

BogDan Vatra <bogdan@kdab.com>
KDAB, Qt on Android



Developer
Days
2013



How many of you have an Android
device ?



Well, you are in luck !

We are going to learn how to use Qt
and Qt Creator to target your Android
device !

BogDan Vatra <bogdan@kdab.com>
KDAB, Qt on Android



Developer
Days
2013

Step by step Qt on Android tutorial

Who am I ?



- I am Bogdan Daniel Vatra (AKA BogDan).
- C/C++ developer for over 14 years.
- Qt developer for over 11 years.
- Initial author of Qt port on Android.
- The author and the current leader of Necessitas project (Qt4 port on Android).
- Active KDE (necessitas) and qt-project contributor.
- Last but not least a KDABian !

Overview



→ **Qt status.**

- Development setup for Android.
- Using Qt Creator for Android.
- Deployment options.

Qt Core

– 5.1 & 5.2

- missing system semaphores and shared memory.

– 5.3

- Shared memory is on my TODO list

Qt Multimedia

- **5.1**
 - video and audio works
 - missing camera support
- **5.2**
 - brings camera support
- **5.3**
 - ATM no other plans

Qt Network

- **5.1**
 - missing SSL support
- **5.2**
 - brings SSL support
- **5.3**
 - ATM no other plans

Qt Quick Controls

- **5.1**
 - missing android native style
- **5.2**
 - brings android native style
- **5.3**
 - ATM no other plans

Qt SQL

- **5.1**
- **5.2**
- **5.3**
 - only sqlite is provided by Qt-Project SDK

Qt WebKit & Qt WebKitWidgets

- **5.1**
- **5.2**
 - missing
- **5.3**
 - we'll see, any volunteer(s) ?

Qt Widgets

- **5.1**
 - missing android native style
- **5.2**
 - brings android native style
- **5.3**
 - ATM no other plans

Qt Essentials status



Developer
Days
2013

Qt GUI

Qt QML

Qt Quick

Qt Quick Layouts

Qt Test

- **just work on all Qt versions**

Qt Android Extras

- **5.1**
 - missing
- **5.2**
 - additional functionality for development on Android
 - QJNIEnvironment, access to the JNI Environment
 - QJNIObject, C++ wrapper around a Java class
- **5.3**
 - android services/binder support is on my TODO list

Qt Bluetooth

- **5.1**
 - missing
- **5.2**
 - missing
- **5.3**
 - on my TODO list

Qt NFC

- **5.1**
 - missing
- **5.2**
 - missing
- **5.3**
 - on my TODO list

Qt Positioning

- **5.1**
 - missing
- **5.2**
 - missing
- **5.3**
 - on my TODO list

Qt D-Bus

- **5.1**
- **5.2**
- **5.3**
 - missing, android uses the binder IPC.

Qt Sensors

- **5.1**
 - commonly used sensors
- **5.2**
 - more sensors added
- **5.3**
 - ATM no other plans

Qt PrintSupport

- **5.1**
- **5.2**
- **5.3**
 - missing, no native print support on Android

Qt OpenGL

- **5.1**
- **5.2**
 - limited to one top level widget
 - can't mix QGLWidget with other QWidget
- **5.3**
 - there is hope to use one more top level widget
 - can mix QGLWidget with other QWidget

Qt SerialPort

- **5.1**
- **5.2**
 - missing
- **5.3**
 - any volunteer(s) ?

Qt Add-Ons status



Developer
Days
2013

Qt Concurrent

Qt Declarative

Qt GraphicalEffects

Qt ImageFormats

Qt Script

Qt ScriptTools

Qt SVG

Qt XML

Qt XMLPatterns

– just work on all Qt versions

Overview



- ✓ Qt status.
- **Development setup for Android.**
 - Using Qt Creator for Android.
 - Deployment options.

Setting up the development environment for Android



Developer
Days
2013

Supported platforms:

- GNU/Linux
- Windows
- Mac

For a painless experience I do recommend GNU/Linux. For the rest of the presentation I'll refer only to GNU/Linux.

Setting up the development environment for Android



Developer
Days
2013

Install **ant** and **(open) JDK 6** (JDK 7 has a known issue when signing the package which is fixed in Qt Creator 3.0).

On debian based systems you can use the following command:
apt-get install ant openjdk-6-jdk

Setting up the development environment for Android



Developer
Days
2013

Download QtProject's SDK from
<http://qt-project.org/download>

Setting up the development environment for Android



Download Android SDK (ver. 22+) from <http://developer.android.com/sdk/index.html>

You need to download ONLY the SDK not ADT Bundle or Android Studio !

...to quickly start developing apps. It includes the essential Android SDK components and a version of the Eclipse IDE with built-in **ADT (Android Developer Tools)** to streamline your Android app development.

With a single download, the ADT Bundle includes everything you need to begin developing apps:

- Eclipse + ADT plugin
- Android SDK Tools
- Android Platform-tools
- The latest Android platform
- The latest Android system image for the emulator

Android Studio Early Access Preview

A new Android development environment called Android Studio, based on IntelliJ IDEA, is now available as an **early access preview**. For more information, see [Getting Started with Android Studio](#).

