basysKom

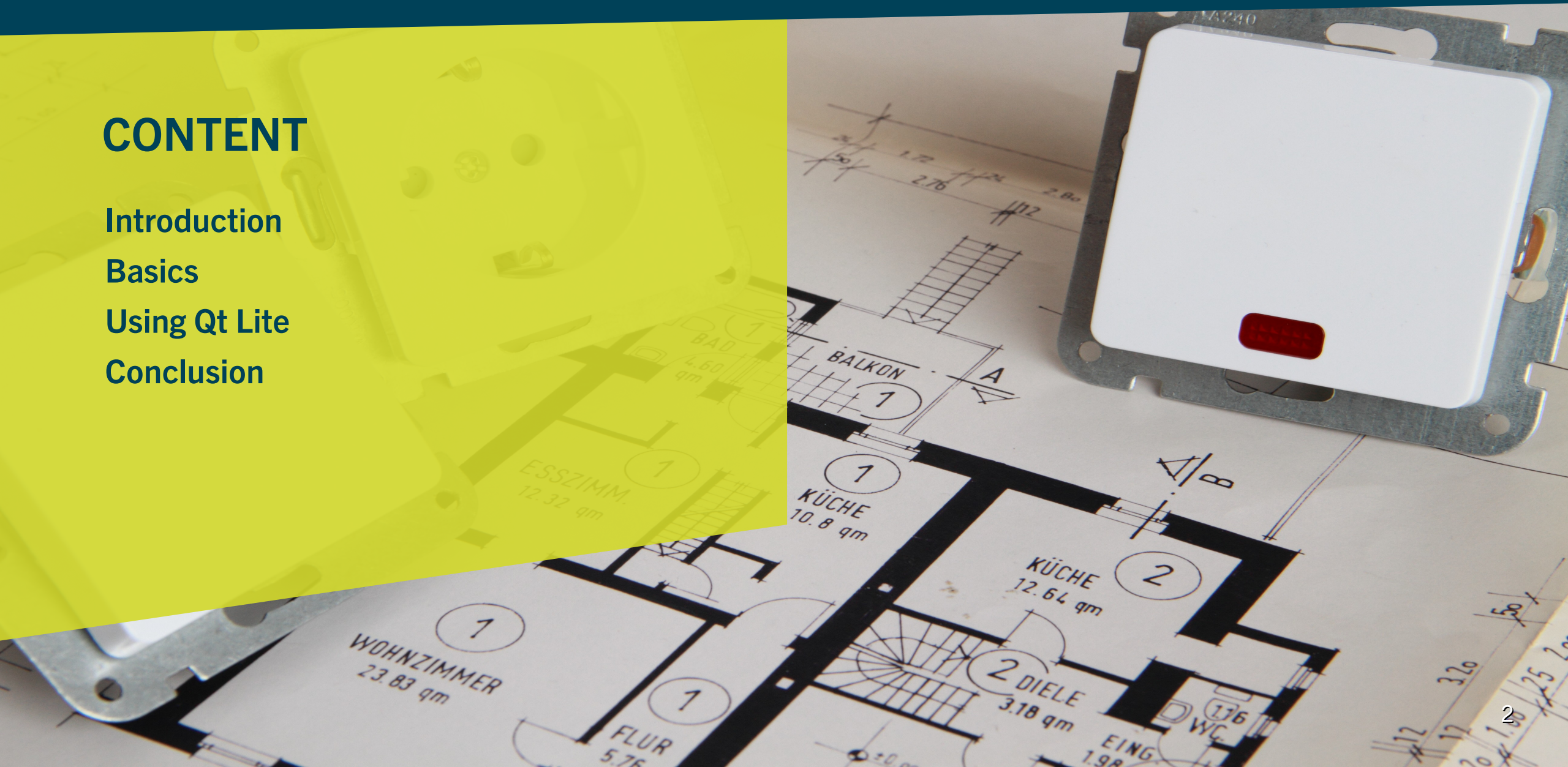
12.10.2017 Frank Meerkötter

PRACTICAL QT LITE

Qt World Summit 2017, Berlin

CONTENT

Introduction
Basics
Using Qt Lite
Conclusion



What is „Qt Lite“?

