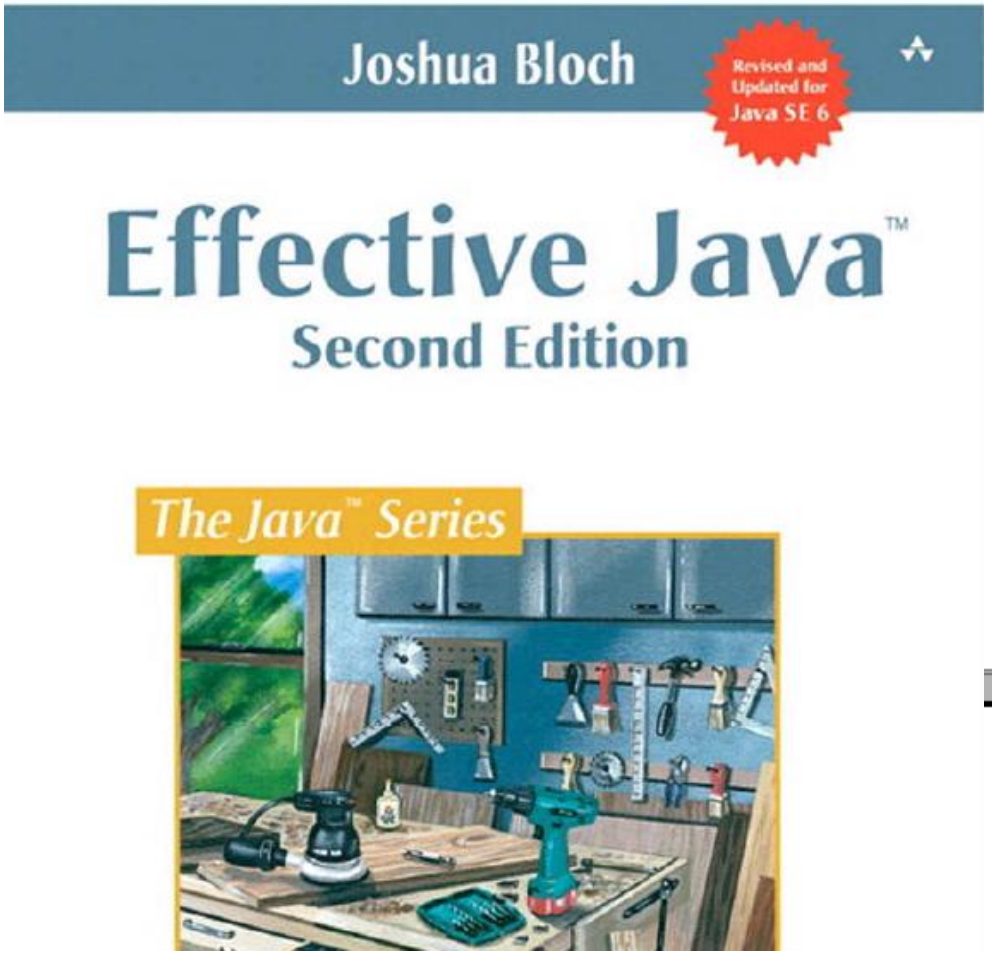
Алгоритмы и структуры данных

Android: Введение

Кузнецов
Андрей Николаевич

Санкт-Петербургский Государственный
Политехнический Университет



Foreword

IF a colleague were to say to you, "Spouse of me this night today manufactures the unusual meal in a home. You will join?" three things would likely cross your mind: third, that you had been invited to dinner; second, that English was not your colleague's first language; and first, a good deal of puzzlement.

Get Started

Everything you need to start developing apps for Android is available here on *developer.android.com*. You'll find everything from the API documentation and design guidelines, to information about the current device landscape and how you can distribute and monetize your app.

No two apps are built in the same way, but we've structured the information you need to build an app into the following three sections that represent the general order for app development.

1. Design

Before you write a single line of code, you need to design the user interface and make it fit the Android user experience. Although you may know what a user will *do* with your app, you should pause to focus on how a user will *interact* with it. Your design should be sleek, simple, powerful, and tailored to the Android experience.