If you prefer to use an existing version of Eclipse or another IDE, you can instead take a more customized approach to installing the Android SDK. See the following instructions:

^ USE AN EXISTING IDE

If you already have an IDE you want to use for Android app development, setting up a new SDK requires that you download the SDK Tools, then select additional Android SDK packages to install (such as the Android platform and system image). If you'll be using an existing version of Eclipse, then you can add the ADT plugin to it.

Download the SDK Tools for Linux

Download the SDK
ADT Bundle for Linux

Setting up the development environment for Android



Developer
Days
2013

Download Android NDK (ver. r9+) from
<http://developer.android.com/tools/sdk/ndk/index.html>

Setting up the development environment for Android



Extract the NDK&SDK and run **android-sdk/tools/android** tool and install **SDK Tools**, **SDK Build-tools** and **Android API-10 SDK Platform**. If you are planning to build Qt yourself you'll need to install also **API-11**.

The screenshot shows the Android SDK Manager interface. The SDK Path is set to `/root/work/qt/android-sdk-linux`. The Packages tab is active, displaying a list of available packages. The 'Tools' section is expanded, showing 'Android SDK Tools' (22.0.5, Installed), 'Android SDK Platform-tools' (18.0.1, Not installed), 'Android SDK Build-tools' (18.0.1, Not installed), and 'Android SDK Build-tools' (17, Not installed). The 'Android 4.3 (API 18)' section is also expanded, showing various system images and SDKs. The 'Android 2.3.3 (API 10)' section is expanded, and the 'SDK Platform' (API 10, Rev 2) is selected and highlighted in blue. The 'Show:' section is set to 'Updates/New' and 'Installed'. The 'Sort by:' section is set to 'API level'. The 'Install 6 packages...' button is visible at the bottom right.

Name	API	Rev.	Status
Tools			
Android SDK Tools		22.0.5	Installed
Android SDK Platform-tools		18.0.1	Not installed
Android SDK Build-tools		18.0.1	Not installed
Android SDK Build-tools		17	Not installed
Android 4.3 (API 18)			
Documentation for Android SDK	18	1	Not installed
SDK Platform	18	1	Not installed
Samples for SDK	18	1	Not installed
ARM EABI v7a System Image	18	2	Not installed
Intel x86 Atom System Image	18	1	Not installed
Google APIs	18	2	Not installed
Sources for Android SDK	18	1	Not installed
Android 4.2.2 (API 17)			
Android 4.1.2 (API 16)			
Android 4.0.3 (API 15)			
Android 4.0 (API 14)			
Android 3.2 (API 13)			
Android 3.1 (API 12)			
Android 3.0 (API 11)			
Android 2.3.3 (API 10)			
SDK Platform	10	2	Not installed
Samples for SDK	10	1	Not installed
Intel x86 Atom System Image	10	2	Not installed
Google APIs	10	2	Not installed

Setting up the development environment for Android



Developer
Days
2013

Enable **USB Debugging** on your device.

On GNU/Linux you have to set the **USB permissions** for your device.

Android provides a detailed page on this matter.

Please check:

<http://developer.android.com/tools/device.html>

run

android-sdk/platform-tools/adb devices

to see if you enabled the USB debugging and set the permissions correctly.

Setting up the development environment for Android



Developer
Days
2013

- Setting up Qt Creator for Android
 - Go to **Tools->Option->Android** settings page
 - Set **Android SDK location**
 - Set **Android NDK location**
 - Make sure **Automatically creates kits for Android tool chains** is **checked**.
 - Set **Ant location**
 - Set **JDK location**
 - Click **Apply** button !
 - If you don't see the Android page, it means that the plugin is disabled and you must first enable it (**Help->About plugins**)

Setting up the development environment for Android



Filter

- Environment
- Text Editor
- FakeVim
- Help
- C++
- Qt Quick
- Build & Run
- Debugger
- Designer
- Analyzer
- Version Control
- Android**
- BlackBerry
- Devices
- Code Pasting

Android

Android Configurations

Android SDK location:

Android NDK location:

Found 6 toolchains for this NDK.

Automatically create kits for Android tool chains

Qt version for architecture mips is missing.
⚠ To add the Qt version, select Options > Build & Run > Qt Versions.

Ant location:

JDK location:

