0
import io.qt.ApplicationManager 1.0

ListView {
    id: appList
    model: ApplicationManager
    delegate: Text {
        text: name + "(" + id + ")"
        MouseArea {
            anchors.fill: parent
            onClick: ApplicationManager.startApplication(id)
        }
    }
}
```

Compositing

```
Component.onCompleted: {      // Connect to signals
    WindowManager.surfaceItemReady.connect(surfaceItemReadyHandler)
    WindowManager.surfaceItemClosing.connect(surfaceItemClosingHandler)
    WindowManager.surfaceItemLost.connect(surfaceItemLostHandler)
}

function surfaceItemReadyHandler(index, item) {     // Handle new surfaces
    filterMouseEventsForWindowContainer.enabled = true
    windowContainer.state = ""
    windowContainer.windowItem = item
    windowContainer.windowItemIndex = index
}
```

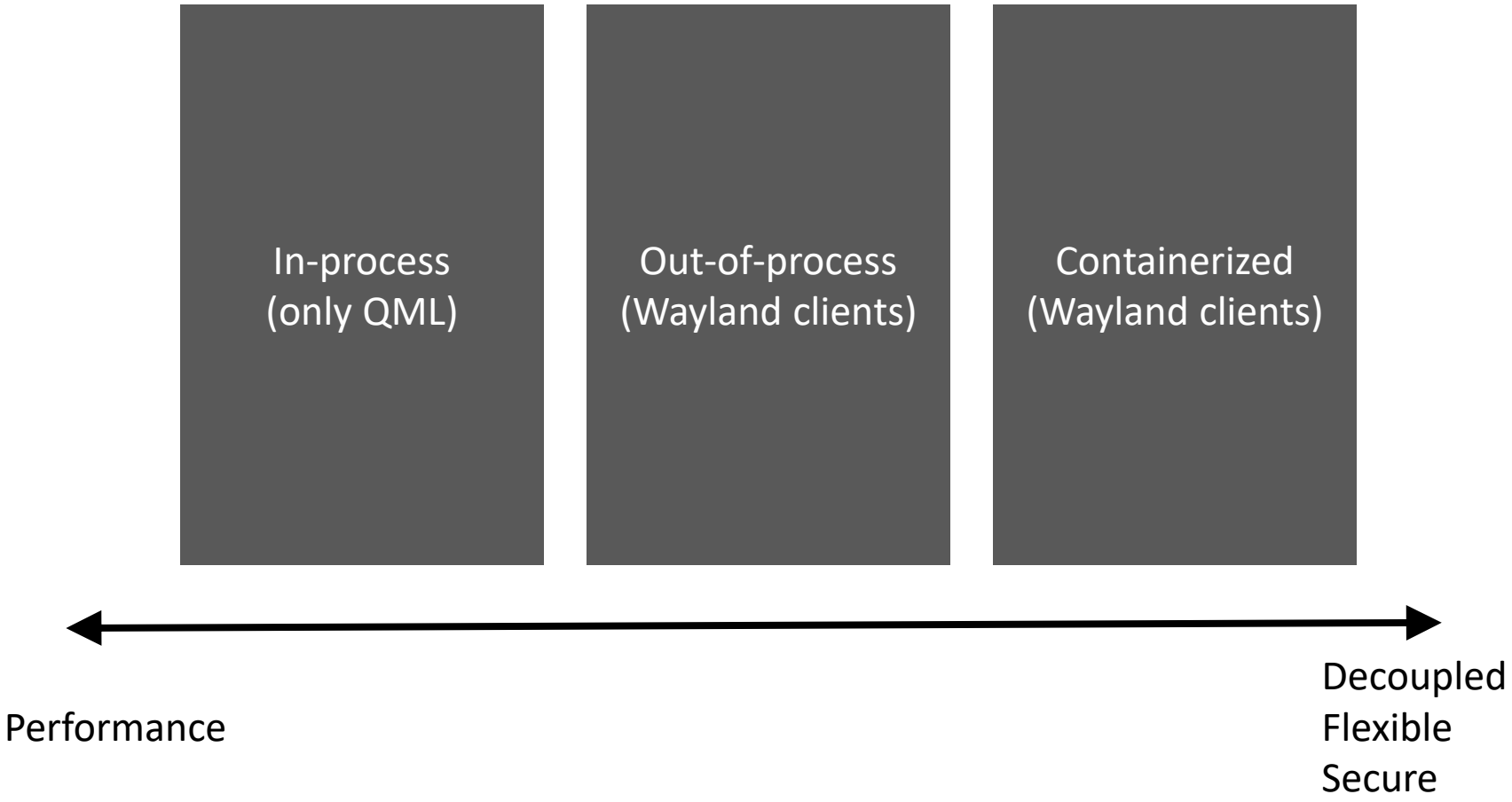
From Surface to Manifest

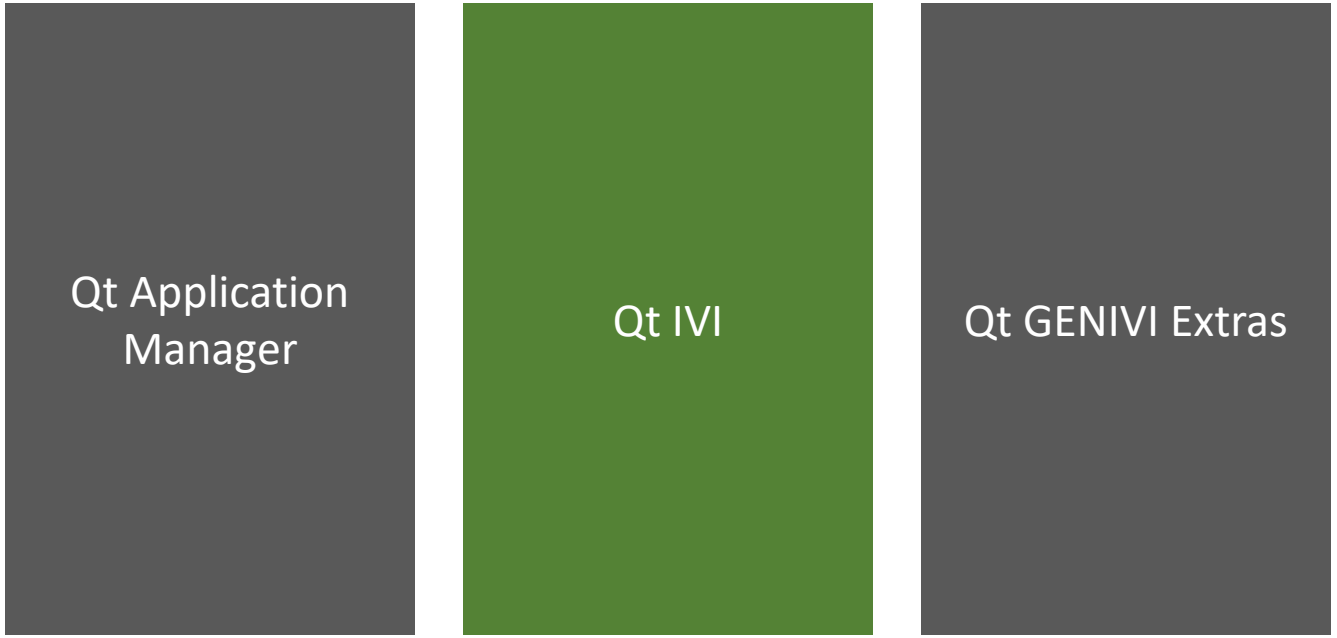
```
// Find App instance in ApplicationManager from surface
var appIdForWindow = WindowManager.get(winIndex).applicationId
var caps = ApplicationManager.capabilities(appIdFromWindow);
```

Single Process Mode

- You can execute QML applications inside the System UI
  - Systems with no or bad Wayland support
  - For performance reasons (e.g. start-up)

Choose Your Priorities




Qt Application Manager
Qt IVI
Qt GENIVI Extras

Qt IVI and QFace

- A pattern for creating a platform abstraction layer for app developers

- Reference APIs
  - VehicleFunctions
  - Media

- QFace provides an IDL and code generator for managing changing APIs

Qt IVI

- Bindable interfaces provided right away...
  - ... that become available when the backend is ready