So whether you're creating an app alone or you're part of a team, study the Design guidelines first.

2. Develop

Once your design is finalized, all you need are the tools to turn your app ideas into reality. Android's framework provides you the APIs to build apps that take full advantage of device hardware, connected accessory devices, the Internet, software features, and more. With the power of Android, there's no limit to the power of your apps.

Everything you need to learn about the app framework and developer tools is in the Develop documentation.

3. Distribute

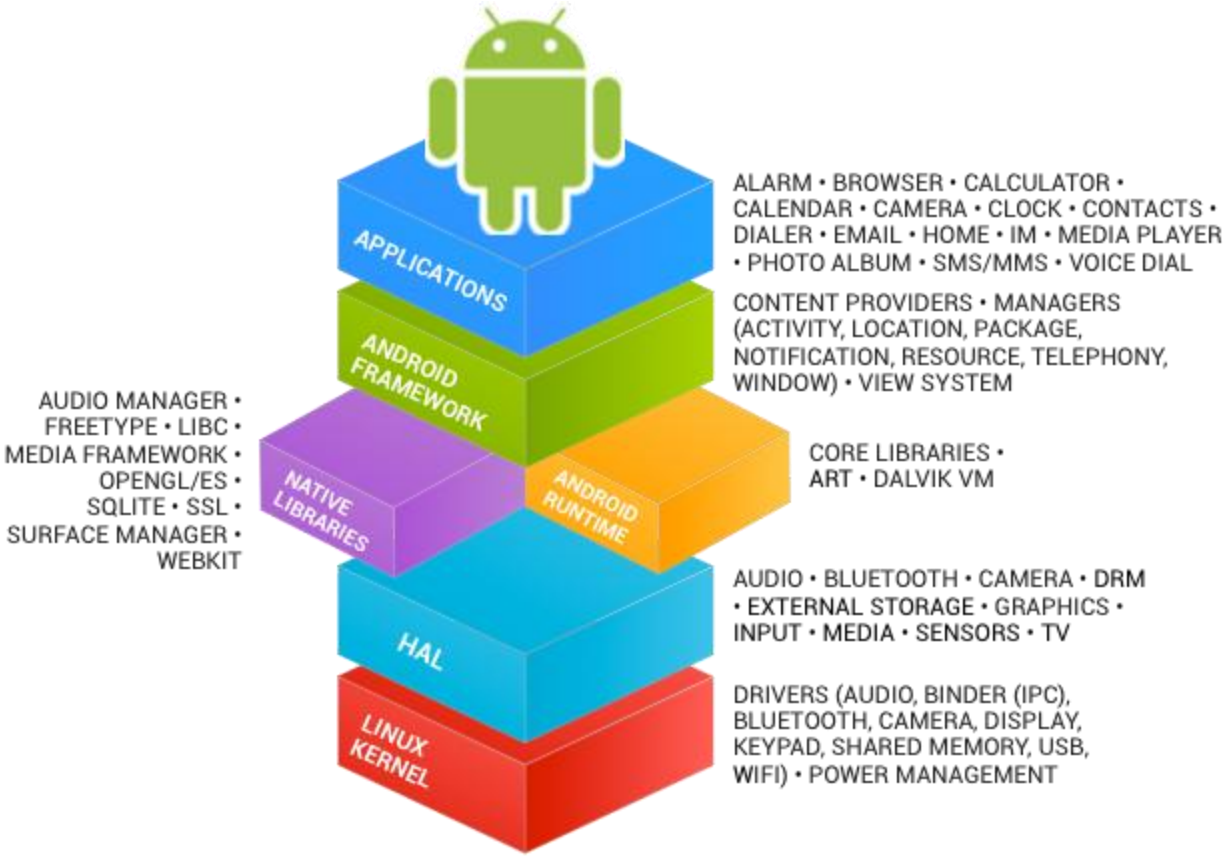
Now your app is complete. You've built it to support a variety of screen sizes and densities, and tested it on the Android emulator and on real devices. You're ready to ship your app.