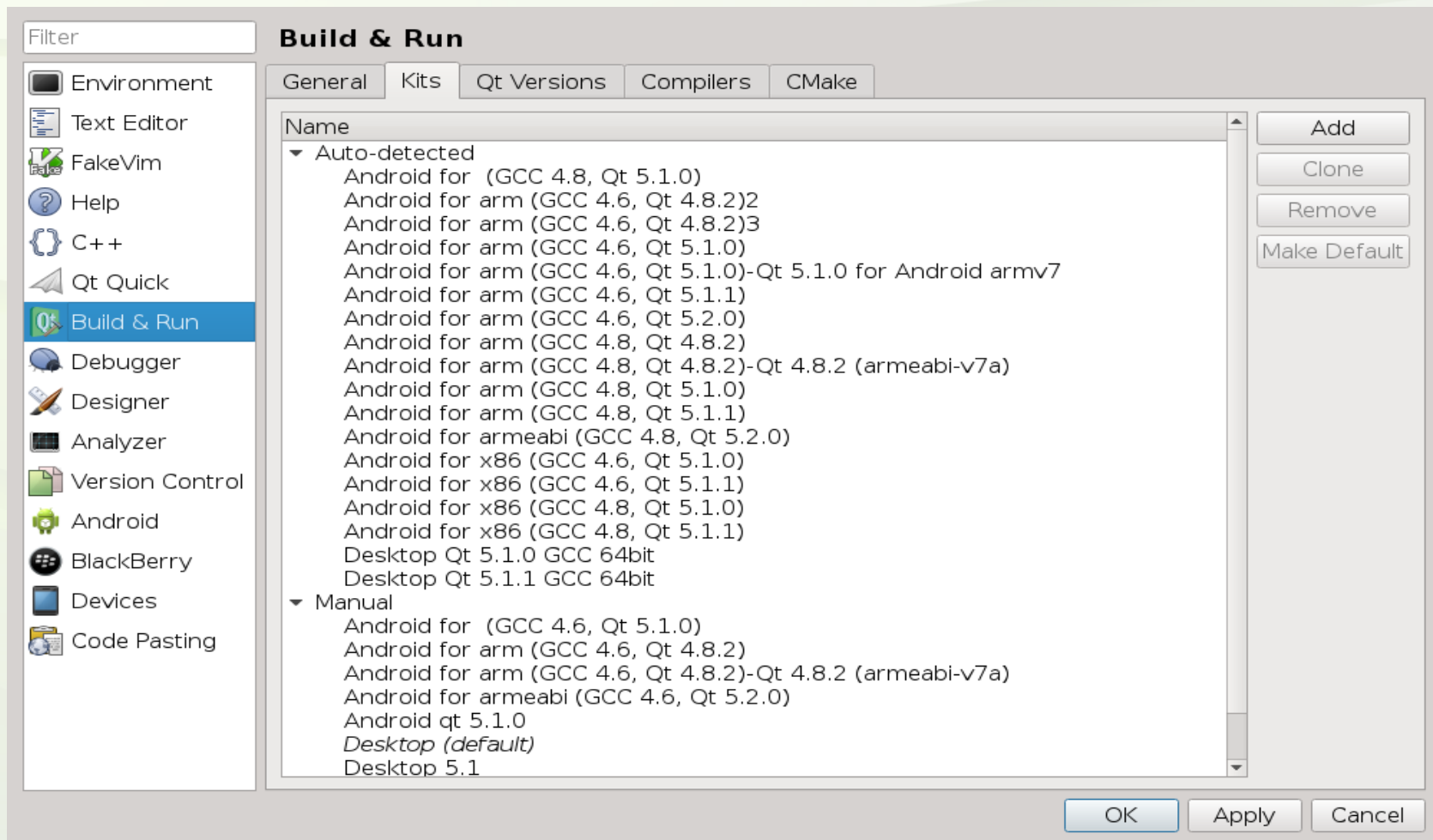
AVD Manager System/data partition size:

AVD Name	AVD Target	CPU/ABI	
android_10	API 10	armeabi	<input type="button" value="Add"/> <input type="button" value="Remove"/> <input type="button" value="Start"/>
x86	API 16	x86	
android-16	API 16	armeabi-v7a	
android-17	API 17	armeabi-v7a	
android-mips	API 17	mips	
android-18	API 18	armeabi-v7a	

Setting up the development environment for Android



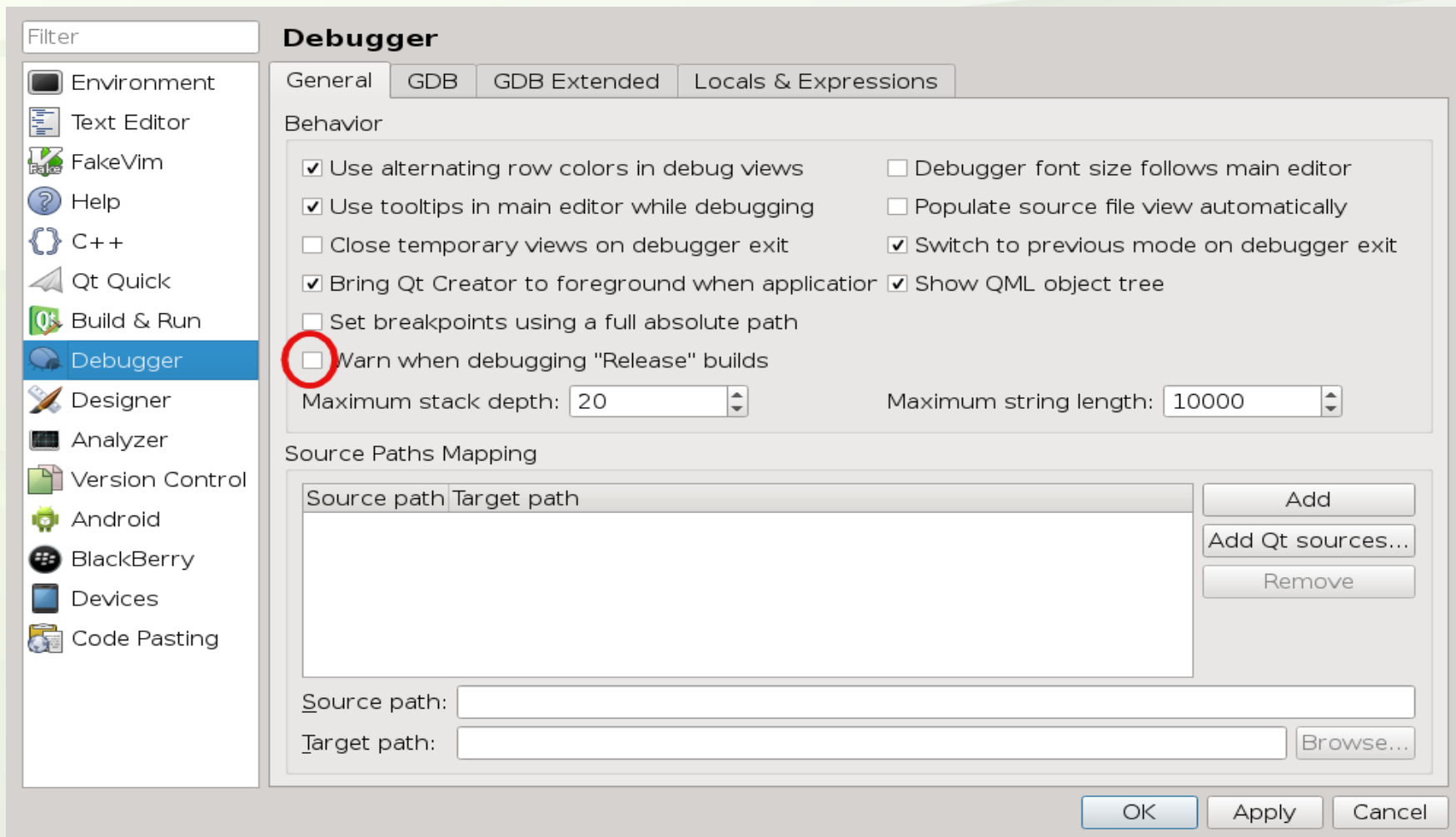
- Setting up Qt Creator for Android
 - check if Qt Creator created the Android kits