Qt Lite Blog Post from Nils

- http://blog.qt.io/blog/2016/08/18/introducing-the-qt-lite-project-qt-for-any-platform-any-thing-any-size/
- Making Qt an option for smaller devices (no GPU, less RAM, less flash)
- The big picture

What is „Practical Qt Lite“?

„Practical Qt Lite“ - a subset

- What can be done here and now
- Reduce the footprint of Qt in Embedded Linux firmwares
- Reducing the size of Qt-based apps?

What I won‘t talk about

Overall techniques to reduce the size of Embedded-Linux firmwares

- Qt might have a big foot print, but it is not the only thing making your firmware fat
- Already well documented (look for ELCE presentation/videos)

Overall techniques to reduce the size of C/C++ programs

- Also well known/documented set of techniques

Still important topics to cover when slimming down a firmware

BASICS

Basics

Before diving into Qt Lite, don‘t forget to cover the basics!

Qt5 is much less monolithic than previous versions

- Make sure to only deploy modules and plugins you actually need
- Check at runtime what is mapped into your process
- Check what is deployed (and not mapped at all)

Basics: -Os

Let the compiler do the work

- -Os in release mode is supported since Qt5.9
- ./configure -release -optimize-size
- Expect to lose *some* performance

Basics: static builds

Qt can be built as static libraries

- Allows the linker to remove unused code
- Especially interesting with -ltcg (link time optimization)
- Make sure to have the right Qt license
- Only viable when there is a single Qt application

Basics: static builds

Pitfall - Why is my application getting bigger with a static Qt?!

- Beware of qmake automatically importing plugins
- CONFIG -= import_plugins

All together now!

- ./configure -static -ltcg -optimize-size

Don‘t forget to strip your builds!

- strip -s

Basics: QRC

Qt Resource System

- Use the QRC instead of deploying QML files directly to the file system
- QRC can use compression
- The compression level can be controlled (man rcc)
- Consider additional uncompressed QRCs for resources that won‘t compress well (.png/.jpg)

QT LITE

Qt Lite (so far)

Two parts

Changes to the build system (and source)

- Built on top of the new configuration system introduced since last year
- See http://files.kde.org/akademy/2016/765_new_configuration_system_in_qt_58.mp4 for details
- Features/Dependencies
- Open Source

Configuration tool

- Qt for Device Creation

We look at both sides

Preparation

Get the Qt sources

- Installer
- git

Configure the build

See

- http://doc.qt.io/qt-5/build-sources.html

Starting point for either approach

Working with the command line only

configure is used to configure your Qt build

- No surprise :-)

Usage

- --list-features
- --feature-<the-feature-name>
- --no-feature-<the-feature-name>

Tricky

- Features depend on other features
- Why does disabling “feature-x” also disable “feature-y”, “feature-z” and “feature-q”?