How you proceed depends on a variety of factors, such as your monetization strategy and which types of devices your app supports. Everything you need to get started with this process is available in the Distribute section.

Go!

Get started by installing Android Studio—the official IDE for Android development, which includes the Android SDK tools.

Архитектура ОС Android



See https://source.android.com/source/index.html

Linux Kernel

- The basic layer is the Linux kernel. The whole Android OS is built on top of the Linux 2.6 Kernel with some further architectural changes made by Google. It is this Linux that interacts with the hardware and contains all the essential hardware drivers. Drivers are programs that control and communicate with the hardware. For example, consider the Bluetooth function. All devices has a Bluetooth hardware in it. Therefore the kernel must include a Bluetooth driver to communicate with the Bluetooth hardware. The Linux kernel also acts as an abstraction layer between the hardware and other software layers. Android uses the Linux for all its core functionality such as Memory management, process management, networking, security settings etc. As the Android is built on a most popular and proven foundation, it made the porting of Android to variety of hardware, a relatively painless task.

See http://www.android-app-market.com/android-architecture.html

Libraries

- The next layer is the Android's native libraries. It is this layer that enables the device to handle different types of data. These libraries are written in c or c++ language and are specific for a particular hardware.
- **Some of the important native libraries include the following:**
  - **Surface Manager:** It is used for compositing window manager with off-screen buffering. Off-screen buffering means you cant directly draw into the screen, but your drawings go to the off-screen buffer. There it is combined with other drawings and form the final screen the user will see. This off screen buffer is the reason behind the transparency of windows.
  - **Media framework:** Media framework provides different media codecs allowing the recording and playback of different media formats
  - **SQLite:** SQLite is the database engine used in android for data storage purposes
  - **WebKit:** It is the browser engine used to display HTML content
  - **OpenGL:** Used to render 2D or 3D graphics content to the screen

See http://www.android-app-market.com/android-architecture.html

Android Runtime

- Android Runtime consists of Dalvik Virtual machine and Core Java libraries.
- **Dalvik/Art Virtual Machine**
  - It is a type of JVM used in android devices to run apps and is optimized for low processing power and low memory environments. Unlike the JVM, the Dalvik Virtual Machine doesn't run .class files, instead it runs .dex files. .dex files are built from .class file at the time of compilation and provides hifger efficiency in low resource environments. The Dalvik VM allows multiple instance of Virtual machine to be created simultaneously providing security, isolation, memory management and threading support. It is developed by Dan Bornstein of Google.
- **Core Java Libraries**
  These are different from Java SE and Java ME libraries. However these libraries provides most of the functionalities defined in the Java SE libraries.

See http://www.android-app-market.com/android-architecture.html

Application Framework

- These are the blocks that our applications directly interacts with. These programs manage the basic functions of phone like resource management, voice call management etc. As a developer, you just consider these are some basic tools with which we are building our applications.
- **Important blocks of Application framework are:**
- **Activity Manager**: Manages the activity life cycle of applications
- **Content Providers:** Manage the data sharing between applications
- **Telephony Manager:** Manages all voice calls. We use telephony manager if we want to access voice calls in our application.
- **Location Manager:** Location management, using GPS or cell tower
- **Resource Manager:** Manage the various types of resources we use in our Application