Setting up the development environment for Android



- Setting up Qt Creator for Android
 - Uncheck **Warn when debugging "Release" builds**



Overview



- ✓ Qt status.
- ✓ Development setup for Android.
- **Using Qt Creator for Android.**
- Deployment options.

Using Qt Creator for Android



This part will cover only the Qt Creator Android specific part not the whole Qt Creator.

- Open/Create a project
- Choose an Android KIT

The screenshot shows the 'Kit Selection' dialog in Qt Creator. On the left, a sidebar contains 'Location' and 'Targets' (selected). The main area is titled 'Kit Selection' and contains the text: 'Qt Creator can use the following kits for project **TestQtQML**:'.

The dialog lists several kits, each with a checkbox and a 'Details' button:

- Debug** [text field: st/build-TestQtQML-Android_for_arm_GCC_4_6_Qt_5_2_0-Debug] [Browse...]
- Release** [text field: t/build-TestQtQML-Android_for_arm_GCC_4_6_Qt_5_2_0-Release] [Browse...]
- Android for arm (GCC 4.8, Qt 5.1.1)** [Manage...] [Details ▲]
- Debug** [text field: st/build-TestQtQML-Android_for_arm_GCC_4_8_Qt_5_1_1-Debug] [Browse...]
- Release** [text field: t/build-TestQtQML-Android_for_arm_GCC_4_8_Qt_5_1_1-Release] [Browse...]
- Android for armeabi (GCC 4.6, Qt 5.2.0)** [Details ▲]
- Debug** [text field: uild-TestQtQML-Android_for_armeabi_GCC_4_6_Qt_5_2_0-Debug] [Browse...]

At the bottom right, there are three buttons: '< Back', 'Next >', and 'Cancel'.

Using Qt Creator for Android

- Select an Android KIT



Using Qt Creator for Android



Developer
Days
2013

After the previous step, Qt Creator will create and add a few files to **android** folder. Most of these files are specific to your project and you should add them to your project SCM.

- The following files are needed to build an android application
 - AndroidManifest.xml
 - version.xml
 - res/*
 - src/*
- Autogenerated file, should not be added to your project SCM.
 - build.xml
 - local.properties
 - proguard-project.txt
 - project.properties
 - assets/*
 - bin/*
 - gen/*
 - libs/*

Using Qt Creator for Android



Developer
Days
2013

Warning: **assets** and **libs** folders are not cleaned automatically, so if you are removing libs/resources, make sure you are removing these folders before you build the final android package.

Using Qt Creator for Android



- Setting up the AndroidManifest.xml
 - Package name, reversed URL (e.g. com.kdab.application)
 - Version code, this field is used by Google Play to upgrade your existing applications
 - Version name, is the version displayed in Android settings
 - Application name, is the name displayed in Android launcher
 - Run, this is the application that java part will run
 - Permissions. Here you must add all permissions that your application needs to access (e.g. internet, sd card, sensors, etc.).

Using Qt Creator for Android



- Setting up the AndroidManifest.xml

A screenshot of the Qt Creator IDE interface. The main window displays the configuration for the AndroidManifest.xml file. The left sidebar shows a project tree for 'TestQtQML' with folders for 'Sources', 'QML', and 'Other files'. Under 'Other files', there is an 'android' folder containing 'res' and 'src/org'. The 'AndroidManifest.xml' file is selected. The main area shows the configuration for the manifest, including package name, version code, version name, application name, run name, application icon, and permissions. The permissions list includes 'android.permission.INTERNET', 'android.permission.WRITE_EXTERNAL_STORAGE', and 'android.permission.ACCESS_CHECKIN_PROPERTIES'. The bottom status bar shows various toolbars and a search bar.

