Android

Derleme (bitmedi)

- Android Studio 3.
  - Android 7.0
- Her hafta Ödev %20
- Proje var %20
- Vize %20
- Final %40
- Lecture Notes
  - Udemy
  - http://www.vogella.com/tutorials/android.html
  - https://www.tutorialspoint.com//android_online_training/index.asp

- Hafta 1 Android işletim sistemi İlk Program.
- Hafta 2 UI etkileşimli uygulamlar yapılacak
- Hafta 3 UI etkileşimli uygulamlar yapılacak

What is Android?

- A software platform and operating system for mobile devices
- Based on the Linux kernel
- Developed by Google and later the Open Handset Alliance (OHA)
- Allows writing managed code in the Java language
- Possibility to write applications in other languages and compiling it to ARM native code (support of Google? No)
- Unveiling of the Android platform was announced on 5 November 2007 with the founding of OHA

1. Introduction (2)

What is the Open Handset Alliance (OHA)? (1)

→ It's a consortium of several companies

mao Intensive Programme

1. Introduction (3)

What is the Open Handset Alliance (OHA)? (2)

- Devoted to advancing open standards for mobile devices
- Develop technologies that will significantly lower the cost of developing and distributing mobile devices and services

1. Introduction (4)

License

Android is under version 2 of the Apache Software License (ASL)

2. Platform (1)

2.1 Hardware

Android is not a single piece of hardware; it's a complete, end-to-end software platform that can be adapted to work on any number of hardware configurations. Everything is there, from the bootloader all the way up to the applications.

2. Platform (2)

2.2 Operating System(s)

- Android uses Linux for its device drivers, memory management, process management, and networking.
- The next level up contains the Android native libraries. They are all written in C/C++ internally, but you'll be calling them through Java interfaces. In this layer you can find the Surface Manager, 2D and 3D graphics, Media codecs, the SQL database (SQLite), and a native web browser engine (WebKit).
- Dalvik Virtual Machine. Dalvik runs dex files, which are coverted at compile time from standard class and jar files.

2. Platform (3)

2.3 Network Connectivity

It supports wireless communications using:

- GSM mobile-phone technology
- 3G
- Edge
- 802.11 Wi-Fi networks

2. Platform (4)

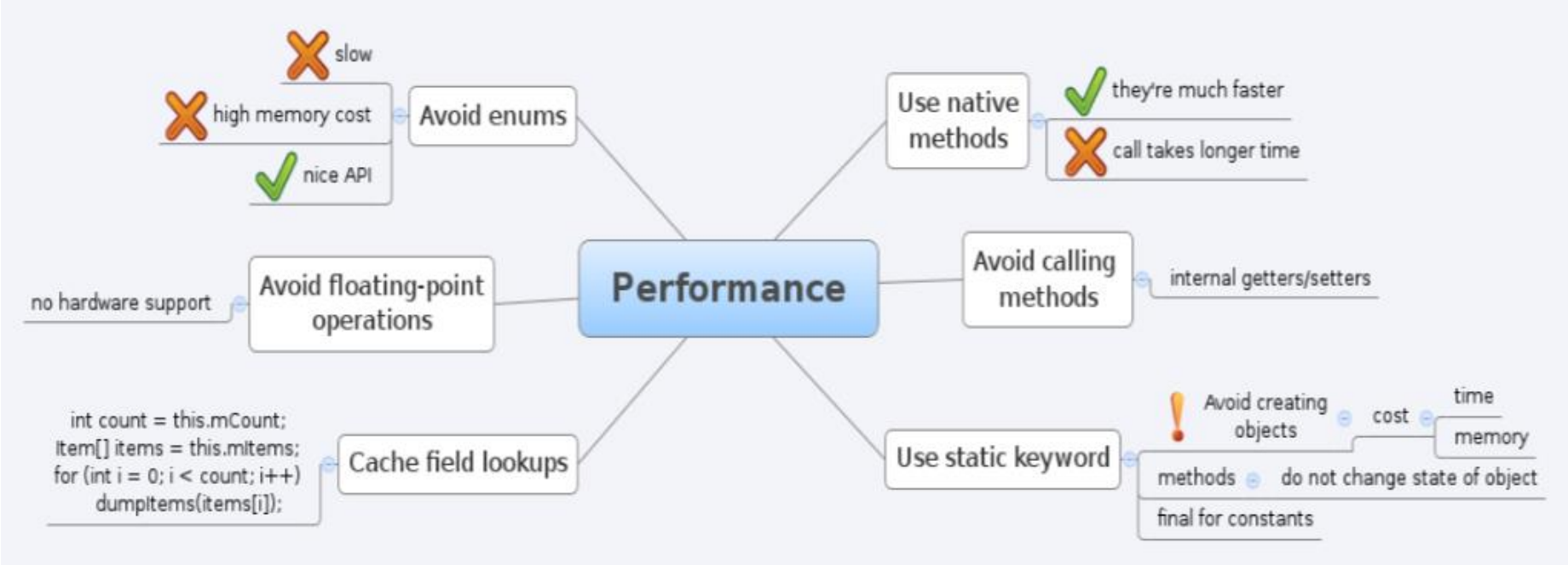
2.4 Security

Android is a multi-process system, in which each application (and parts of the system) runs in its own process. Most security between applications and the system is enforced at the process level through standard Linux facilities, such as user and group IDs that are assigned to applications.

Additional finer-grained security features are provided through a "permission" mechanism that enforces restrictions on the specific operations that a particular process can perform, and per-URI permissions for granting ad-hoc access to specific pieces of data.

2. Platform (5)

2.5 Performance



Intensive Programme

2. Platform (6)

2.6 Future possibilities

- Google Android Sales to Overtake iPhone in 2012
- The OHA is committed to make their vision a reality: to deploy the Android platform for every mobile operator, handset manufacturers and developers to build innovative devices
- Intel doesn’t want to lose ownership of the netbook market, so they need to prepare for anything, including Android
- Fujitsu launched an initiative to offer consulting and engineering expertise to help run Android on embedded hardware, which aside from cellphones, mobile internet devices, and portable media players, could include GPS devices, thin-client computers and set-top boxes.
- More Android devices are coming and some will push the envelope even further

3. Software development (1)

3.1 Development requirements

- Java
- Android SDK
- Eclipse IDE (optional)

3. Software development (2)

3.2 IDE and Tools

Android SDK

- Class Library
- Developer Tools
    - dx – Dalvik Cross-Assembler
    - aapt – Android Asset Packaging Tool
    - adb – Android Debug Bridge
    - ddms – Dalvik Debug Monitor Service
- Emulator and System Images
- Documentation and Sample Code

Eclipse IDE + ADT (Android Development Tools)

- Reduces Development and Testing Time
- Makes User Interface-Creation easier
- Makes Application Description Easier

3. Software development (3)

3.3 Programming Language(s)

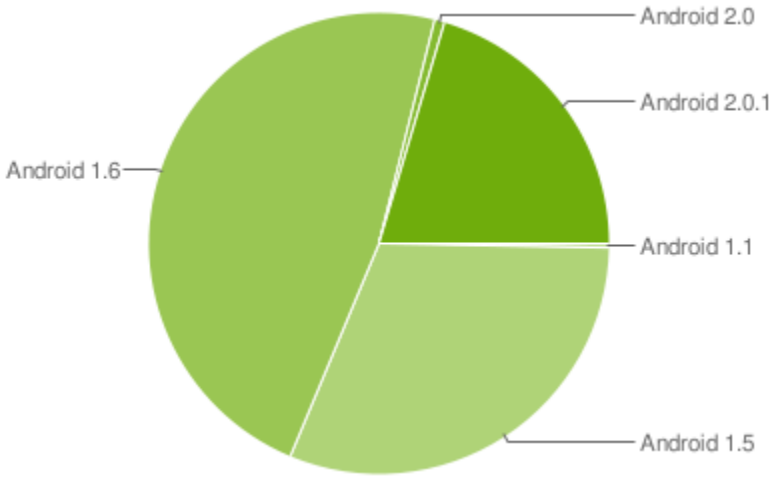
- Java – officially supported
- C/C++ – also possible but not supported

4. Overall evaluation (1)

4.1 Advantages

There are a host of advantages that Google's Android will derive from being an **open source software**. Some of the advantages include:

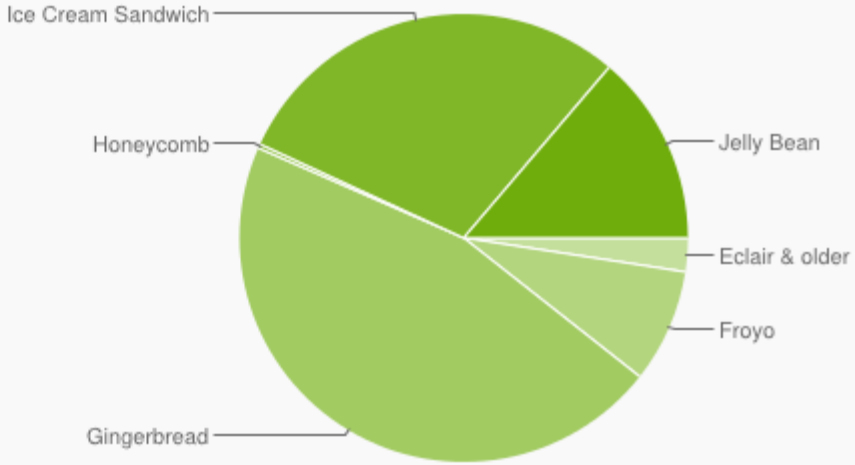
- The ability for anyone to customize the Google Android platform
- The consumer will benefit from having a wide range of mobile applications to choose from since the monopoly will be broken by Google Android
- Men will be able to customize a mobile phones using Google Android platform like never before
- Features like weather details, opening screen, live RSS feeds and even the icons on the opening screen will be able to be customized
- As a result of many mobile phones carrying Google Android, companies will come up with such innovative products like the location
- In addition the entertainment functionalities will be taken a notch higher by Google Android being able to offer online real time multiplayer games



Platform Versions

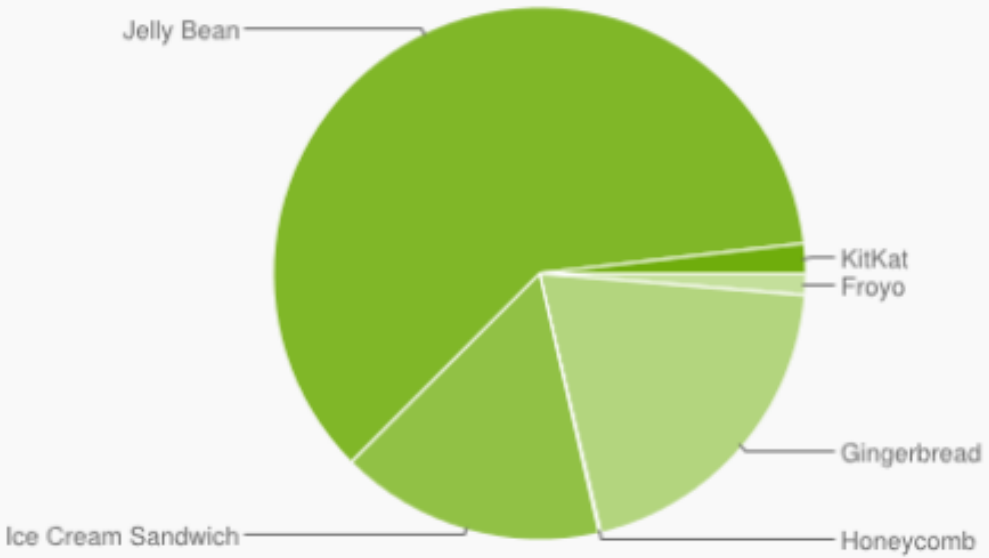
http://developer.android.com/about/dashboards/index.html