See http://www.android-app-market.com/android-architecture.html

Applications

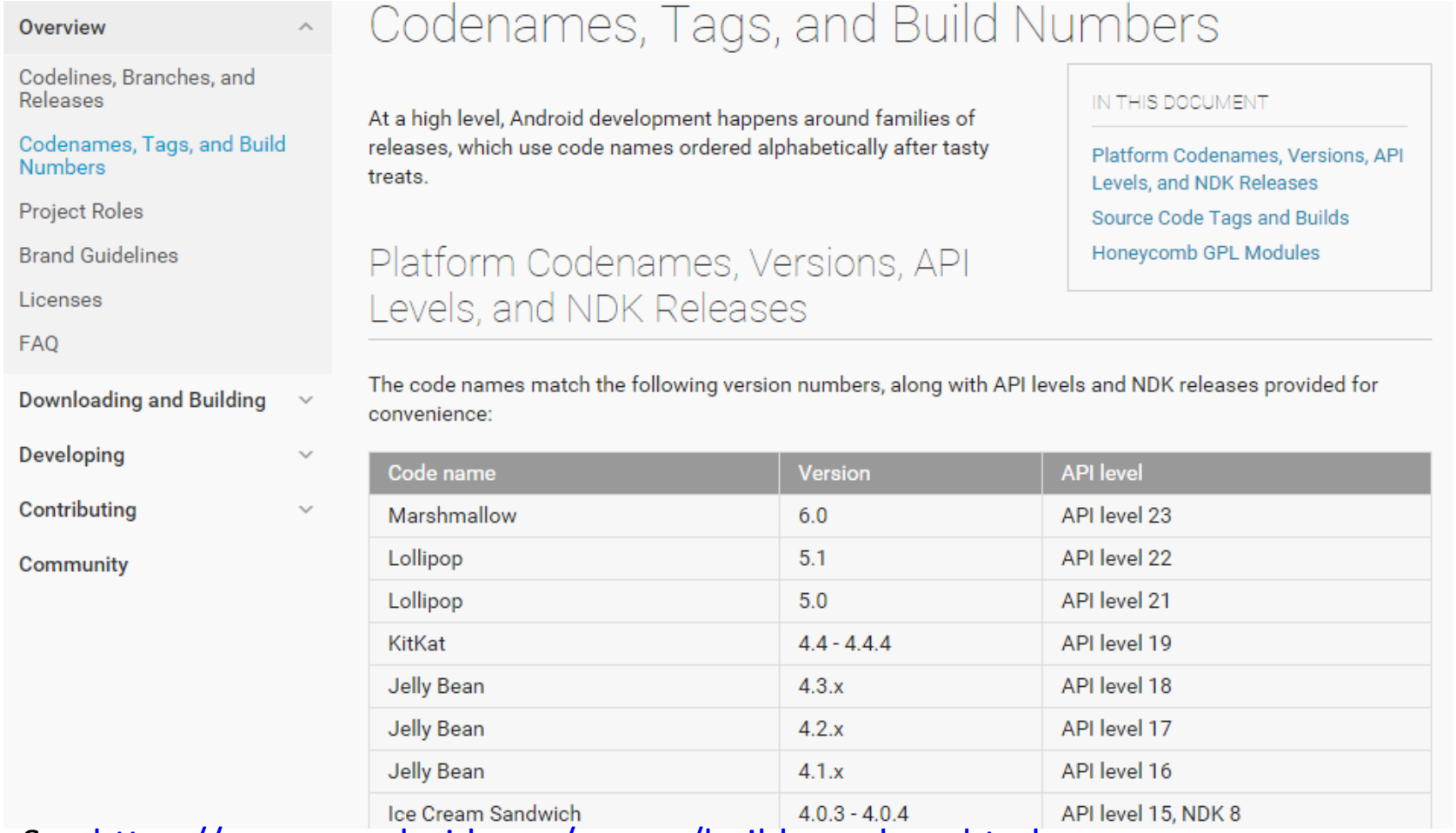
- Applications are the top layer in the Android architecture and this is where our applications are gonna fit. Several standard applications comes pre-installed with every device, such as:
  - SMS client app
  - Dialer
  - Web browser
  - Contact manager
- As a developer we are able to write an app which replace any existing system app. That is, you are not limited in accessing any particular feature. You are practically limitless and can whatever you want to do with the android (as long as the users of your app permits it). Thus Android is opening endless opportunities to the developer.

See http://www.android-app-market.com/android-architecture.html

ВОПРОС

- Зачем вообще нужна ОС на (мобильном) устройстве?

Имена релизов и версии API



Overview
Codelines, Branches, and Releases
Codenames, Tags, and Build Numbers
Project Roles
Brand Guidelines
Licenses
FAQ
Downloading and Building
Developing
Contributing
Community

Codenames, Tags, and Build Numbers

At a high level, Android development happens around families of releases, which use code names ordered alphabetically after tasty treats.