File Edit Build Debug Analyze Tools Window Help

Projects AndroidManifest.xml General XML Source Line: 1, Col: 1

TestQtQML

- TestQtQML.pro
- qtquick2applicationviewer
- Sources
- QML
- Other files
 - android
 - res
 - src/org

AndroidManifest.xml

version.xml

Package

Package name: org.qtproject.example.TestQtQML

Version code: 1

Version name: 1.0

Application

Application name: TestQtQML

Run: TestQtQML

Application icon:

Permissions

android.permission.INTERNET

android.permission.WRITE_EXTERNAL_STORAGE

Remove

android.permission.ACCESS_CHECKIN_PROPERTIES

Add

Open Documents

AndroidManifest.xml

main.qml

TestQtQML

Debug

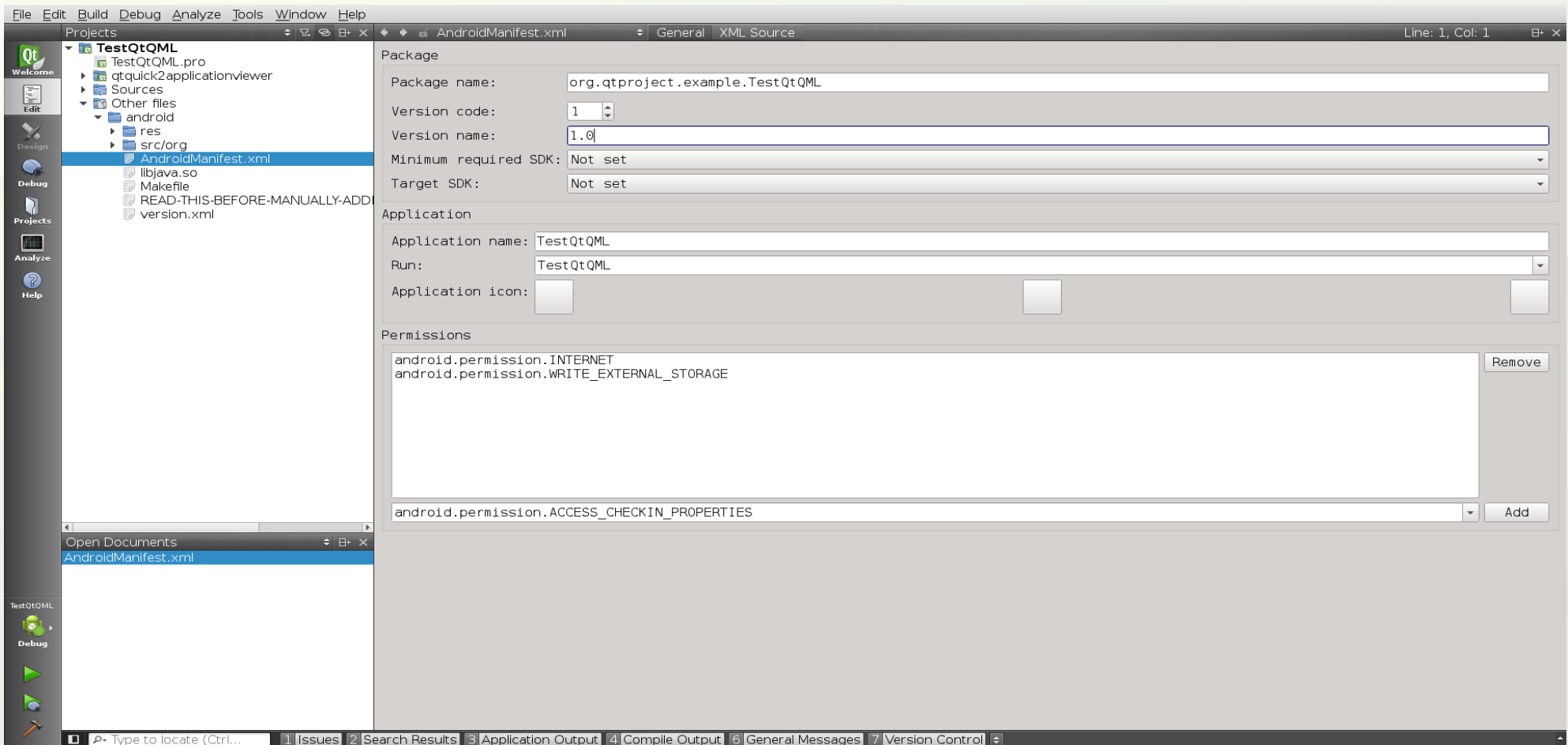
Type to locate (Ctrl...)

