Developing apps for Sailfish OS: introduction

Session 1. Hello world

Before we begin

There are few questions that ought to be answered before we begin:

- What is Sailfish OS?
- What is this course about?
- Who am I?

What is Sailfish OS?

To put in few words, it's an open source operating system for mobile devices, the centre of a new, open and independent ecosystem that is being created by Jolla, a company founded by ex-Nokia N9 team members.

For technical details and a more complete description, please check sailfishos.org

What is this course about?

For any ecosystem to grow and be successful, it needs apps, and whether you build your apps for fun, or profit (or both), you are contributing to the ecosystem as well.

In this course we'll go through the very basics that are required to get started and build a relatively simple application for Sailfish OS.

Who am I?

A developer, an open source enthusiast, a nerd.

I will not claim to have years of experience with the system that has just appeared... ;) This will be a learning process for me as well. Though I have been an active Qt user since it was in it's 3.x's.

Let's get started

In this session we will:

- Learn how to setup the development environment
- Learn how to build and run an application in the emulator
- Scratch some surface on the anatomy of Sailfish OS applications and build our very own "Hello world"

Getting the SDK

At the time of writing this, only Linux version of SDK is available (Mac and Windows versions will be released later).

There is no requirement for a particular flavor of Linux, so (most probably) you are good to use your favourite distribution.

Download the SDK from here: https://sailfishos.org/develop.html

Installing the SDK

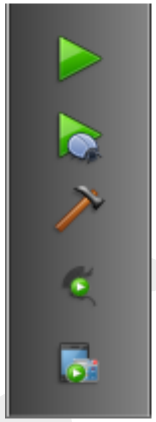
Installation is as simple as running the installer (don't forget to chmod first) and going through the wizard. If you have installed Qt SDK before, it will look exactly the same.

The only requirement is VirtualBox, which will be used to run the build machine and emulator. Get it from www.virtualbox.org (or install the package prepared by your Linux distribution).

Running the SDK

Once you launch the Sailfish IDE, you will see a standard-looking Qt Creator interface. Go ahead and create a new project. Choose "SailfishOS Qt Quick Application" template.

On the bottom of the left-side bar you will find two Sailfish IDE specific buttons: "Start Sdk" and "Start Emulator". Press both of them.



Running applications

After both virtual machines have booted, we're ready to build and run Sailfish OS applications.

The template project which we've just created already contains some nice UI. You can check it in the emulator by pressing "Run" in the IDE.

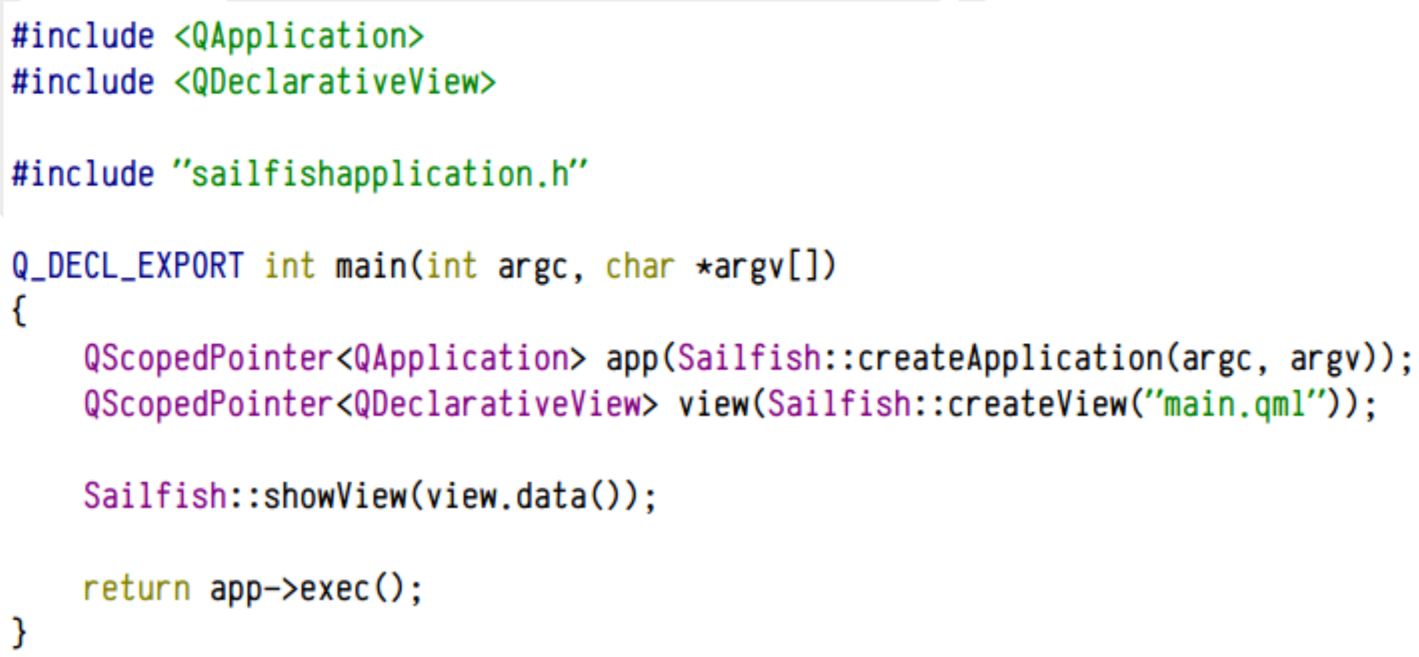
Basic application structure

Let's take a look at what's inside the template project, and focus on the "Sources" and "QML" directories for now (we'll check the others later, in more advanced topics).