IN THIS DOCUMENT
Platform Codenames, Versions, API Levels, and NDK Releases
Source Code Tags and Builds
Honeycomb GPL Modules

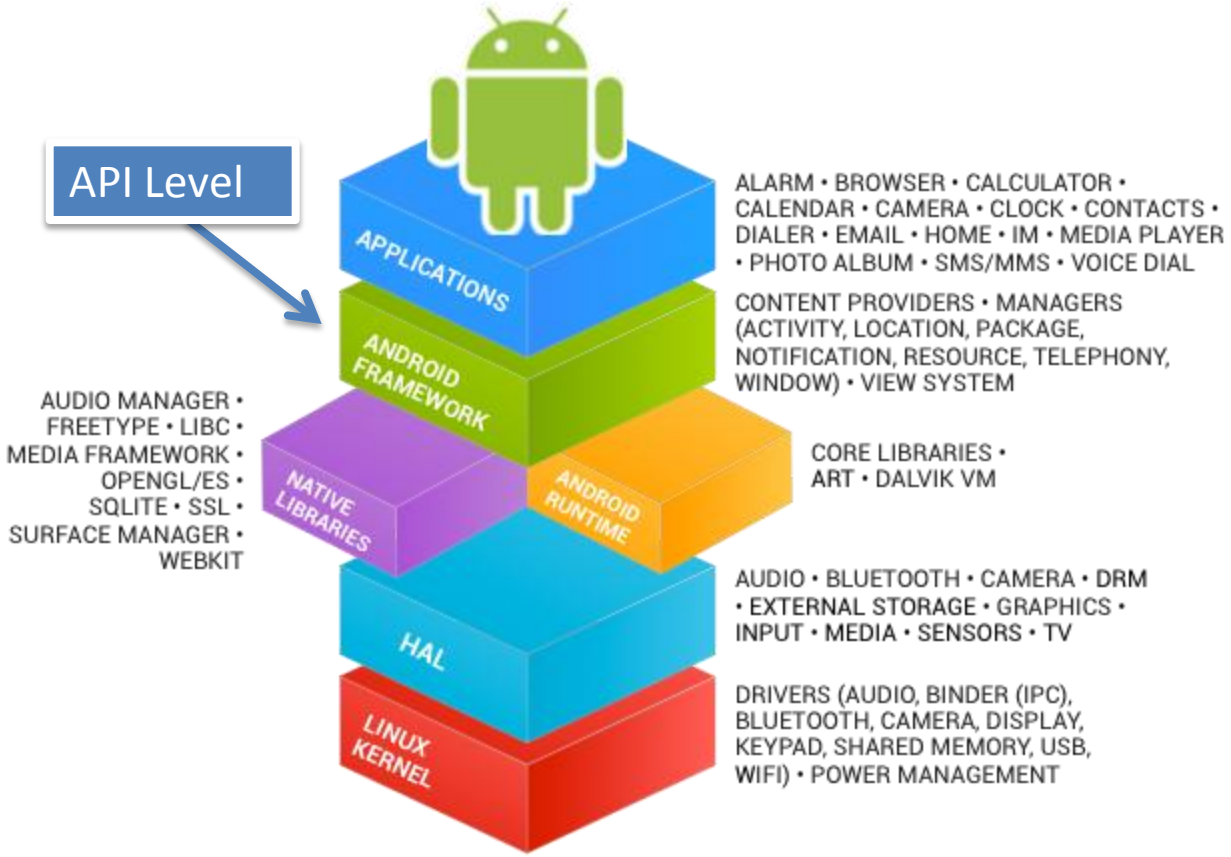
Platform Codenames, Versions, API Levels, and NDK Releases

The code names match the following version numbers, along with API levels and NDK releases provided for convenience:

Code name	Version	API level
Marshmallow	6.0	API level 23
Lollipop	5.1	API level 22
Lollipop	5.0	API level 21
KitKat	4.4 - 4.4.4	API level 19
Jelly Bean	4.3.x	API level 18
Jelly Bean	4.2.x	API level 17
Jelly Bean	4.1.x	API level 16
Ice Cream Sandwich	4.0.3 - 4.0.4	API level 15, NDK 8