1 Issues 2 Search Results 3 Application Output 4 Compile Output 6 General Messages 7 Version Control

Using Qt Creator for Android



- Setting up the AndroidManifest.xml
 - Qt Creator 3.0 allows you to choose Minimum and Target SDK



Using Qt Creator for Android



- Setting up the AndroidManifest.xml
 - Android manifest is quite complicated sometime you need to edit manually

A screenshot of the Qt Creator IDE. The main window displays the 'AndroidManifest.xml' file in 'XML Source' view. The 'XML Source' tab is highlighted with a red circle. The code is an AndroidManifest.xml file for a Qt-based Android application. The file includes various metadata, permissions, and activity declarations. The left sidebar shows the project structure for 'TestQtQML', with 'AndroidManifest.xml' selected. The bottom status bar shows various toolbars and a search bar.

```
1 <?xml version='1.0' encoding='utf-8' ?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android" android:versionCode="1" package="org.qtproject.example.TestQtQML" andrc
3 <application android:name="org.qtproject.qt5.android.bindings.QtApplication" android:label="@string/app_name">
4 <activity android:name="org.qtproject.qt5.android.bindings.QtActivity" android:configChanges="orientation|locale|fontScale|keyboard|
5 <intent-filter>
6 <action android:name="android.intent.action.MAIN"/>
7 <category android:name="android.intent.category.LAUNCHER"/>
8 </intent-filter>
9 <meta-data android:name="android.app.qt_sources_resource_id" android:resource="@array/qt_sources"/>
10 <meta-data android:name="android.app.repository" android:value="@string/repository"/>
11 <meta-data android:name="android.app.qt_libs_resource_id" android:resource="@array/qt_libs"/>
12 <meta-data android:name="android.app.bundled_libs_resource_id" android:resource="@array/bundled_libs"/>
13 <meta-data android:name="android.app.lib_name" android:value="TestQtQML"/>
14 <!-- Deploy Qt libs as part of package -->
15 <meta-data android:name="android.app.bundle_local_qt_libs" android:value="1"/>
16 <meta-data android:name="android.app.bundled_in_lib_resource_id" android:resource="@array/bundled_in_lib"/>
17 <meta-data android:name="android.app.bundled_in_assets_resource_id" android:resource="@array/bundled_in_assets"/>
18 <!-- Run with local libs -->
19 <meta-data android:name="android.app.use_local_qt_libs" android:value="1"/>
20 <meta-data android:name="android.app.libs_prefix" android:value="/data/local/tmp/qt/">
21 <meta-data android:name="android.app.load_local_libs" android:value="plugins/platforms/android/libqtforandroid.so:libs/libgnustl
22 <meta-data android:name="android.app.load_local_jars" android:value="jar/QtAndroid-bundled.jar:">
23 <meta-data android:name="android.app.static_init_classes" android:value=":">
24 <!-- Messages maps -->
25 <meta-data android:name="android.app.ministro_not_found_msg" android:value="@string/ministro_not_found_msg"/>
26 <meta-data android:name="android.app.ministro_needed_msg" android:value="@string/ministro_needed_msg"/>
27 <meta-data android:name="android.app.fatal_error_msg" android:value="@string/fatal_error_msg"/>
28 <!-- Messages maps -->
29 <!-- Splash screen -->
30 <meta-data android:name="android.app.splash_screen" android:resource="@layout/splash"/>
31 <!-- Splash screen -->
32 </activity>
33 </application>
34 <supports-screens android:normalScreens="true" android:smallScreens="true" android:largeScreens="true" android:anyDensity="true"/>
35 <uses-permission android:name="android.permission.INTERNET"/>
36 <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
37 </manifest>
38
```