Sailfish applications are based on Qt/Qt Quick, therefore we have both, C++ and QML code.

main.cpp

Like in all C/C++ applications, the entry point is the main function:

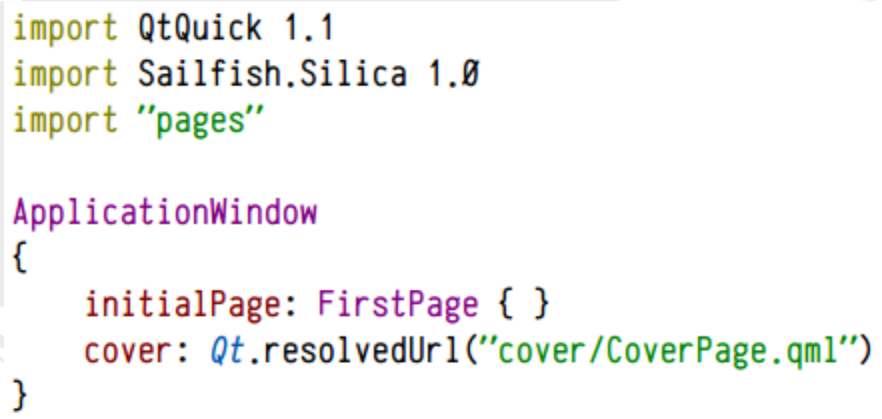


```
#include <QApplication>
#include <QDeclarativeView>

#include ”sailfishapplication.h”

Q_DECL_EXPORT int main(int argc, char *argv[])
{
    QScopedPointer<QApplication> app(Sailfish::createApplication(argc, argv));
    QScopedPointer<QDeclarativeView> view(Sailfish::createView(”main.qml”));

    Sailfish::showView(view.data());

    return app->exec();
}
```

As you can see, it is used just to set up the view for QML with the contents from main.qml, and show it. In most simple cases, main.cpp is left unchanged and all the changes go into QML files.

main.qml

As specified in main.cpp, main.qml is the module with which the application is started. Here's what it looks like:

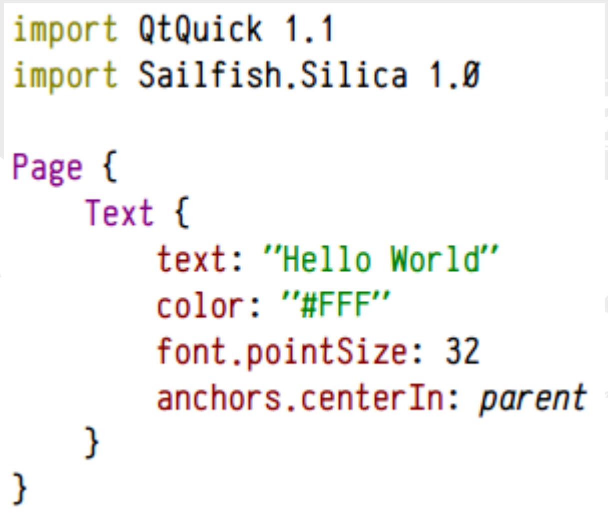


```
import QtQuick 1.1
import Sailfish.Silica 1.0
import "pages"

ApplicationWindow
{
    initialPage: FirstPage { }
    cover: Qt.resolvedUrl("cover/CoverPage.qml")
}
```

Let's go through all the statements. First, the import statements, used to load dependencies, which are: standard QtQuick, Sailfish.Silica (the Sailfish OS specific components), and our own application pages from "pages" directory. ApplicationWindow is defined with FirstPage (from "pages/FirstPage.qml") as the initial page, and cover from "cover/CoverPage.qml" (we'll talk about covers in later sessions).

Saying "Hello World"

Let's create a new file, Hello.qml, inside QML/pages directory with the following contents:



```
import QtQuick 1.1
import Sailfish.Silica 1.0

Page {
    Text {
        text: "Hello World"
        color: "#FFF"
        font.pointSize: 32
        anchors.centerIn: parent
    }
}
```

As you can see, we've defined a page with a single text component, which is centered in it's parent (the page), has a large font and says "Hello World".
In order to see this page in action, we have to go back to main.qml and change the initialPage from FirstPage to our own Hello, then press Run.

That was easy...

So far we've set up the development environment and created a "Hello World" application. In the next sessions we'll gradually be touching more advanced topics, go deeper into the details, and build more complex applications.

Thank you for your time

and till next session