Version	Codename	API	Distribution
1.6	Donut	4	0.2%
2.1	Eclair	7	2.2%
2.2	Froyo	8	8.1%
2.3 - 2.3.2	Gingerbread	9	0.2%
2.3.3 - 2.3.7		10	45.4%
3.1	Honeycomb	12	0.3%
3.2		13	1.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	29.0%
4.1	Jelly Bean	16	12.2%
4.2		17	1.4%



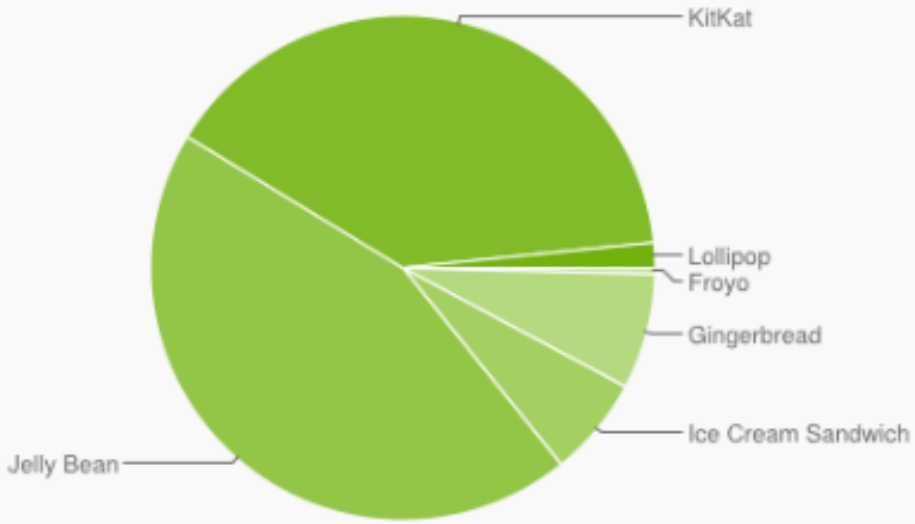
Data collected during a 14-day period ending on February 4, 2013

Version	Codename	API	Distribution
2.2	Froyo	8	1.3%
2.3.3 - 2.3.7	Gingerbread	10	20.0%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	16.1%
4.1.x	Jelly Bean	16	35.5%
4.2.x		17	16.3%
4.3		18	8.9%
4.4	KitKat	19	1.8%



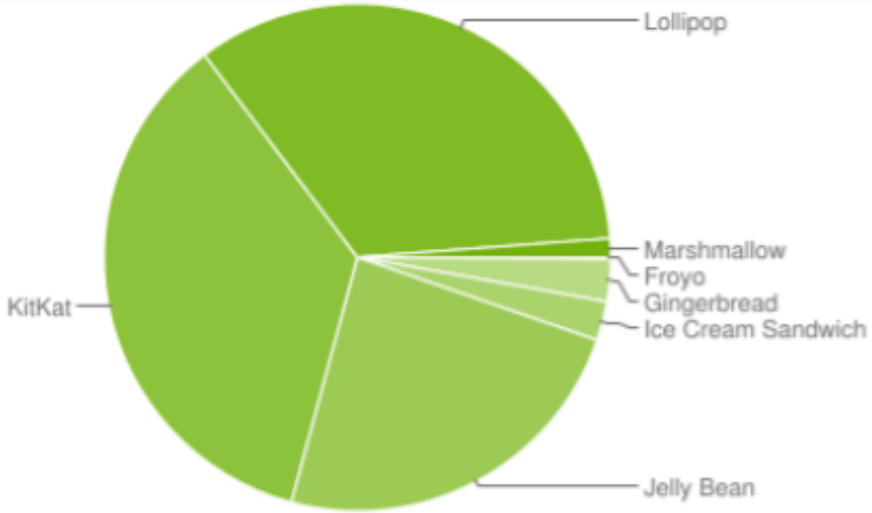
Data collected during a 7-day period ending on February 4, 2014.
Any versions with less than 0.1% distribution are not shown.

Version	Codename	API	Distribution
2.2	Froyo	8	0.4%
2.3.3 - 2.3.7	Gingerbread	10	7.4%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	6.4%
4.1.x	Jelly Bean	16	18.4%
4.2.x		17	19.8%
4.3		18	6.3%
4.4	KitKat	19	39.7%
5.0	Lollipop	21	1.6%



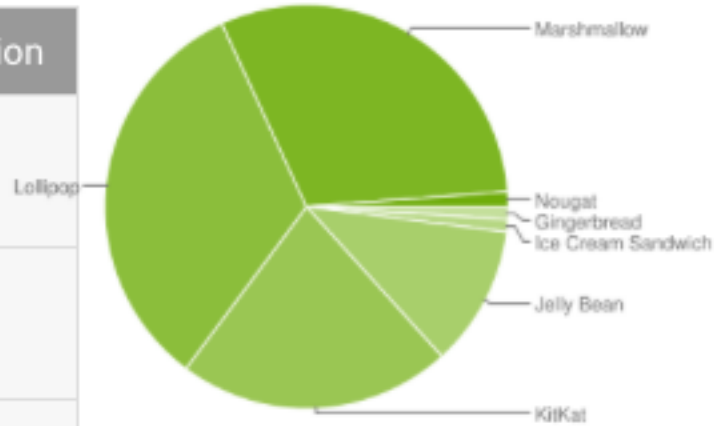
Data collected during a 7-day period ending on February 2, 2015.
Any versions with less than 0.1% distribution are not shown.

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.7%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	2.5%
4.1.x	Jelly Bean	16	8.8%
4.2.x		17	11.7%
4.3		18	3.4%
4.4	KitKat	19	35.5%
5.0	Lollipop	21	17.0%
5.1		22	17.1%
6.0	Marshmallow	23	1.2%



ata collected during a 7-day period ending on February 1, 2016.

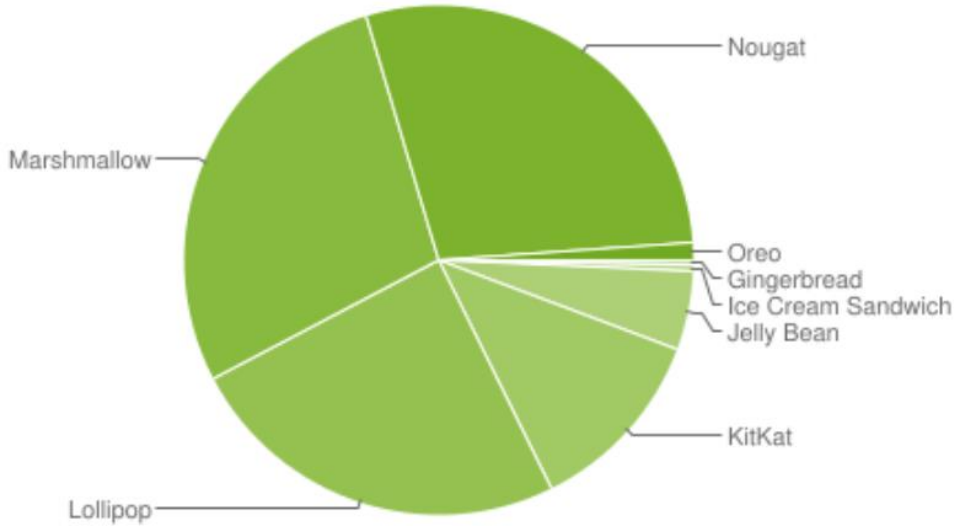
Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	1.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.0%
4.1.x	Jelly Bean	16	4.0%
4.2.x		17	5.7%
4.3		18	1.6%
4.4	KitKat	19	21.9%
5.0	Lollipop	21	9.8%
5.1		22	23.1%
6.0	Marshmallow	23	30.7%
7.0	Nougat	24	0.9%
7.1		25	0.3%



Data collected during a 7-day period ending on February 6, 2017.

Any versions with less than 0.1% distribution are not shown.

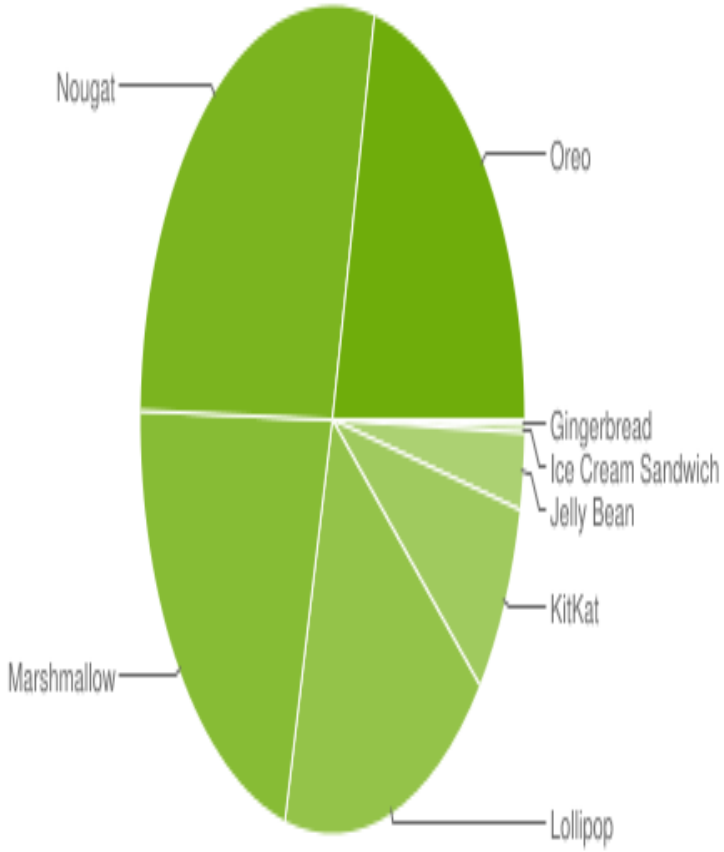
Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.4%
4.1.x	Jelly Bean	16	1.7%
4.2.x		17	2.6%
4.3		18	0.7%
4.4	KitKat	19	12.0%
5.0	Lollipop	21	5.4%
5.1		22	19.2%
6.0	Marshmallow	23	28.1%
7.0	Nougat	24	22.3%
7.1		25	6.2%
8.0	Oreo	26	0.8%
8.1		27	0.3%



Data collected during a 7-day period ending on February 5, 2018.