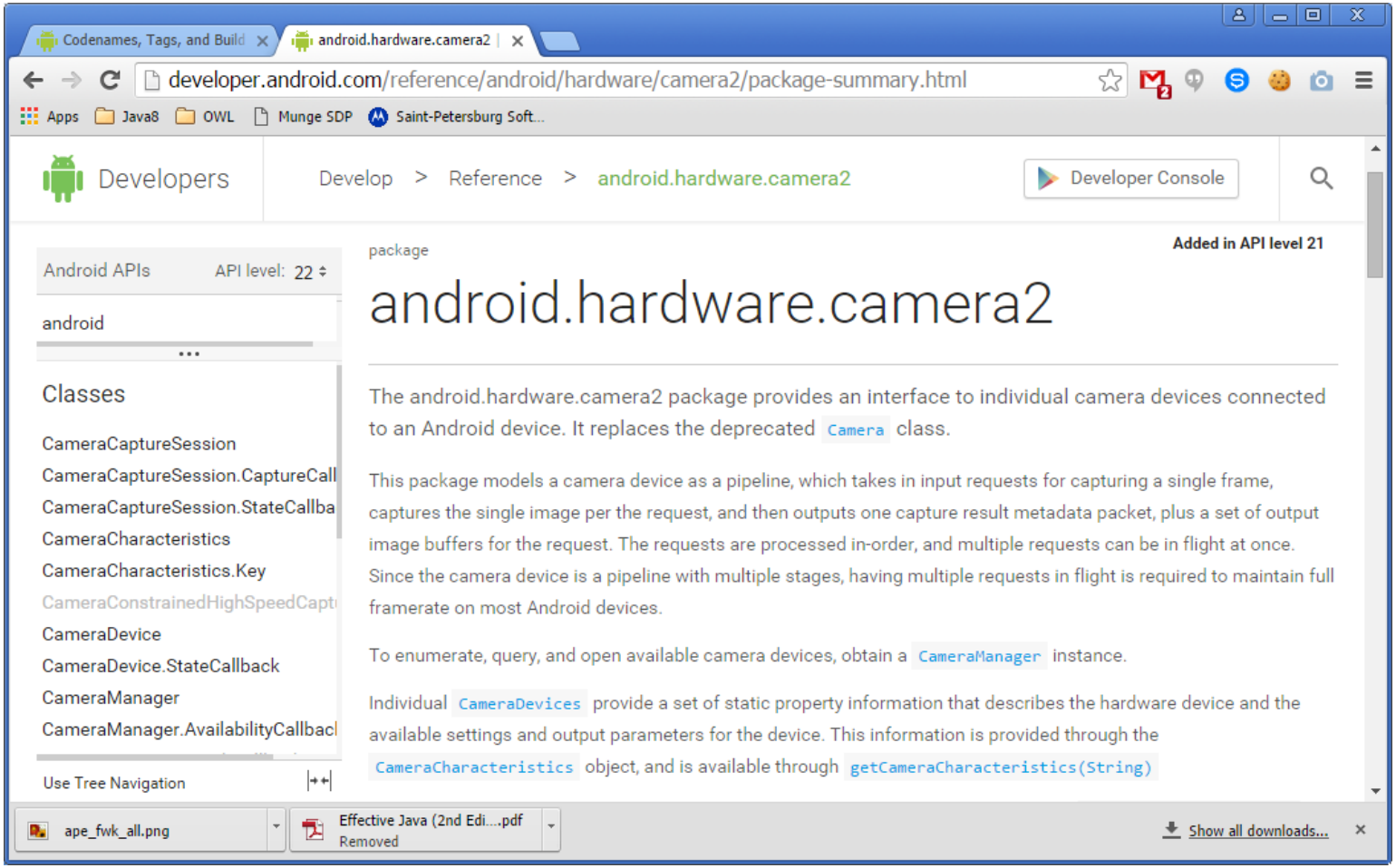
See https://source.android.com/source/build-numbers.html