```
import QtIvi.VehicleFunctions 1.0

ClimateControl {
    id: climateControl
    autoDiscovery: true
    onIsValidChanged: { ... }
}
```

- Dynamic loading of backends
  - Different versions based on hw
  - For simulation of desktop
  - For testing

Qface in a Nutshell

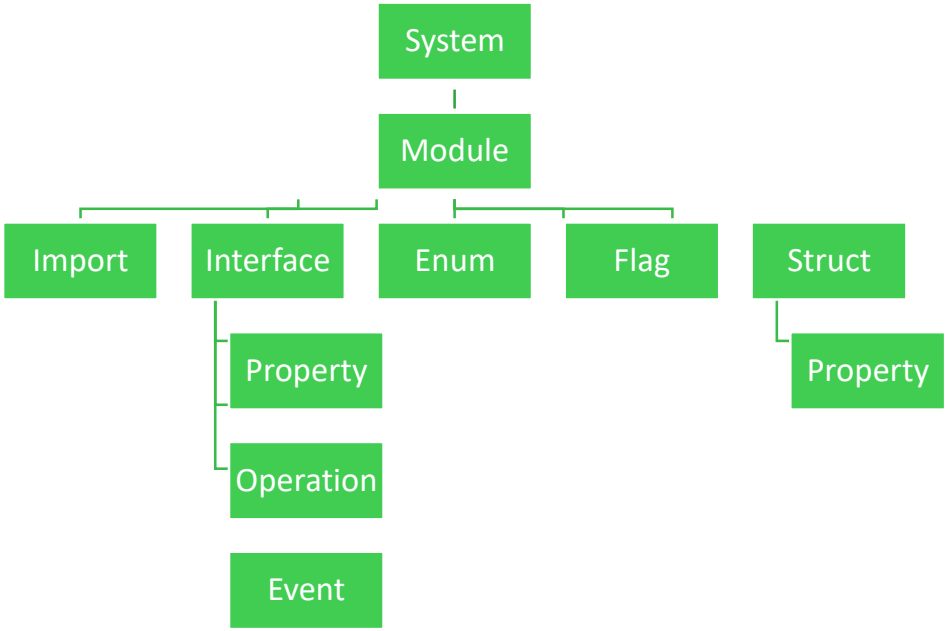
- Qt based IDL
- IDL supports
  - Interfaces
  - Data types, structs, enums, etc
  - Annotations – meta-data for the generators
- Jinja based generators
  - Used for Python web frameworks
  - Lets you traverse the model
  - Very easy to write custom generators

Qt Oriented API to Model

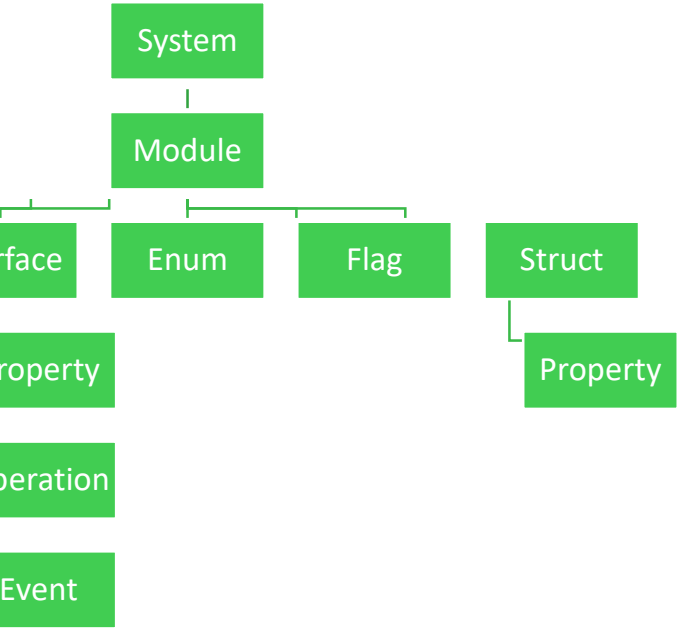
```
module org.example 1.0

interface Echo {
    string message;
    void echo(string message);
    signal broadcast(string message);
    Status status;
}