Any versions with less than 0.1% distribution are not shown.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.2%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.1%
4.2.x		17	1.5%
4.3		18	0.4%
4.4	KitKat	19	7.6%
5.0	Lollipop	21	3.5%
5.1		22	14.4%
6.0	Marshmallow	23	21.3%
7.0	Nougat	24	18.1%
7.1		25	10.1%
8.0	Oreo	26	14.0%
8.1		27	7.5%



Data collected during a 7-day period ending on October 26, 2018 (update coming soon: data feed under maintenance).

Any versions with less than 0.1% distribution are not shown.

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.0 Ice Cream Sandwich	15	
4.1 Jelly Bean	16	99.8%
4.2 Jelly Bean	17	99.2%
4.3 Jelly Bean	18	98.4%
4.4 KitKat	19	98.1%
5.0 Lollipop	21	94.1%
5.1 Lollipop	22	92.3%
6.0 Marshmallow	23	84.9%
7.0 Nougat	24	73.7%
7.1 Nougat	25	66.2%
8.0 Oreo	26	60.8%
8.1 Oreo	27	53.5%
9.0 Pie	28	39.5%
10. Android 10	29	8.2%

Android 10

System

Foldables support
5G support
Gesture navigation
ART optimizations
Neural Networks API 1.2
Thermal API

User Interface

Smart Reply in notifications
Dark theme
Settings panels
Sharing shortcuts

Camera and media

Dynamic depth for photos
Audio playback capture
New codecs
Native MIDI API
Vulkan everywhere
Directional microphones

Security an

New locatio
Storage enc
TLS 1.3 by d
Platform har
Improved bi

2021 MART

https://developer.android.com/about/versions/10

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.1 Jelly Bean	16	
4.2 Jelly Bean	17	99.8%
4.3 Jelly Bean	18	99.5%
4.4 KitKat	19	99.4%
5.0 Lollipop	21	98.0%
5.1 Lollipop	22	97.3%
6.0 Marshmallow	23	94.1%
7.0 Nougat	24	89.0%
7.1 Nougat	25	85.6%
8.0 Oreo	26	82.7%
8.1 Oreo	27	78.7%
9.0 Pie	28	69.0%
10. Q	29	50.8%
11. R	30	24.3%

The minimum SDK version determines the lowest level of Android that your app will run on.

You typically want to target as many users as possible, so you would ideally want to support everyone -- with a minimum SDK version of 1. However, that has some disadvantages, such as lack of features, and very few people use devices that old anymore.

Your choice of minimum SDK level should be a tradeoff between the distribution of users you wish to target and the features that your application will need.

Click each Android Version/API level for more information.

ŞUBAT 20222

Worldwide Mobile Device Sales to End Users by Operating System in 3Q12 (Thousands of Units)

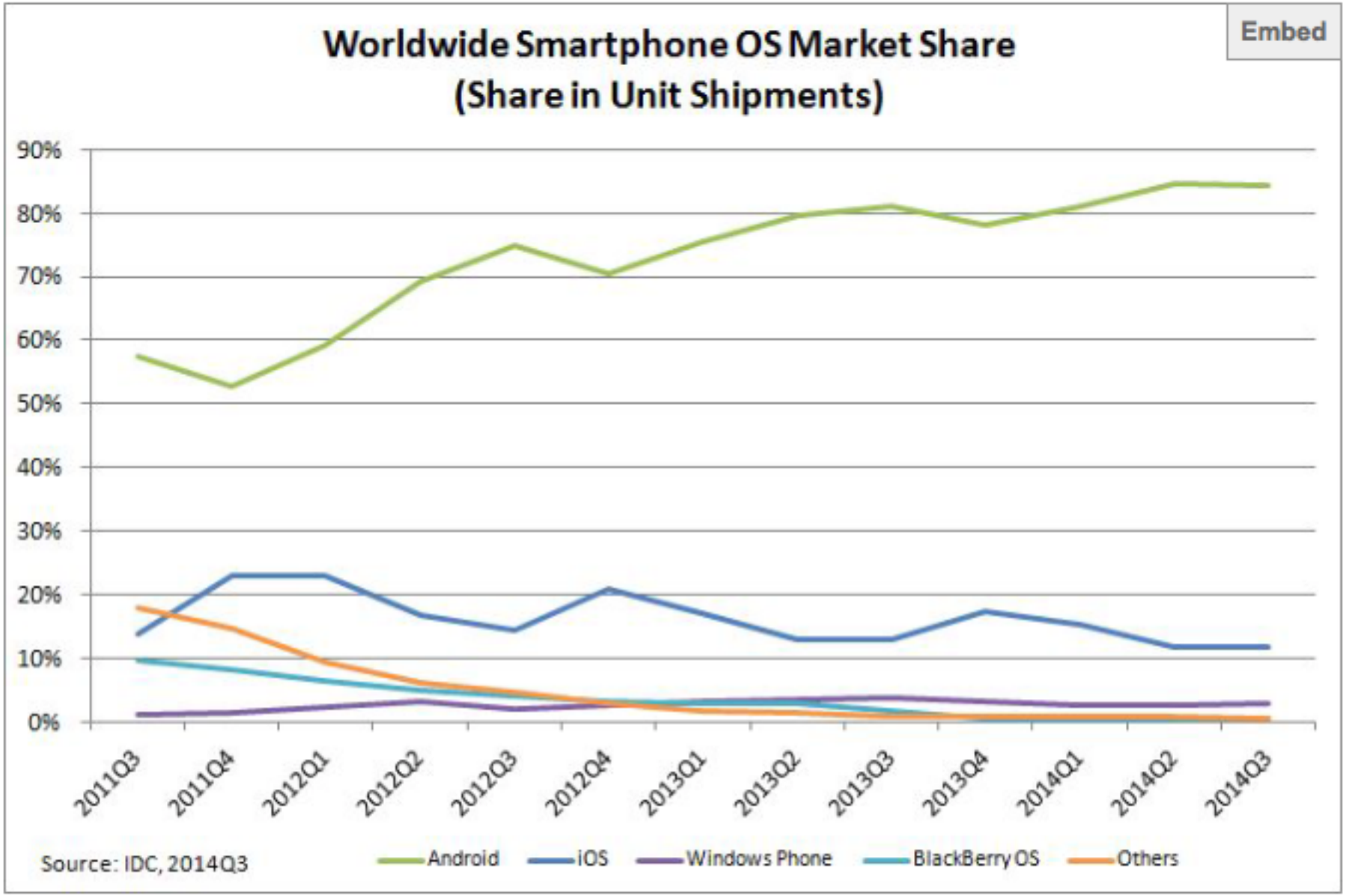
Operating System	3Q12 Units	3Q12 Market Share (%)	3Q11 Units	3Q11 Market Share (%)
Android	122,480.0	72.4	60,490.4	52.5
iOS	23,550.3	13.9	17,295.3	15.0
Research In Motion	8,946.8	5.3	12,701.1	11.0
Bada	5,054.7	3.0	2,478.5	2.2
Symbian	4,404.9	2.6	19,500.1	16.9
Microsoft	4,058.2	2.4	1,701.9	1.5
Others	683.7	0.4	1,018.1	0.9
Total	**169,178.6**	**100.0**	**115,185.4**	**100.0**

Source: Gartner (November 2012)

Note: shipments shown by vendor are for own-brand devices

SmartPhone Market Share *

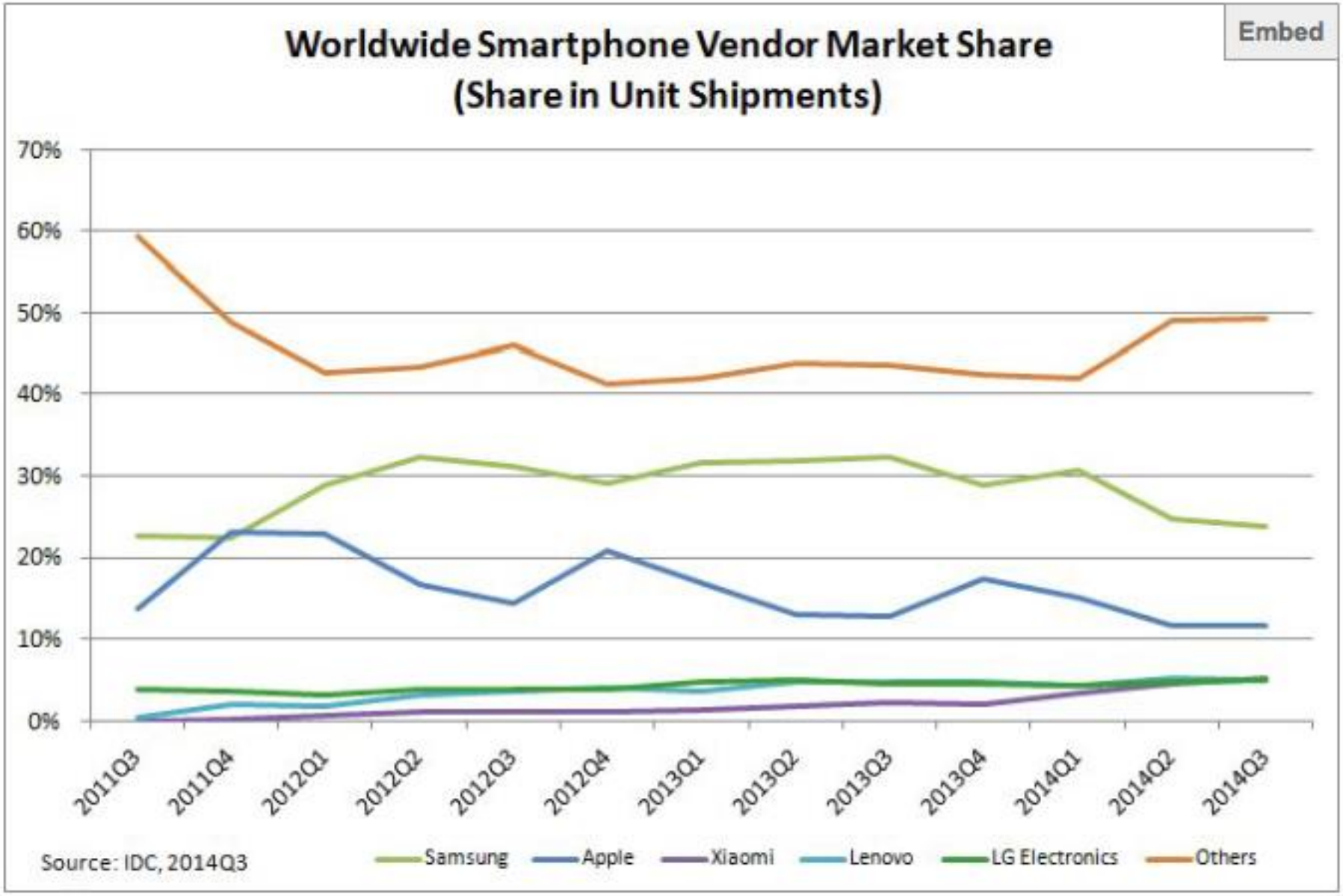
* http://bgr.com/2012/11/08/smartphone-market-share-q3-2012/