Версия API в архитектуре Android



See https://source.android.com/source/index.html

Значимость версии API для разработчика



Инструментарий Разработчика (IntelliJ)

- Android Studio
  - http://developer.android.com/sdk/index.html

Инструментарий Разработчика (Eclipse)

- Android SDK
  - http://developer.android.com/sdk/index.html
- Eclipse IDE for Mobile Developers
  - http://eclipse.org/mobile/
- ADT Plugin для Eclipse
  - https://dl-ssl.google.com/android/eclipse/
- Java SE Development Kit 7
  - http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html

Информация больше не доступна по URL: http://developer.android.com/sdk/exploring.html

Android SDK (1)

- SDK Tools (<sdk>/tools/)
  - Инструменты для разработки, отладки и тестирования (рассмотрен далее)
- SDK Platform-tools (<sdk>/platform-tools/)
  - Инструменты для разработки, отладки и тестирования (рассмотрен далее)
- Documentation (<sdk>/docs/)
  - Offline копия фоициальной документации Android platform APIs.

Android SDK (2)

- SDK Platform (<sdk>/platforms/<android-version>/)
  - Одна SDK платформа для каждой версии Android.
- System Images (<sdk>/platforms/<android-version>/)
- Sources for Android SDK (<sdk>/sources/)
- Samples for SDK (<sdk>/platforms/<android-version>/samples/)
- Android Support & Google APIs

http://developer.android.com/tools/help/index.html

Android SDK Tools

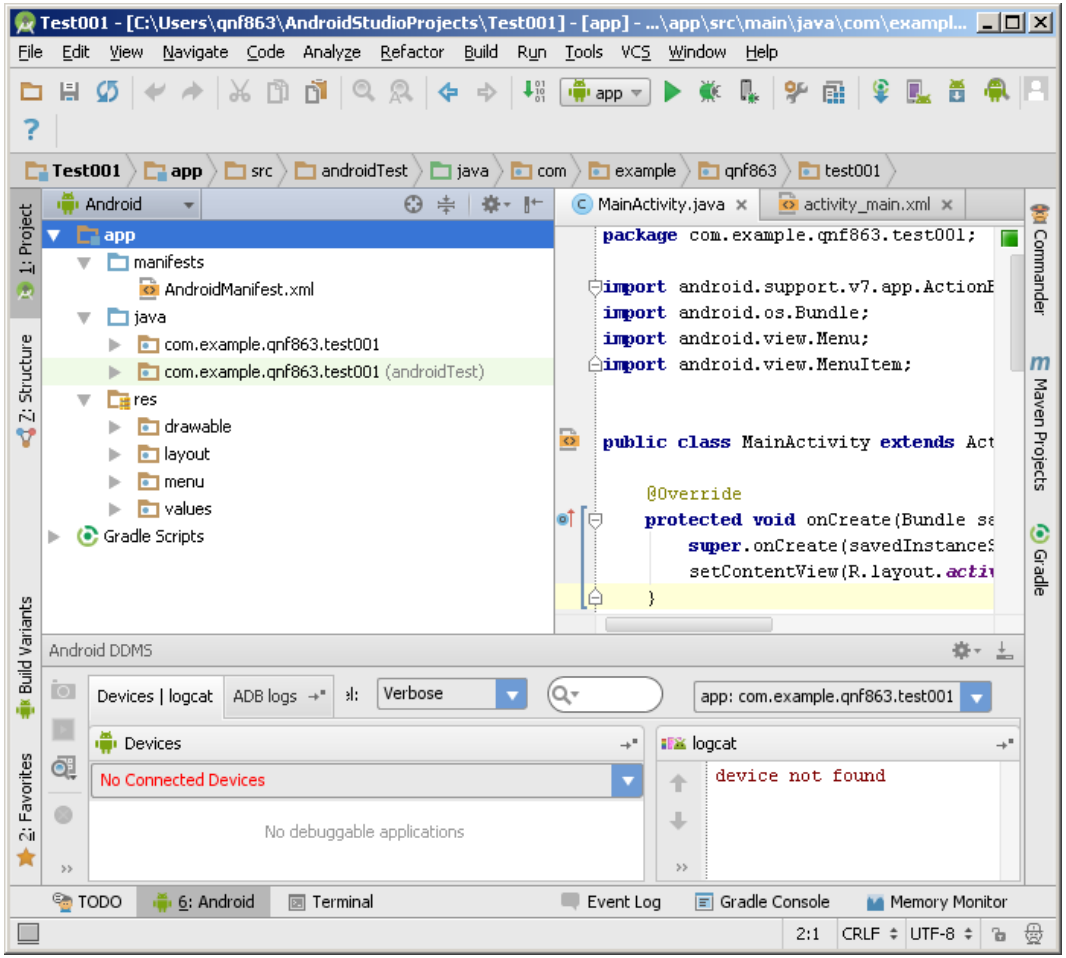
- Android (tools/android)
  - Позволяет управлять виртуальными устройствами (AVD), пректами и установленными компонентами SDK.
- Device Monitor (tools/monitor)
  - Инструмент для отладки и анализа приложений во время исполнения на устройстве или эмуляторе.
- Android Emulator (tools/emulator)
- Monkey
  - Генератор псевдо-случайной последовательности действий пользователя для целей тестирования.
- Monkeyrunner
  - Предоставляет API для упраления устройством или эмулятором для целей тестирования.
- И многое другое ...