enum Status {
    Null, Loading, Ready, Error
}
```



Model to Output

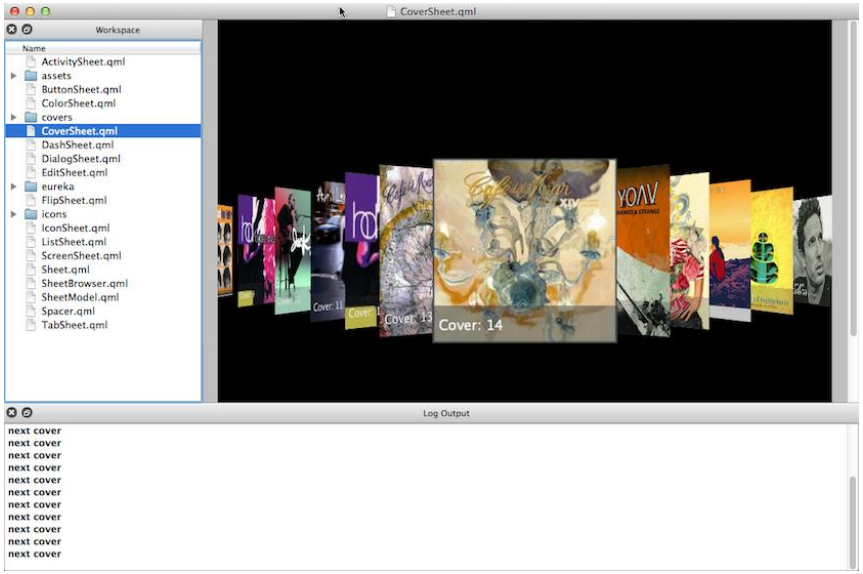



```
{% for module in system.modules %}
    {%- for interface in module.interfaces -%}
    SERVICE, {{module}}.{{interface}}
    {% endfor -%}
    {%- for struct in module.structs -%}
    STRUCT , {{module}}.{{struct}}
    {% endfor -%}
    {%- for enum in module.enums -%}
    ENUM   , {{module}}.{{enum}}
    {% endfor -%}
{% endfor %}
```


Qt Application Manager
Qt IVI
Qt GENIVI Extras

QmlLive

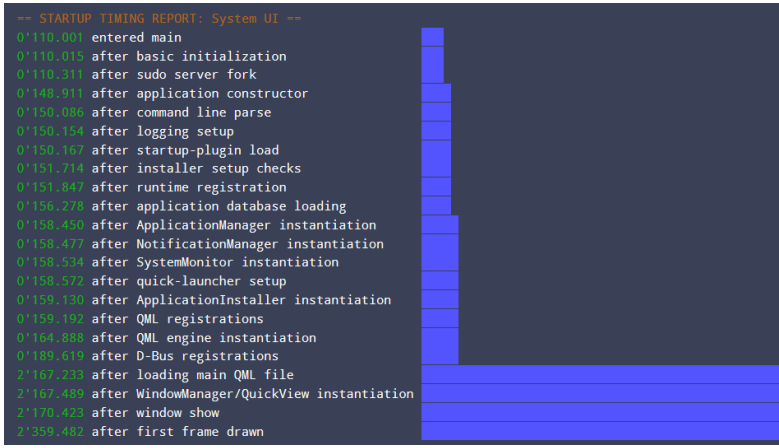
- Live reloader with server/client architecture
  - Reload live on target from developer machine



- Thanks Jolla for Contributions!

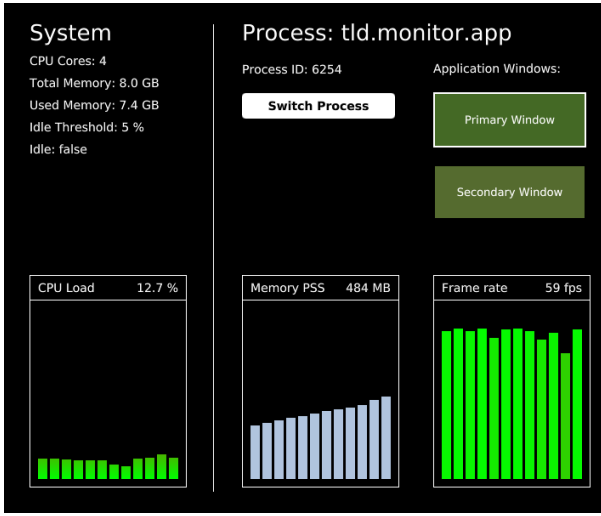
Start-up performance API

- From Qt Application Manager



- StartTimer
  - Measures times to checkpoints
  - For Apps and System UI

- ProcessMonitor and SystemMonitor
  - Framerate
  - Resource usage (mem, CPU)



Getting Involved

- Code

  http://code.qt.io/cgit/

- Docs

  https://doc.qt.io/QtAutomotiveSuite/index.html

- Yocto-based system

  http://pelux.io/

What problems do we share?

What solutions do we share?

jthelin@luxoft.com