Embed
Worldwide Smartphone OS Market Share
(Share in Unit Shipments)
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
2011Q3
2011Q4
2012Q1
2012Q2
2012Q3
2012Q4
2013Q1
2013Q2
2013Q3
2013Q4
2014Q1
2014Q2
2014Q3
Source: IDC, 2014Q3
Android
iOS
Windows Phone
BlackBerry OS
Others

-

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
Q3 2014	84.4%	11.7%	2.9%	0.5%	0.6%
Q3 2013	81.2%	12.8%	3.6%	1.7%	0.6%
Q3 2012	74.9%	14.4%	2.0%	4.1%	4.5%
Q3 2011	57.4%	13.8%	1.2%	9.6%	18.0%

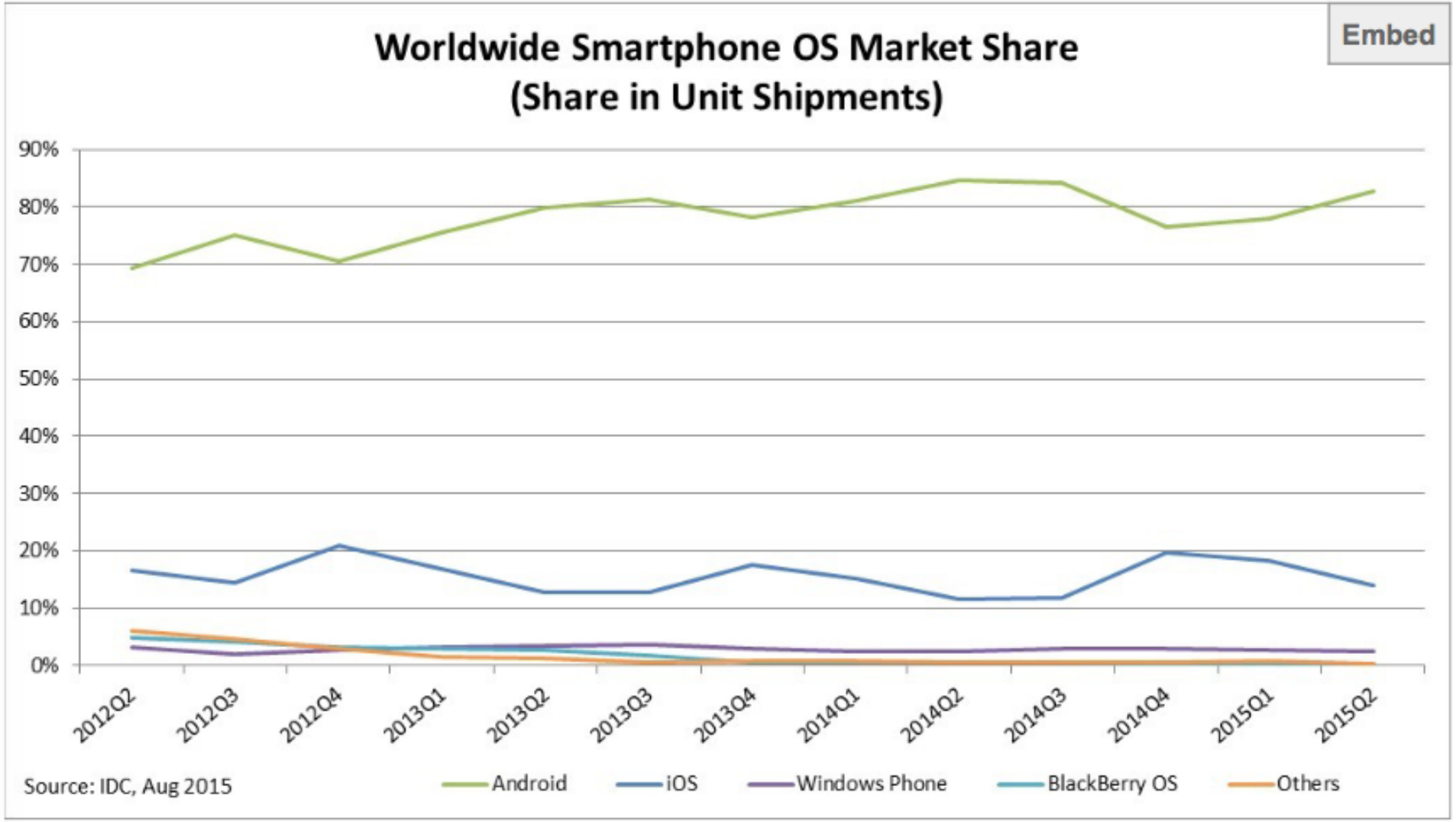
Source: IDC, 2014 Q3


Embed
Worldwide Smartphone Vendor Market Share
(Share in Unit Shipments)
70%
60%
50%
40%
30%
20%
10%
0%
2011Q3
2011Q4
2012Q1
2012Q2
2012Q3
2012Q4
2013Q1
2013Q2
2013Q3
2013Q4
2014Q1
2014Q2
2014Q3
Source: IDC, 2014Q3
Samsung
Apple
Xiaomi
Lenovo
LG Electronics
Others

-

Period	Samsung	Apple	Xiaomi	Lenovo	LG	Others
Q3 2014	23.7%	11.7%	5.2%	5.1%	5.0%	49.3%
Q3 2013	32.2%	12.8%	2.1%	4.7%	4.6%	43.6%
Q3 2012	31.2%	14.4%	1.0%	3.7%	3.7%	46.0%
Q3 2011	22.7%	13.8%	-	0.4%	3.7%	59.4%

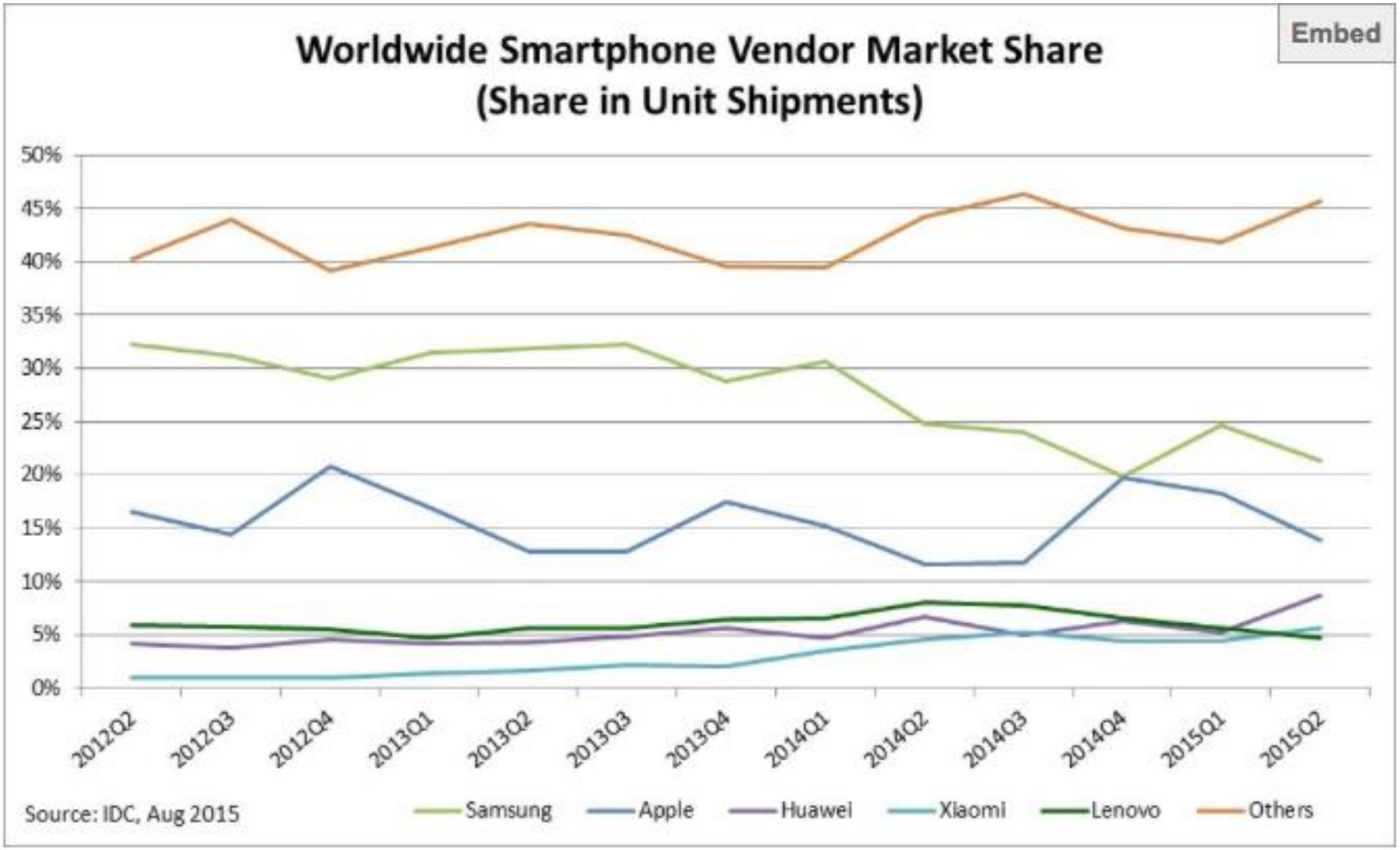
Source: IDC, 2014 Q3


Embed
Worldwide Smartphone OS Market Share
(Share in Unit Shipments)
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
2012Q2
2012Q3
2012Q4
2013Q1
2013Q2
2013Q3
2013Q4
2014Q1
2014Q2
2014Q3
2014Q4
2015Q1
2015Q2
Source: IDC, Aug 2015
Android
iOS
Windows Phone
BlackBerry OS
Others

-

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
2015Q2	82.8%	13.9%	2.6%	0.3%	0.4%
2014Q2	84.8%	11.6%	2.5%	0.5%	0.7%
2013Q2	79.8%	12.9%	3.4%	2.8%	1.2%
2012Q2	69.3%	16.6%	3.1%	4.9%	6.1%