http://developer.android.com/tools/help/index.html

Android SDK Platform Tools

- Android Debug Bridge (adb).
  - Позволяет управлять устройством или эмулятором
  - Позволяет инсталировать Android application (.apk) файлы на устройство.
- Остальные platform tools, такое как обычно вызываются средой разработки:
  - aidl, aapt, dexdump и dx

Структура проекта Android



Структура Android приложения

- Приложение предоставляет несколько точек входа
  - Android приложение – это совокупность компонентов, которые могут запускаться **операционной системой** независимо друг от друга.

Программные компоненты Android

- Activities
- Services
- Content Providers
- Broadcast Receivers

Разработчику нужно унаследовать соответствующий базовый класс компонента

Компоненты активируются с помощью Intent

Activities

- Class: android.app.Activity
- “An activity is a single, focused thing that the user can do. ” - http://developer.android.com
- Почти все activities взаимодействуют с пользователем, поэтому базовый класс Activity создает окно, в котором разработчик может отобразить UI.

Services

- Class: android.app.Service
- “Service is an application component representing either an application's desire to perform a longer-running operation while not interacting with the user or to supply functionality for other applications to use” - http://developer.android.com
- В Android есть 2 типа сервисов:
  - **Unbound Services (aka Started Service)**
    - Однажды запущенный, сервис работает до тех пор, пока сам себя не остановит.
  - **Bound Services**
    - Сервис работает только пока работает запустивший его компонент. Обычно сервис обменивается данными с этим компонентом.

Content Providers

- Class: android.content.ContentProvider
- “Content providers are <…> providing content to applications.” - http://developer.android.com
- Используются в случае, когда необходимо предоставлять доступ к одним и тем же данным разным приложениям.
  - Если разделять даные не требуется, удобно воспользоваться встроенной поддержкой SQLiteDatabase.

- Пример: приложение «контакты»

Broadcast Receivers

- Class: android.content.BroadcastReceiver
- “Broadcast receivers are <…> used to receive messages that are broadcasted by the Android system or other Android applications.” - http://developer.android.com
- Примеры:
  - Низкий заряд батареи
  - Экран выключен
  - Изменился временной пояс
  - И т.п.

Intents

- Class: android.content.Intent
- “An intent is an abstract description of an operation to be performed .” - http://developer.android.com
- С помощью Intent можно запускать Activity, BroadcastReceiver и Service в своем и чужих приложениях.

- В Android различают два типа Intent:
  - **Explicit Intents:**
    - Обычно используется внутри приложения.
    - Intent содержит полное имя компонента, который должен его обработать.
  - **Implicit Intents:**
    - Intent послылается ОС и Android сам ищет компонент, который может его обработать.
    - Пример: send an e-mail

ВОПРОС

Список источников

- http://developer.android.com
- http://sources.android.com