https://blog.basyskom.com/2017/qt-lite/

The Qt Configuration Tool

Documentation

- https://doc.qt.io/QtForDeviceCreation/qt-configuration-tool.html

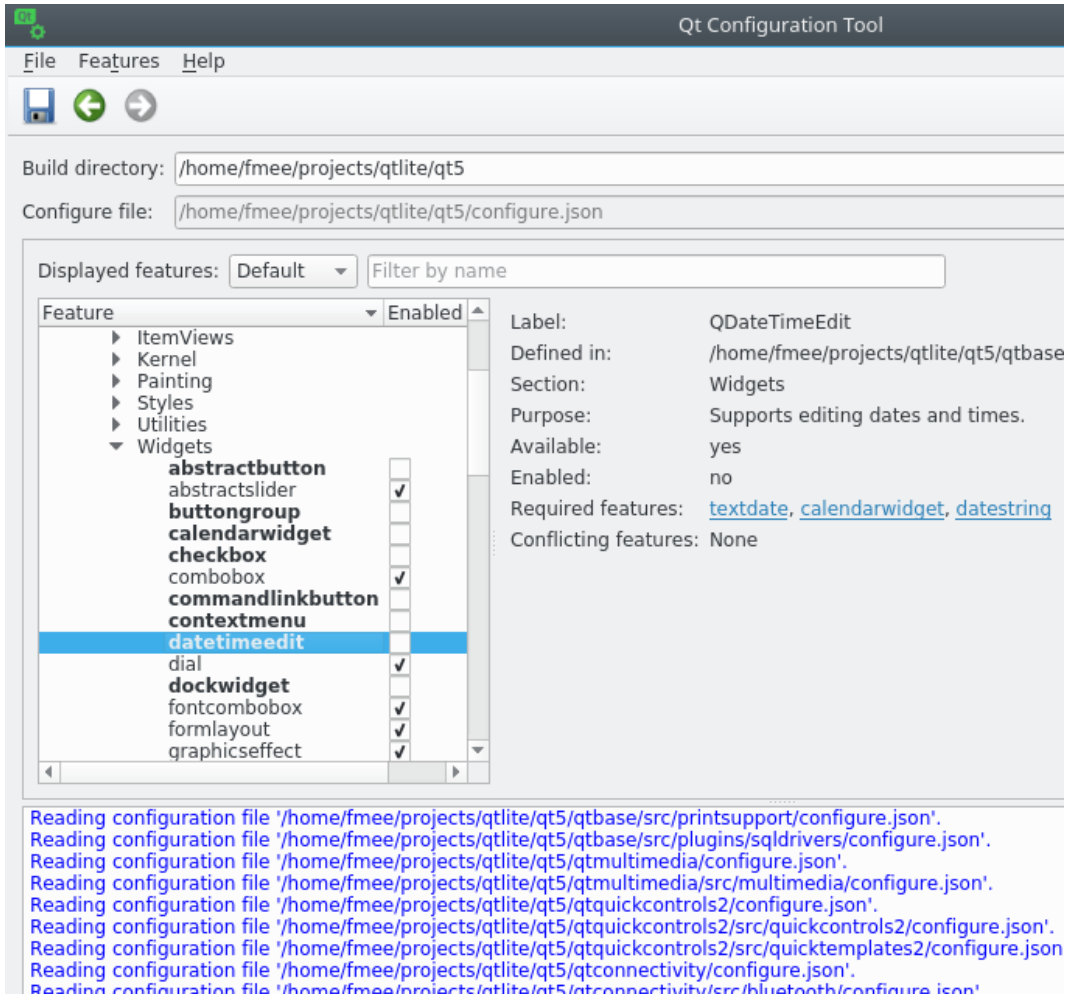
Pro

- Lists feature dependencies
- Highlights features disabled due to dependencies

Con

- No reverse dependencies
- Not much help in resolving/understanding dependency chains

No configuration presets (yet?)