Using Qt Creator for Android



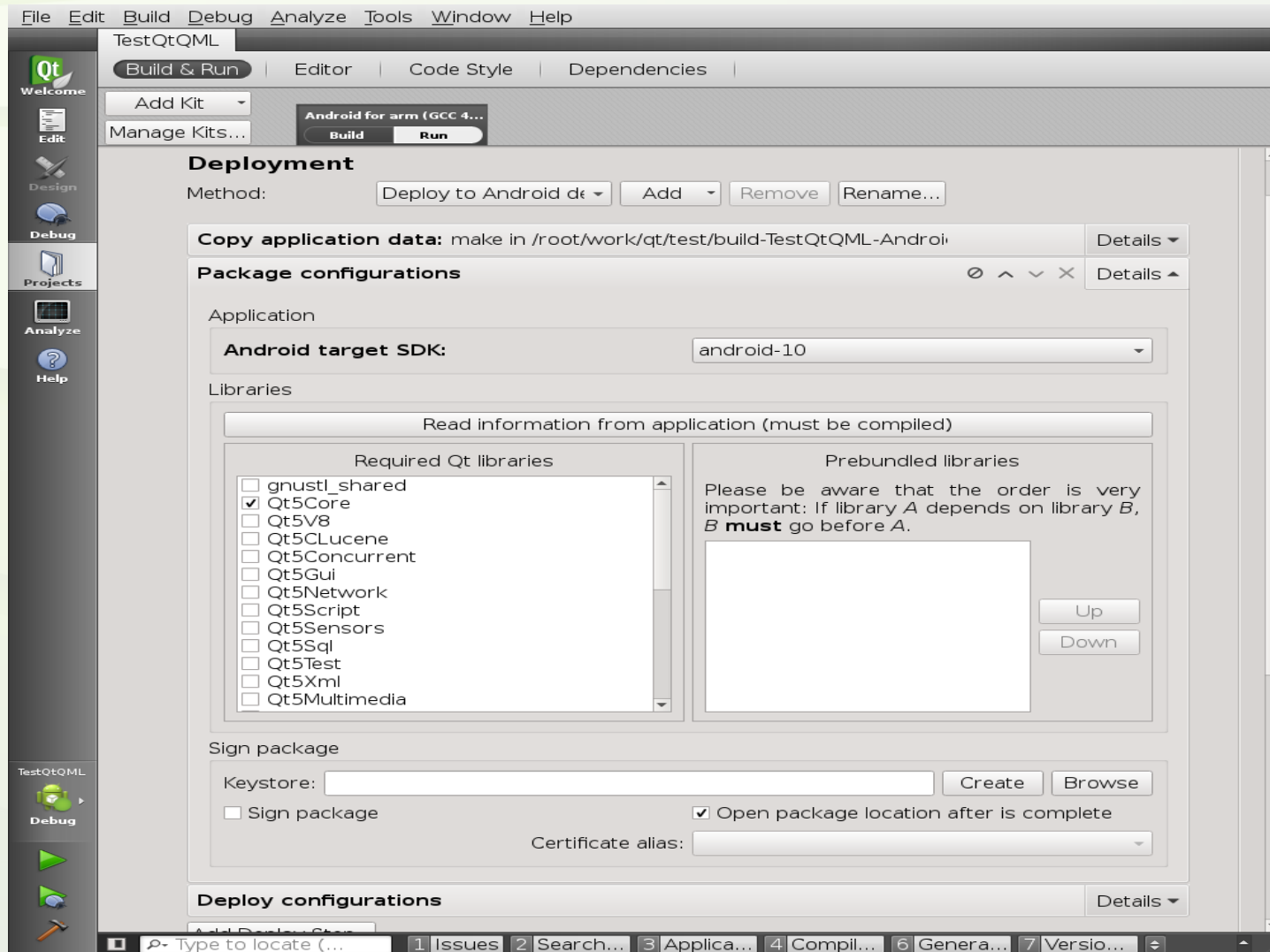
Developer
Days
2013

- Setting up the AndroidManifest.xml
 - For more informations about Android Manifest, please check
<http://developer.android.com/guide/topics/manifest/manifest-intro.html>

Using Qt Creator for Android



- Package configuration



Using Qt Creator for Android



Developer
Days
2013

- Signing the application.
 - create a certificate

Keystore

Password: [masked]

Retype password: [masked]

Show password Password is ok

Certificate

Alias name: play

Keysize: 2048

Validity (days): 10000

Password: [masked]

Retype password: [masked]

Show password Password is ok

Certificate Distinguished Names

First and last name: BogDan Vatra

Organizational unit (e.g. Necessitas):