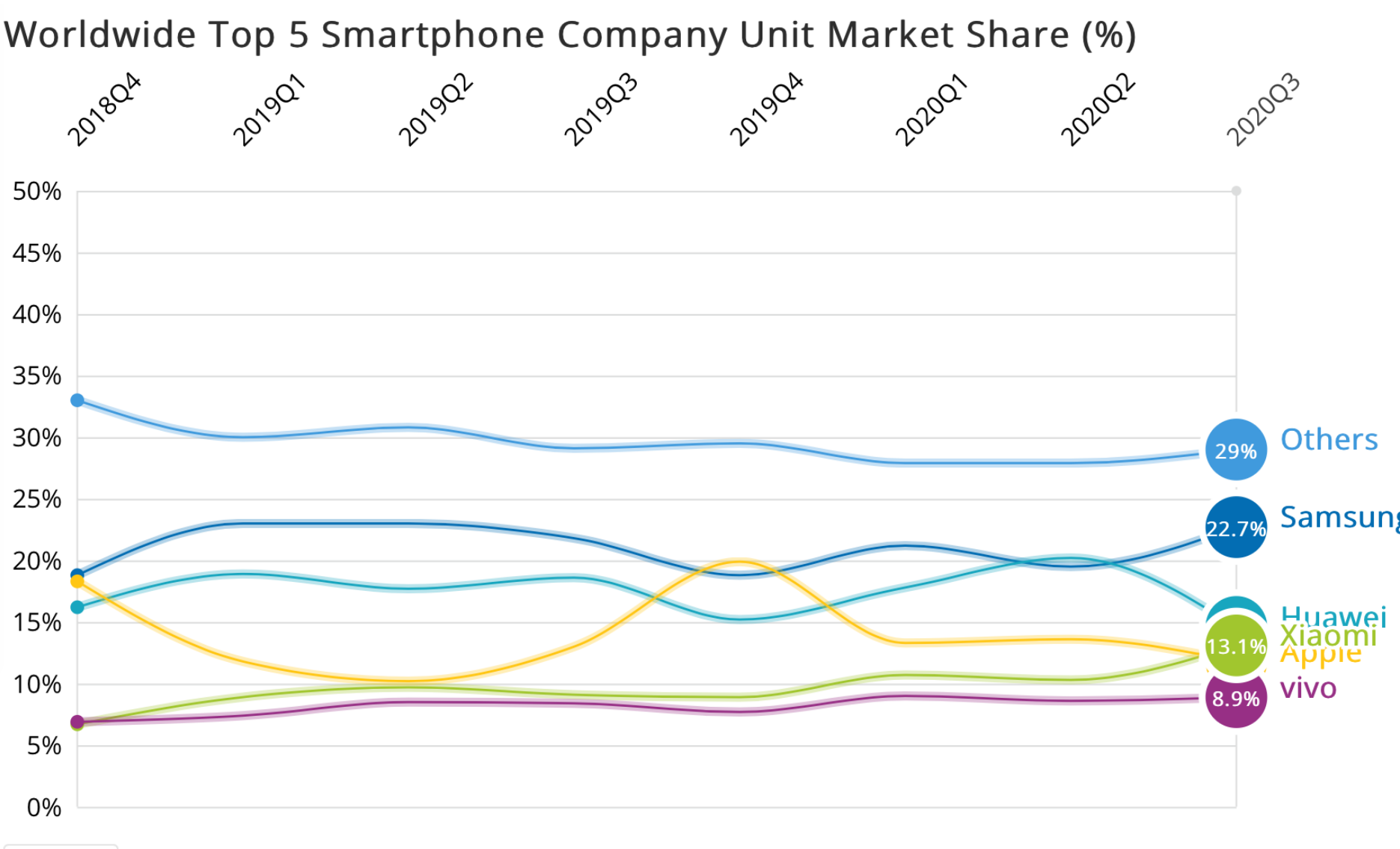
Source: IDC, Aug 2015


Embed
Worldwide Smartphone Vendor Market Share
(Share in Unit Shipments)
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
2012Q2
2012Q3
2012Q4
2013Q1
2013Q2
2013Q3
2013Q4
2014Q1
2014Q2
2014Q3
2014Q4
2015Q1
2015Q2
Source: IDC, Aug 2015
Samsung
Apple
Huawei
Xiaomi
Lenovo
Others

Period	Samsung	Apple	Huawei	Xiaomi	Lenovo*	Others
2015Q2	21.4%	13.9%	8.7%	5.6%	4.7%	45.7%
2014Q2	24.8%	11.6%	6.7%	4.6%	8.0%	44.3%
2013Q2	31.9%	12.9%	4.3%	1.7%	5.7%	43.6%
2012Q2	32.2%	16.6%	4.1%	1.0%	5.9%	40.2%

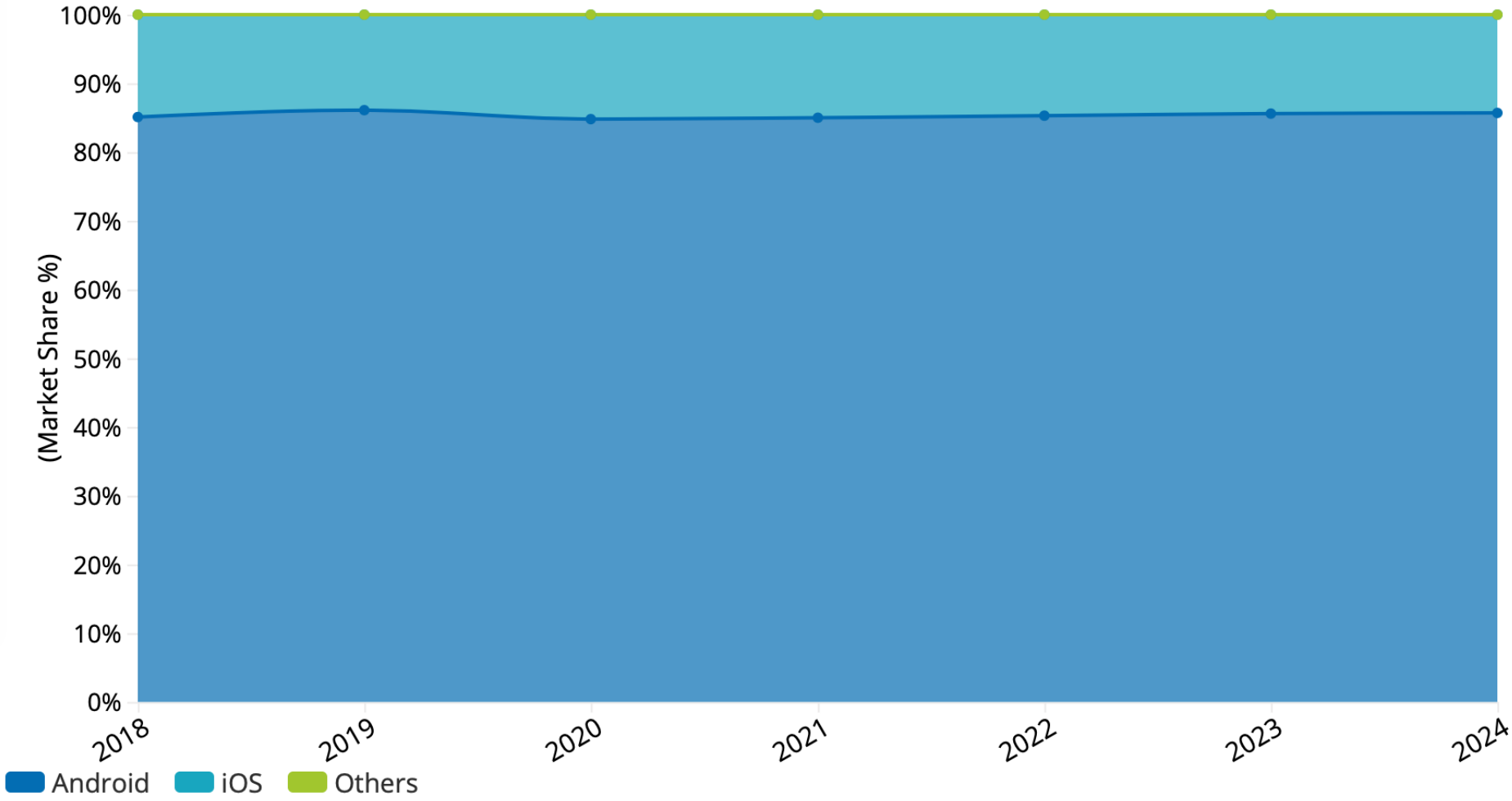
Source: IDC, Aug 2015

** Motorola figures have been captured under Lenovo.*


Worldwide Top 5 Smartphone Company Unit Market Share (%)
2018Q4
2019Q1
2019Q2
2019Q3
2019Q4
2020Q1
2020Q2
2020Q3
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
29%
Others
22.7%
Samsung
Huawei
13.1%
Xiaomi
Apple
8.9%
vivo

Quarter	2018Q4	2019Q1	2019Q2	2019Q3	2019Q4	2020Q1	2020Q2	2020Q3	
Samsung	18.8%	23.0%	23.0%	21.8%	18.8%	21.2%	19.5%	22.7%	
Huawei	16.2%	18.9%	17.7%	18.6%	15.2%	17.8%	20.2%	14.6%	
Xiaomi	6.7%	8.9%	9.7%	9.1%	8.9%	10.7%	10.3%	13.1%	
Apple	18.3%	11.8%	10.2%	13.0%	19.9%	13.3%	13.6%	11.8%	
vivo	6.9%	7.4%	8.5%	8.4%	7.7%	9.0%	8.6%	8.9%	
Others	33.0%	30.0%	30.8%	29.1%	29.5%	27.9%	27.9%	29.0%	
TOTAL	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**

Worldwide Smartphone Shipment OS Market Share Forecast



Enter series to show

Year	2018	2019	2020	2021	2022	2023	2024
Android	85.1%	86.1%	84.8%	85.0%	85.3%	85.6%	85.7%
iOS	14.9%	13.9%	15.2%	15.0%	14.7%	14.4%	14.3%
Others	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
TOTAL	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**	**100.0%**

Get More Data

Top 5 Smartphone Companies, Worldwide Shipments, Market Share, and Year-Over-Year Growth, Q4 2020 (shipments in millions of units)