Building

./configure -redo && make && make install

Caveats

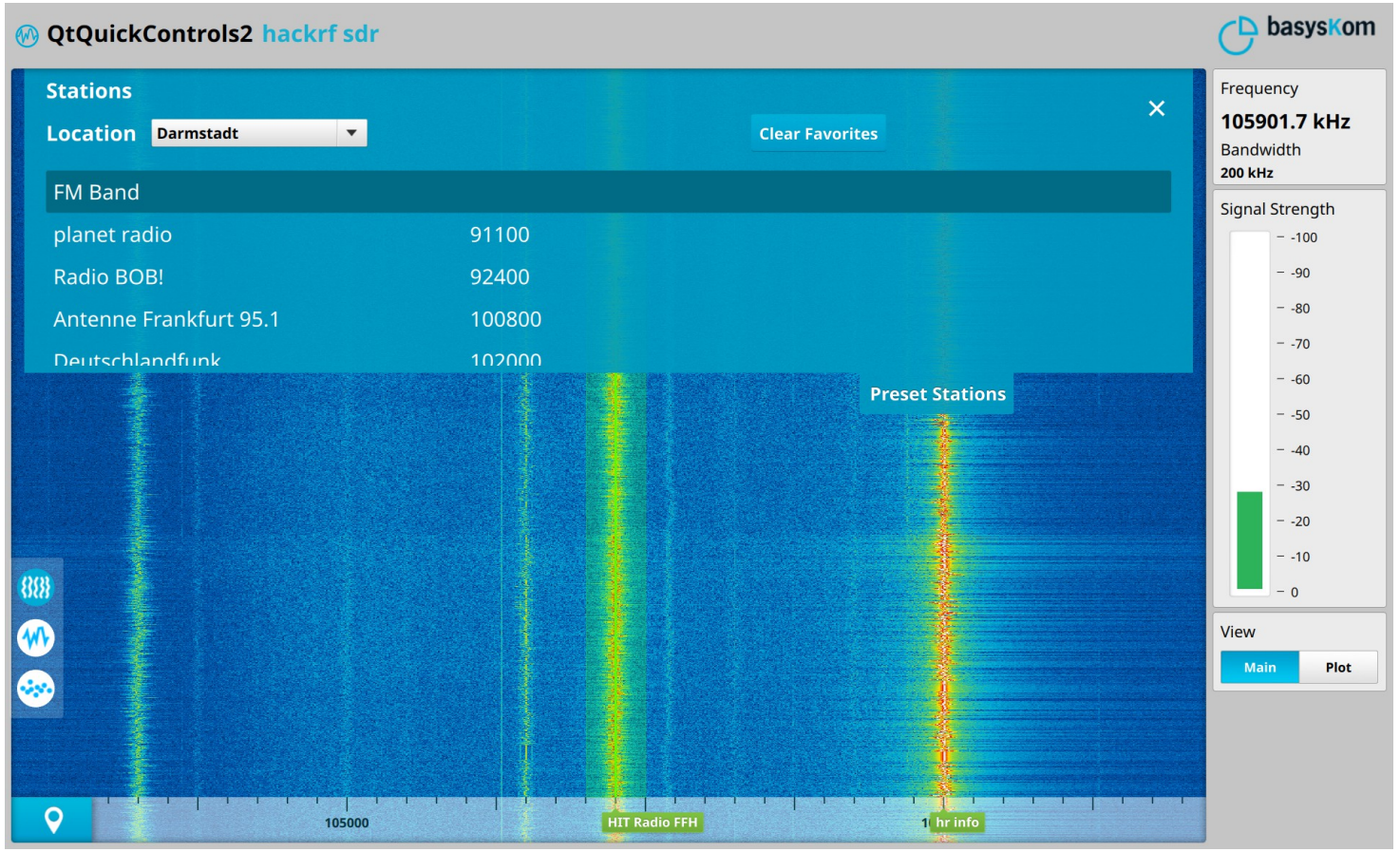
- Be prepared to resolve build issues (vastly improved with 5.9.x)
- Make sure to disable tools, examples and tests
- -nomake-tools, -no-compile-examples, -nomake-test
- Disable pre-compiled headers (-no-pch)
- Not all configurations will successfully build

CONCLUSION

Showcase

SDR showcase

Qt Quick 2, Qt Quick Controls 2



Results X86_64

Setup

- Qt5.9.2, GCC5.4
- Ubuntu 16.04

How is the size calculated?

Dynamic builds

- Size of the application binary + Qt shared objects

Static builds

- Size of the application binary

Default configuration		
-O2	26 878 kB	
-Os	21 074 kB	78%
-Os, -static	15 795 kB	58%
-Os, -static, -ltcg	12 963 kB	48%
QMLSDR minimal configuration		
-O2	21 538 kB	80%
-Os	16 939 kB	63%
-Os, -static	12 864 kB	48%
-Os, static, ltcg	10 405 kB	38%

Results ARM

Setup

- Qt5.9.2, GCC6.2, ARMv7
- Boot2Qt Image

Default configuration		
-O2	22 523 kB	
QMLSDR minimal configuration		
-Os, static, ltcg	8 846 kB	39%

How is the size calculated?

Dynamic builds

- Size of the application binary + Qt shared objects

Static builds

- Size of the application binary

Room for improvement

Provide “known to work” configuration presets

Improve the Qt Configuration Tool

- What does disabling „feature x“ will buy me?
- Consequences of disabling „feature x“?

Have CI support for Qt Lite

- At least for some presets?

Have the unit tests also cover Qt Lite

- At least for some presets?

Conclusion

A smaller Qt is possible

Make sure to make the compiler work for you

Qt Lite provides the base for smaller build configurations of Qt

- Some knowledge of Qt internals is (and most probably will always be) required

Looking forward to a smaller Qt, opening up to smaller devices

QUESTIONS?

FRANK.MEERKOETTER@BASYSKOM.COM