Organization (e.g. KDE): KDAB

City or locality: Brasov

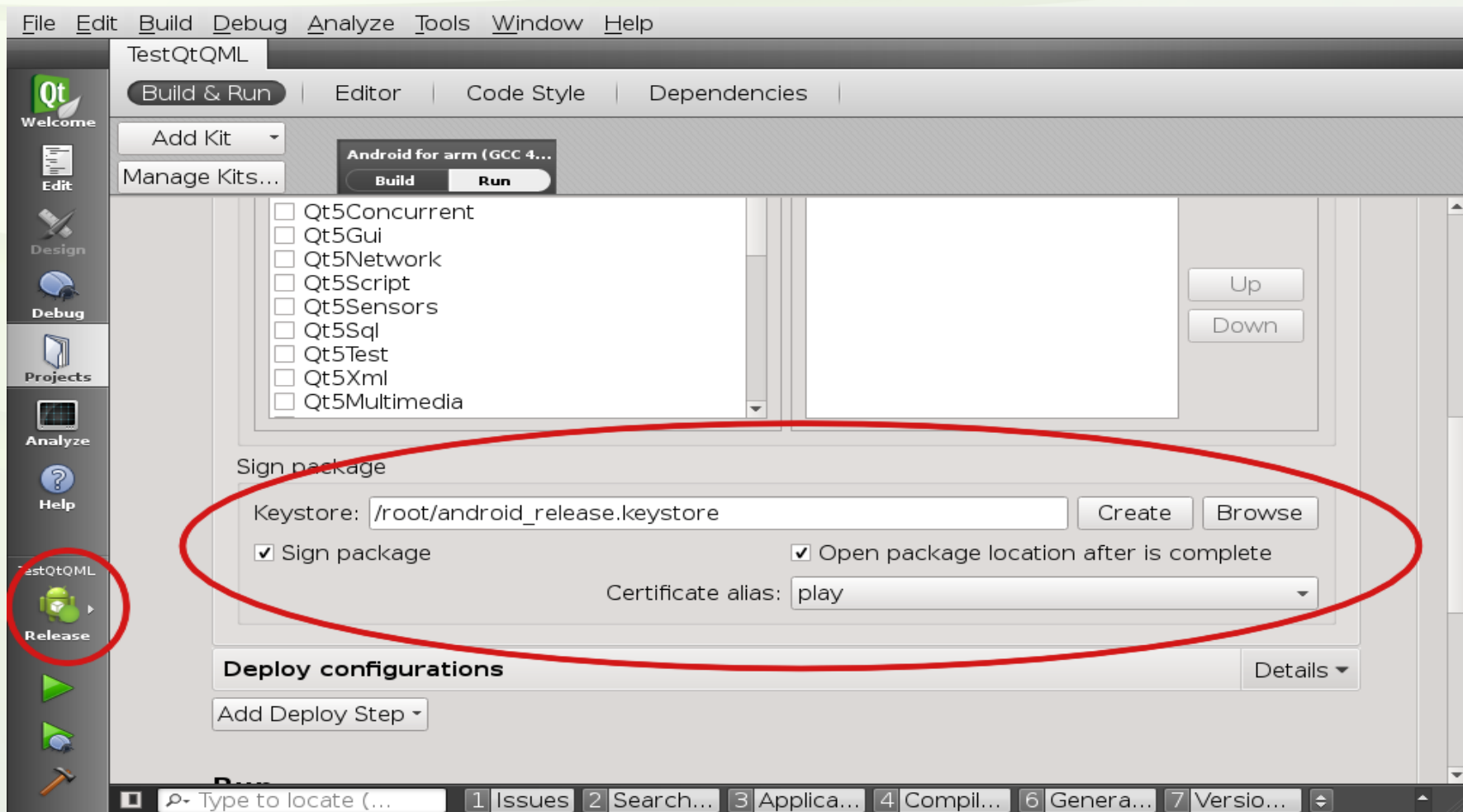
State or province: Brasov

Two-letter country code for this unit (e.g. RO): RO

Save Close

Using Qt Creator for Android

- Signing the application.
 - switch to release mode
 - open keystore and check “sign package”



Using Qt Creator for Android



Developer
Days
2013

- Signing the application
 - If you want to target more than one platform with the same package, then you must build and run in release mode the application for every platform and then sign it !
 - When Qt Creator opens the location of the signed package, there you will have a few .apk files. Only the one with "**-signed**" in the name is the one which is signed and ready for publishing.

Overview



- ✓ Qt status.
- ✓ Development setup for Android.
- ✓ Using Qt Creator for Android.
- **Deployment options.**

Choosing the right deploying system

Qt Creator supports three deploying systems.

- Use Ministro service to install Qt.
- Deploy local Qt libraries to temporary directory.
- Bundle Qt libraries into the APK.

The screenshot shows the Qt Creator IDE interface. The main window displays the 'Deploy configurations' dialog for the project 'TestQtQML'. The dialog is divided into two sections: 'Qt Deployment' and 'Advanced Actions'. In the 'Qt Deployment' section, three radio buttons are visible: 'Use Ministro service to install Qt' (which is selected), 'Deploy local Qt libraries to temporary direc', and 'Bundle Qt libraries in APK'. In the 'Advanced Actions' section, there are two buttons: 'Clean Temporary Libraries Directory on Device' and 'Install Ministro from APK'. The IDE's sidebar on the left shows the 'Release' tab selected, and the bottom status bar displays various toolbars and a search field.

Choosing the right deploying system



- **Deploy local Qt libraries to temporary directory.**
 - This deploy system is used mostly by Qt hackers when hacking on Qt itself.

Choosing the right deploying system



- **Bundle Qt libraries into the APK.**

- This feature was added recently to Qt Creator. Beside your application and your resources Qt Creator adds all Qt libraries your application needs to run.
- Pro
 - The APK contains everything it needs to run.
- Con
 - The APK is HUGE due to Qt libs which are pretty big (+40Mb/platform).
 - All Qt libs must be unpacked! So your application will need a lot of free space to run (+50Mb)
 - Most of the low-end devices users can't afford to spend that much free space.
 - Due to big size you can't target more than one platform/apk. You must create an apk for every platform (armv5, armv7, x86).
 - No VFP on armv5 devices or NEON on armv7 devices.
 - Qt not shared with other Qt apps.
 - No separate libs update.

Choosing the right deploying system



- **Use Ministro service to install Qt.**
 - Why Ministro was invented?
 - In 2009/2010 most devices have limited free space (<100 Mb).
 - Google Market had a lower package size limit than today's 50Mb limit/apk.

Choosing the right deploying system



- **Use Ministro service to install Qt.**
 - How it works
 - your package will contain **ONLY** your application's .so file(s), its resources.
 - application starts and connect to Ministro service
 - opens Android play for the user to install Ministro
 - sends to Ministro the dependencies list
 - downloads missing files
 - Ministro sends back another list with everything the application needs to load.
 - The Application loads everything and starts the Qt application.

Choosing the right deploying system



- **Use Ministro service to install Qt.**

- Pro

- Using Ministro, the user needs to download *ONLY* once the Qt libs.
 - Ministro can detect VFP on armv5 and NEON on armv7 download respective libs.
 - Ministro can update Qt libs, without requiring app update.
 - You can easily target all Android platforms with a single APK.
 - You can use your own Ministro sources with your own libraries.

- Con

- Not very user friendly?
 - Ministro upgrades Qt libraries and it might break things?

Choosing the right deploying system



- **Use Ministro service to install Qt.**
 - Ministro uses a Debian like release scheme with three different repositories
 - **unstable**
 - **testing**
 - **stable**

Choosing the right deploying system



Developer
Days
2013

- **Use Ministro service to install Qt.**

- Every major Qt release will use a different location for Ministro.

- <http://download.qt-project.org/ministro/android/qt5/qt-5.1/>
 - <http://download.qt-project.org/ministro/android/qt5/qt-5.2/>

That's all folks!



Thank you for your time !

Any questions?