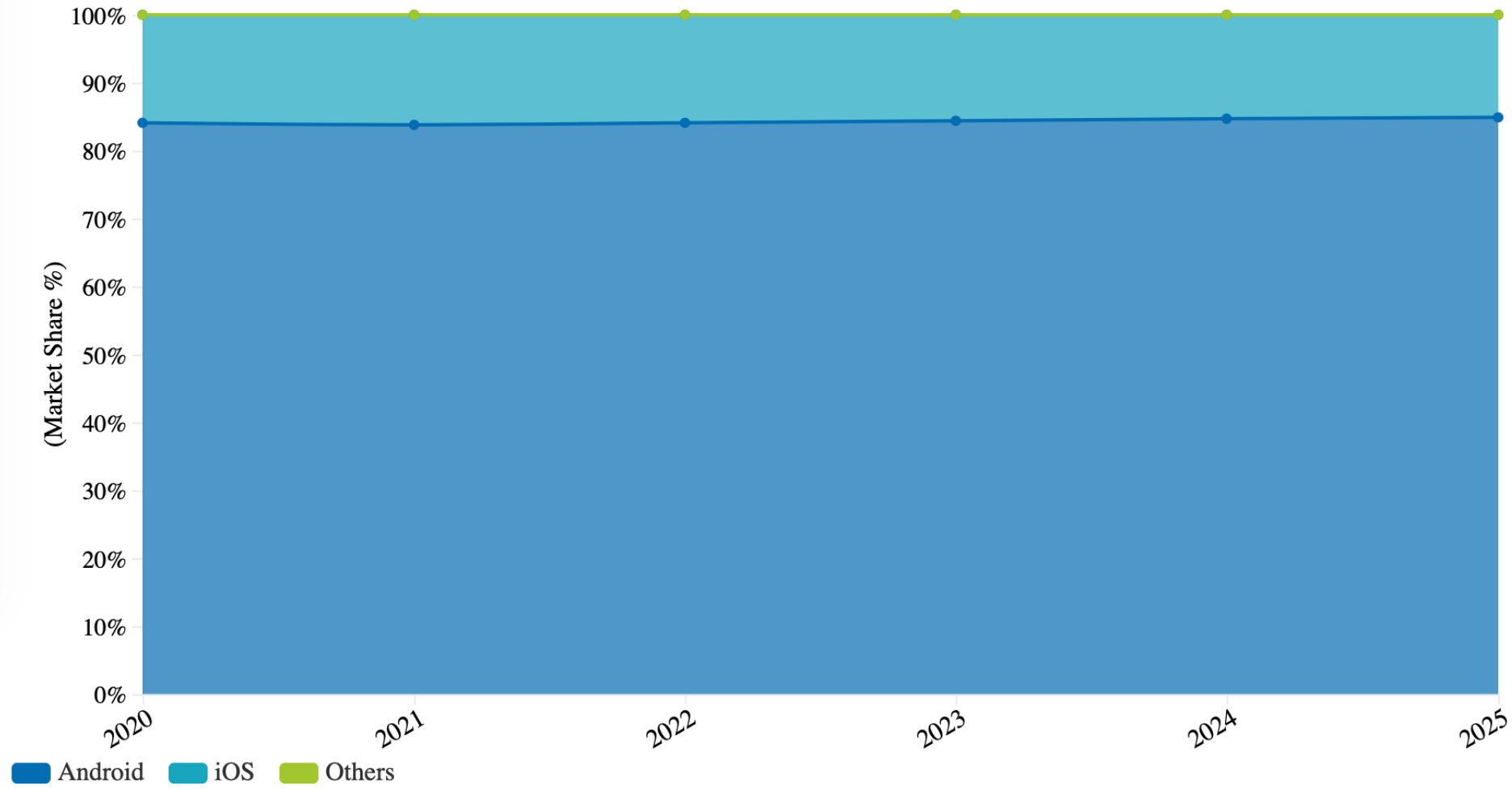
Company	2020Q4 Shipment Volumes	2020Q4 Market Share	2019Q4 Shipment Volumes	2019Q4 Market Share	Year-Over-Year Change
Apple	90.1	23.4%	73.8	19.9%	22.2%
Samsung	73.9	19.1%	69.5	18.8%	6.2%
Xiaomi	43.3	11.2%	32.8	8.9%	32.0%
OPPO	33.8	8.8%	30.6	8.3%	10.7%
Huawei	32.3	8.4%	56.2	15.2%	-42.4%
Others	112.4	29.1%	107.1	28.9%	5.0%
Total	**385.9**	**100.0%**	**369.9**	**100.0%**	**4.3%**

Source: IDC Quarterly Mobile Phone Tracker, January 27, 2021

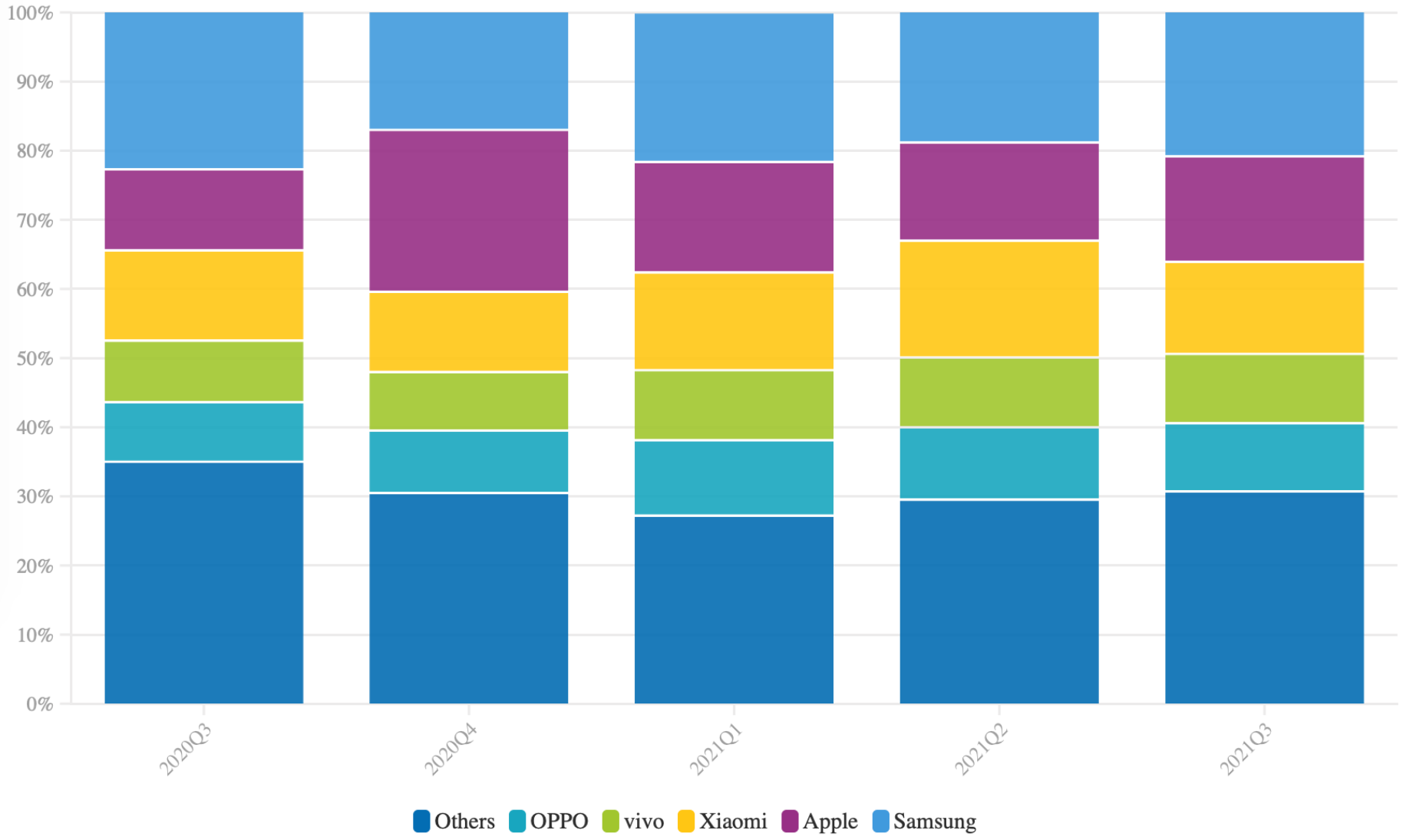
Company	2021 Shipments (in millions)	2021 Market Share	2020 Shipments (in millions)	2020 Market Share	Year-Over-Year Change
Samsung	272.0	20.1%	256.6	20.0%	**6.0%**
Apple	235.7	17.4%	203.4	15.9%	**15.9%**
Xiaomi	191.0	14.1%	147.8	11.5%	**29.3%**
Oppo	133.5	9.9%	111.2	8.7%	**20.1%**
vivo	128.3	9.5%	111.7	8.7%	**14.8%**
Others	394.3	29.1%	450.5	35.2%	**-12.5%**
Total	**1,354.8**	**100%**	**1,281.2**	**100%**	**5.7%**

The second half of 2021 failed to meet expectations, revealed Ryan Reith, group VP at IDC. The biggest issue in front of companies was the component shortages, and it will continue through the first half of 2022 as well.

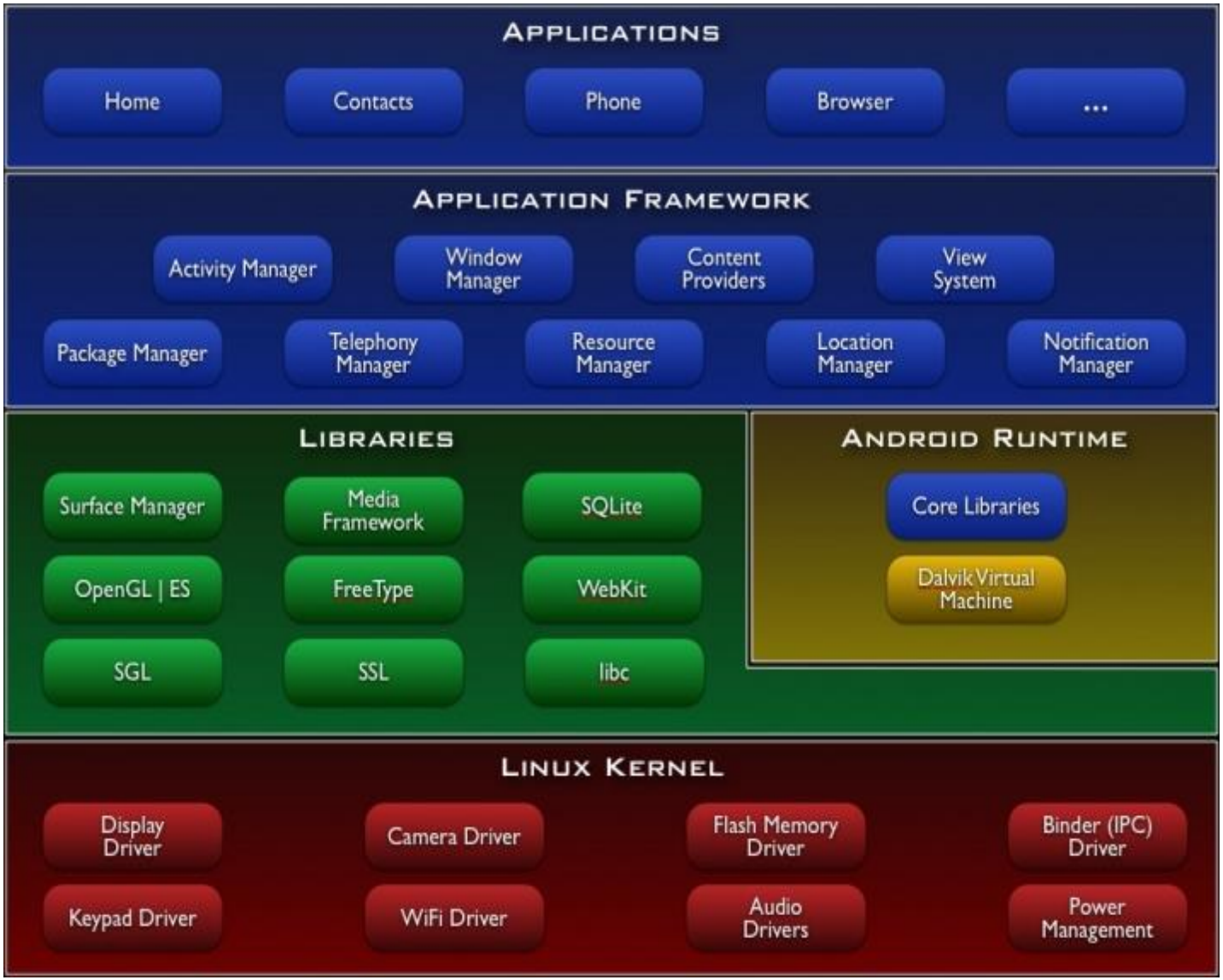
Worldwide Smartphone Shipment OS Market Share Forecast



