**Sergey**

Q1: How the YambaClient library is done?

A1: The source code of the YambaClient is open sourced on <https://github.com/twitter-university/YambaClientLib> and the „date bug“ is located in <https://github.com/twitter-university/YambaClientLib/blob/master/src/com/marakana/android/yamba/clientlib/YambaClient.java> at the the line 356:

**else** **if** **(**endsWithTags**(**stack**,** "status"**,** "created\_at"**))** **{**

createdAt **=** dateFormat**.**parse**(**text**);**

**}**

The library is using the *DefaultHttpClient* for http connection and is parsing the XML using the „*ugly*“ XmlPullParser.

Q2: How to debug problems on runtime?

A2: Demo live on **WeatherApp** from Marius for DDMS (Dalvik Debug Monitors Server) debugging.

More info about low level debugging from Internals course:

* Vineri mai mult insight despre kernel debugging (Dmesg), native debugging (GDB) si platform/app debugging ( via DDMS si JDWP – Java Debug Wire Protocol )
* Step by step debugging NDK - <http://mhandroid.wordpress.com/2011/01/23/using-eclipse-for-android-cc-debugging/>
* Check Larry article - <http://www.eweek.com/c/a/Linux-and-Open-Source/How-to-Set-Up-Android-Platform-Development-and-Debugging/>
* Check alternatives: tombstone, ndk-stack si addr2line - <http://bytesthink.com/blog/?p=133>

**Konstantin**

Q1: Watermark/filter on a camera preview.

A1: Check the sample in the platform:

<https://github.com/android/platform_frameworks_base/tree/master/media/mca/samples/CameraEffectsRecordingSample>

Check also the demo from Marius on the QR code scanner in Valrom sample based on Zxing open source project: <https://code.google.com/p/zxing/source/browse/trunk#trunk/android/src/com/google/zxing/client/android>

Watermark on video, even if it is not asked can be done using a proxy ffmpeg server:

<http://superuser.com/questions/678168/add-an-image-as-watermark-to-a-video-file-with-ffmpeg>

<http://ffmpeg.org/ffmpeg-filters.html#overlay>

**Alexey** B.

Q1: Expose GPU Android( Renderscript , ... )

A1: Filterscript

Filterscript defines constraints on the existing Renderscript APIs that allow the resulting code to run on a wider variety of processors (CPUs, GPUs, and DSPs). To create Filterscript files, create .fs files instead of .rs files, and specify #pragma rs\_fp\_relaxed to tell the Renderscript runtime your scripts do not require strict IEEE 754-2008 floating point precision. This precision allows flush-to-zero for denorms and round-towards-zero. In addition, your Filterscript scripts must not use 32-bit built-in types and must specify a custom root function by using the \_\_attribute\_\_((kernel)) attribute because Filterscript does not support pointers, which the default signature of the root() function defines.

More info on <http://developer.android.com/guide/topics/renderscript/compute.html>

**Artem**

Q1: Decompile some APK-s

A1: See my presentation on <http://www.slideshare.net/fastlink2/droidcon-eastern-europe-2013-how-secure-is-an-androidapp> and live demo with the VK app ☺.

Q2: How to discover hidden api-s

A2: Android has two types of APIs that are not accessible via SDK.

* package com.android.internal - internal API
* collection of classes and functions marked with @hide javadoc attribute

Steps to enable them, check on <http://devmaze.wordpress.com/2011/01/18/using-com-android-internal-part-1-introduction/> Compiling the platform by removing the @hide will help you get visibility.

**Andrey**

Q1: Simple but hardware accelerated video player

A1: Check the open source products: like <https://code.google.com/p/dolphin-player/>, <https://github.com/havlenapetr/FFMpeg> and closed source like Nexplayer - <http://www.nextreaming.com/>

**Elena Guschina**

Q1: GPS and location tracking

A1: Based on the sample located on <http://developer.android.com/training/location/receive-location-updates.html> and <http://developer.android.com/training/location/index.html> I will demo the /Samples/LocationUpdates

**Ilya**

Q1: Unit test, ui test alternatives. How to prepare the apps for testing

A1: Marius will demo the Testing.html and will show alternative solutions.

Official UI testing <http://developer.android.com/tools/testing/testing_ui.html> with alternatives Robotium - <https://code.google.com/p/robotium/> and Testdroid/Testrecorder - <http://bitbar.com/> . Keep an eye on Espresso kit - <https://code.google.com/p/android-test-kit/> , official probably tool in the testing toolkit.

Introduction to monkeyrunner <http://developer.android.com/tools/help/monkeyrunner_concepts.html>

Q3: Maybe a flashlight with dimmed light

A3: Learn from the best guys: <https://play.google.com/store/apps/details?id=com.nadstech.bestflashlight>

**Dmitry Zinchenko**

Q1: Shortcut launcher for other apps

A1: Use the PackageManager to launch the existing apps

*PackageManager pm = getPackageManager();*

*Intent intent = pm.getLaunchIntentForPackage("com.example.package");*

*startActivity(intent);*

You can iterate the packages using the code bellow:

*final PackageManager pm = getPackageManager();*

*List<ApplicationInfo> packages = pm.getInstalledApplications(PackageManager.GET\_META\_DATA);*

*for (ApplicationInfo packageInfo : packages) {*

*Log.d(TAG, "Installed package :" + packageInfo.packageName);*

*Log.d(TAG, "Launch Activity :" + pm.getLaunchIntentForPackage(packageInfo.packageName));*

*}*

More info on: <http://qtcstation.com/2011/02/how-to-launch-another-app-from-your-app/>

**Sergey Perepelkin**

Q1: Open CV library in Android

A1: Check the existing official tutorials on <http://docs.opencv.org/doc/tutorials/introduction/android_binary_package/android_dev_intro.html> and <http://opencv.org/platforms/android.html>

**Valentin Kubarev**

Q1: Simple image editor like Windows Paint

A1: Check the example I created on /Samples/00\_Paint\_2D.

**Julia Shashkina**

Q1: Simple video transcoder

A1: FFmpeg usages in Android via open source projects:

<http://bambuser.com/opensource>

**Egor, Pavel, Nina, Alexey**

Q1: JNI/NDK - cross example native/java

A1: Check the /sample/FibonacciNative . We will do an extensive demo on this code.

Marius is using the Internals samoke:

* Page 53-95 NDK and JNI
* Sample with *FibonacciNative* at page 76
* Check the samples from NDK folder /android/ndk/samples
* The New Circle has a cool sample online at: <https://thenewcircle.com/s/post/49/using_ndk_to_call_c_code_from_android_apps>
* The code is available as ZIP archive: <https://github.com/marakana/FibonacciNative/zipball/master>
* By Git: git clone <https://github.com/marakana/FibonacciNative.git>

**Anton**

Q1: Video processing

A1: OpenCV for Android or BoofCV

OpenCV is a computer vision library that enables you to process video in real-time. Check the existing official tutorials on <http://docs.opencv.org/doc/tutorials/introduction/android_binary_package/android_dev_intro.html> and <http://opencv.org/platforms/android.html>

BoofCV is an open source Java library for real-time computer vision and robotics applications. Written from scratch for ease of use and high performance, it often outperforms even native libraries. Functionality includes optimized low-level image processing routines, feature tracking, and geometric computer vision. Check the sample tutorial for Android on : <http://boofcv.org/index.php?title=Example_Android_Video>

**Alexey Varyzgin**

Q1: What a customer wants from Android and from an android app

A1: Free discussion with Marius on customer journey and platform direction.