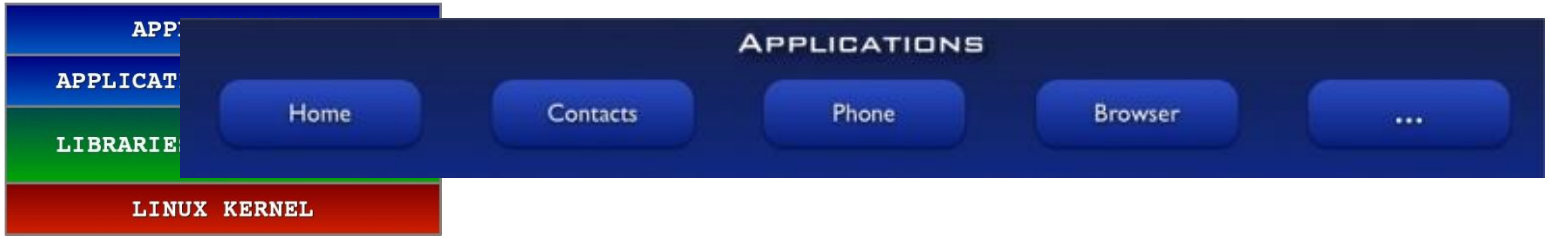
Enter series to show


Top 5 Smartphone Companies, 2021Q3
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
2020Q3
2020Q4
2021Q1
2021Q2
2021Q3
Others
OPPO
vivo
Xiaomi
Apple
Samsung

Android Architecture



Android S/W Stack - Application



- Android provides a set of core applications:
  - ✓ Email Client
  - ✓ SMS Program
  - ✓ Calendar
  - ✓ Maps
  - ✓ Browser
  - ✓ Contacts
  - ✓ Etc

- All applications are written using the Java language.

Android S/W Stack – App Framework

- Enabling and simplifying the reuse of components
  - ✓ Developers have full access to the same framework APIs used by the core applications.
  - ✓ Users are allowed to replace components.

Android S/W Stack – App Framework (Cont)

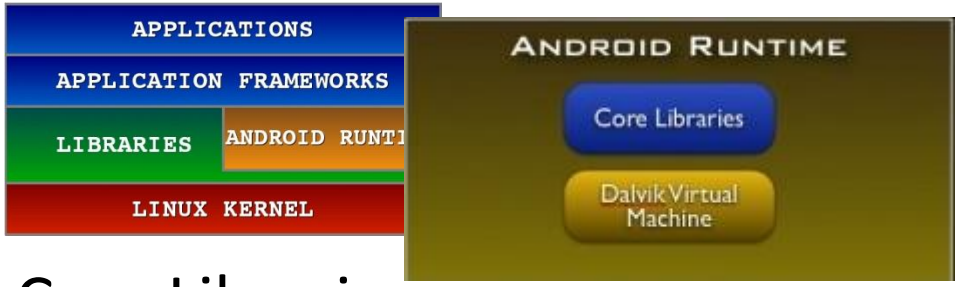
- Features

Feature	Role
View System	Used to build an application, including lists, grids, text boxes, buttons, and embedded web browser
Content Provider	Enabling applications to access data from other applications or to share their own data
Resource Manager	Providing access to non-code resources (localized strings, graphics, and layout files)
Notification Manager	Enabling all applications to display customer alerts in the status bar
Activity Manager	Managing the lifecycle of applications and providing a common navigation backstack

Android S/W Stack - Libraries

- Including a set of C/C++ libraries used by components of the Android system
- Exposed to developers through the Android application framework

Android S/W Stack - Runtime



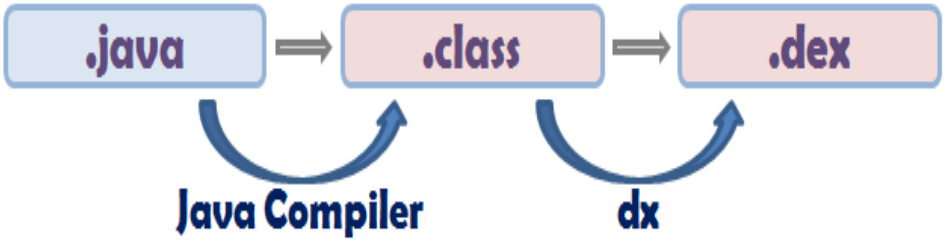
- Core Libraries
  - ✓ Providing most of the functionality available in the core libraries of the Java language
  - ✓ APIs
    - ➢ Data Structures
    - ➢ Utilities
    - ➢ File Access
    - ➢ Network Access
    - ➢ Graphics
    - ➢ Etc

Android S/W Stack – Runtime (Cont)

- Dalvik Virtual Machine
  - ✓ Providing environment on which every Android application runs
    - ➢ Each Android application runs in its own process, with its own instance of the Dalvik VM.
    - ➢ Dalvik has been written such that a device can run multiple VMs efficiently.
  - ✓ Register-based virtual machine

Android S/W Stack – Runtime (Cont)

- Dalvik Virtual Machine (Cont)
  - ✓Executing the Dalvik Executable (.dex) format
    - ➢.dex format is optimized for minimal memory footprint.
    - ➢Compilation



  - ✓Relying on
    - ➢Threading
    - ➢Low-level memory management

Android S/W Stack – Linux Kernel

- Relying on Linux Kernel 2.6 for core system services
  - ✓ Memory and Process Management
  - ✓ Network Stack
  - ✓ Driver Model
  - ✓ Security
- Providing an abstraction layer between the H/W and the rest of the S/W stack

Hardware-oriented Features

Feature	Description
Camera	A class that enables your application to interact with the camera to snap a photo, acquire images for a preview screen, and modify parameters used to govern how the camera operates.
Sensor	Class representing a sensor. Use getSensorList(int) to get the list of available Sensors.
SensorManager	A class that permits access to the sensors available within the Android platform.
SensorEventListener	An interface used for receiving notifications from the SensorManager when sensor values have changed. An application implements this interface to monitor one or more sensors available in the hardware.
SensorEvent	This class represents a sensor event and holds information such as the sensor type (e.g., accelerometer, orientation, etc.), the time-stamp, accuracy and of course the sensor's data.
MediaRecorder	A class, used to record media samples, that can be useful for recording audio activity within a specific location (such as a baby nursery). Audio clippings can also be analyzed for identification purposes in an access-control or security application. For example, it could be helpful to open the door to your time-share with your voice, rather than having to meet with the realtor to get a key.
GeomagneticField	This class is used to estimated estimate magnetic field at a given point on Earth, and in particular, to compute the magnetic declination from true north.
FaceDetector	A class that permits basic recognition of a person's face as contained in a bitmap. Using this as a device lock means no more passwords to remember — biometrics capability on a cell phone.

Sensor and SensorManager

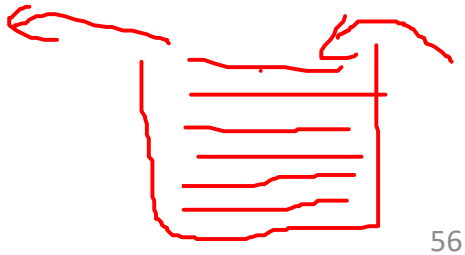
- Sensor type (Sensor class)
  - Orientation, accelerometer, light, magnetic field, proximity, temperature, etc.
- Sampling rate
  - Fastest, game, normal, user interface.
  - When an application requests a specific sampling rate, it is really only a hint, or suggestion, to the sensor subsystem. There is no guarantee of a particular rate being available.
- Accuracy
  - High, low, medium, unreliable.

Uygulama temelleri

- Java ile yazılır
  - Derlenince Android package .apk olur
  - Her uygulama kendi içinde çalışır
- Her uygulama bileşenlerden, manifest dosyasından ve kaynaklardan (resources) oluşur.
- Bileşenler (components)
  - Activities
  - Services
  - Content Providers
  - Broadcast Receivers

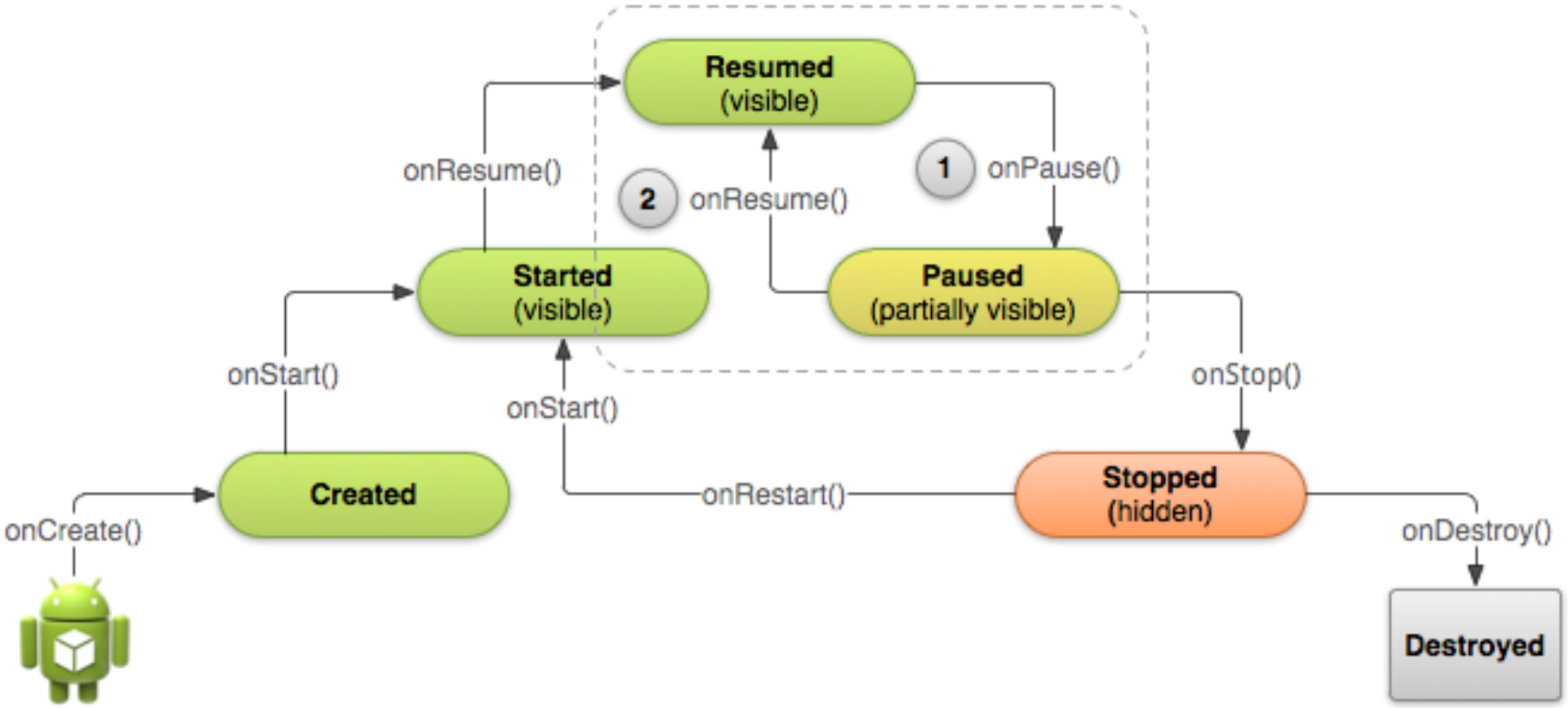
Activity ler

- Kullanıcı arabirimine sahip TEK ekranı temsil eder
  - O anda ekranda gözüken ara birimleri temsil eder
- Bir çok uygulama birden fazla aktivite içerebilir.
- Yeni bir aktivite başladığında, bir önceki aktivite back stack a atılır
  - Kullanıcı back tuşuna basında önceki aktivite öne çıkar



Activiteler

- Kullanıcı arabirimi XML ile veya JAVA içinde halledilebilir.
  - XML ile yapmak önerilir
- onStart(), onPause() etc.


Resumed
(visible)
onResume()
1
onPause()
2
onResume()
Started
(visible)
Paused
(partially visible)
onStart()
onStart()
onStop()
Created
onRestart()
Stopped
(hidden)
onCreate()
onDestroy()
Destroyed

Servisler

- Arkada uzun süreli çalışan işlemleri temsil eder
- Bir kullanıcı arabirimi yoktur
- Ağ uygulamaları, müzik çalma gibi işler için önerilir.
- Servisler kendisini çalıştıran aktivite den bağımsız olarak çalışırlar. Aktivite kapansa da Servis kapanmak zorunda değildir.
- Servisler diğer uygulamaların bileşenlerine bağlı olabilir.

Content Provider

- Verileri saklamak ve çağırmak ve tüm uygulamalar tarafından kullanılabilmesini sağlanmak için kullanır
- Default olarak farklı uygulamalar arasında bilgi paylaşımı yoktur. Bu yüzden uygulamalar arasında veri paylaşılmasını sağlayan tek yoldur.
- Bunu veri kümesini tekil olarak temsil eden açık bir URI'n kullanılması ile yapılır. Eğer başka uygulamaların erişmesini istediğim bilgilerim varsa bir CONTENT PROVİDER yapıp, diğer uygulamaların erişmesine sağlanır.
- Veriler veri tabanı modelindeki bir TABLO gibi gösterilir.
- Android bu tip şeyler için bir çok PROVİDER içeriri CONTACTS, MEDİA vb.

Broadcast Reciever

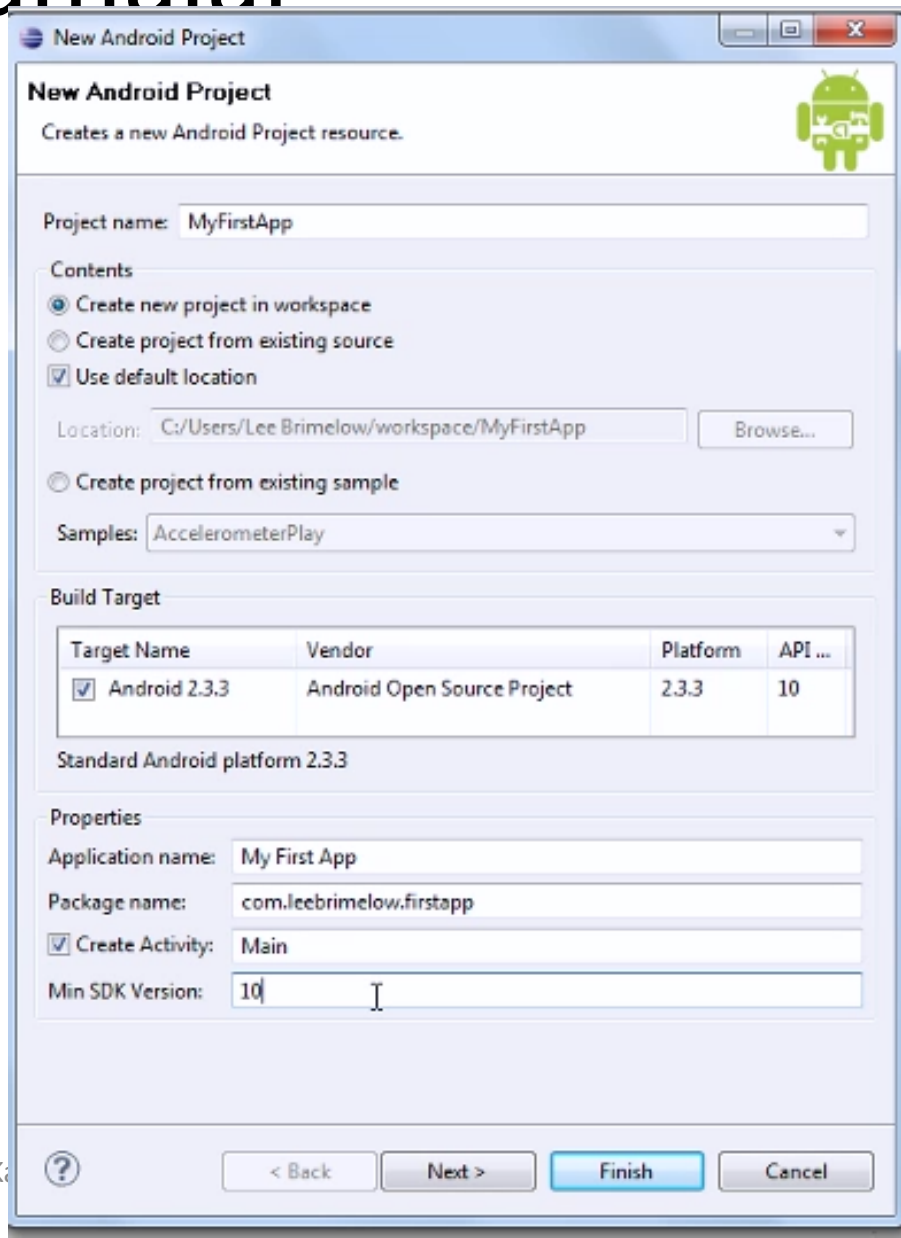
- Buda sistem de broadcast edilen tüm duyurulara cevap verebilen bir bileşendir.
  - Örnek, ekranın kapanması, bataryanın azalması, vb.
  - Uygulama bu mesajları alıp ona göre uygulama yapilecektir.
- Aynı zamanda kendi uygulamamızda kendi yayınlarını gönderebilir.
- Broadcast receiver bir görsel arabirim içermezler
- Ama Status çubuğu uyarıları üreterek kullanıcıya uyarı gönderebilir.

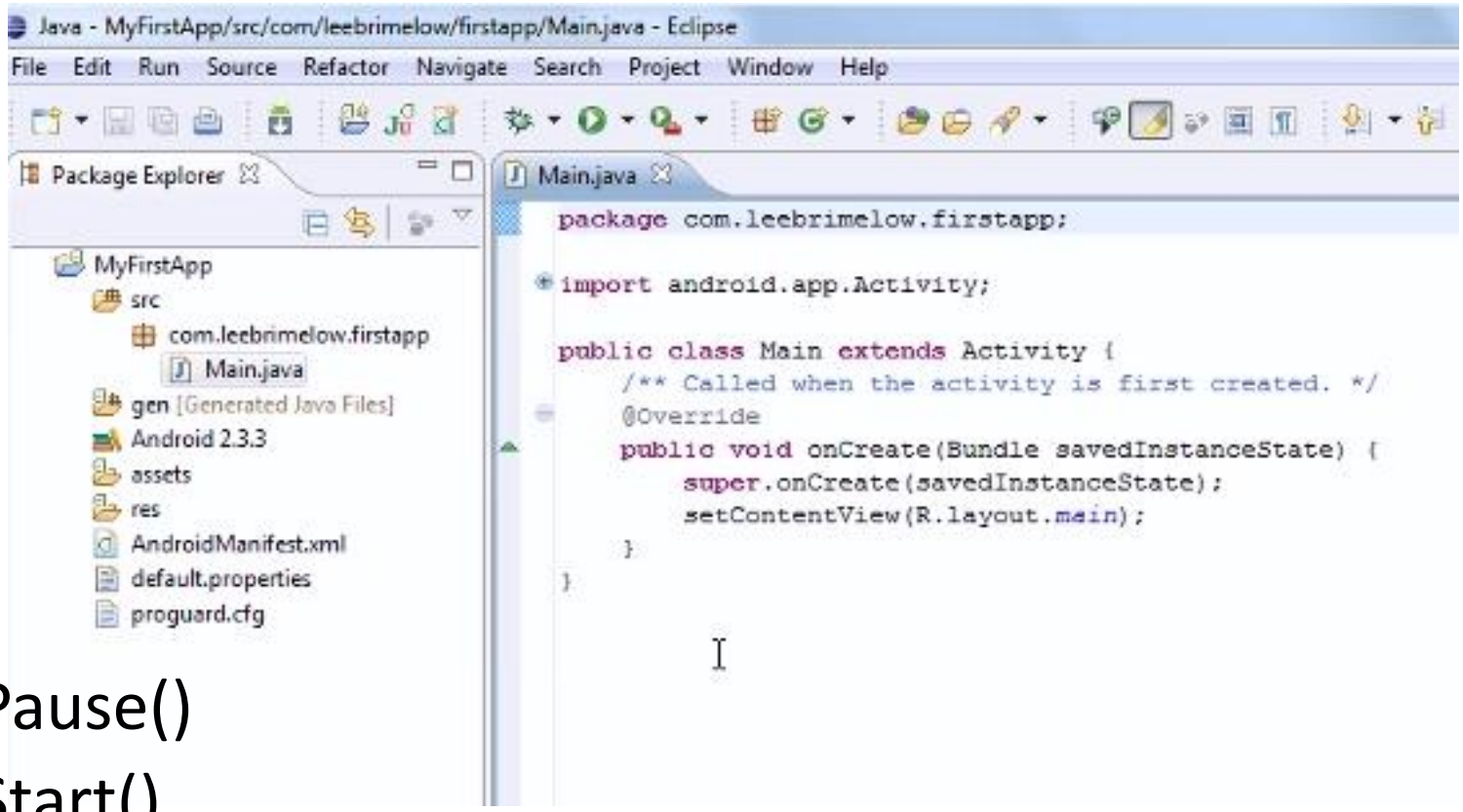
Android Manifest.xml

- Tüm uygulamalar AndroidManifest.xml adında uygulamanın kök dizininde bulunan bir doya içermelidir.
- Bu aynı zamanda kullanılacak olan Bileşenleri (acitivte,, service vb) içerir
- Ayrıca uygulamayı çalıştırmak için gerekli izinleri içerir.
  - Apk. Doyası kurulurken gerekli izinleri sorar…
- Minimum Android API level gösterir

İlk uygulamalar

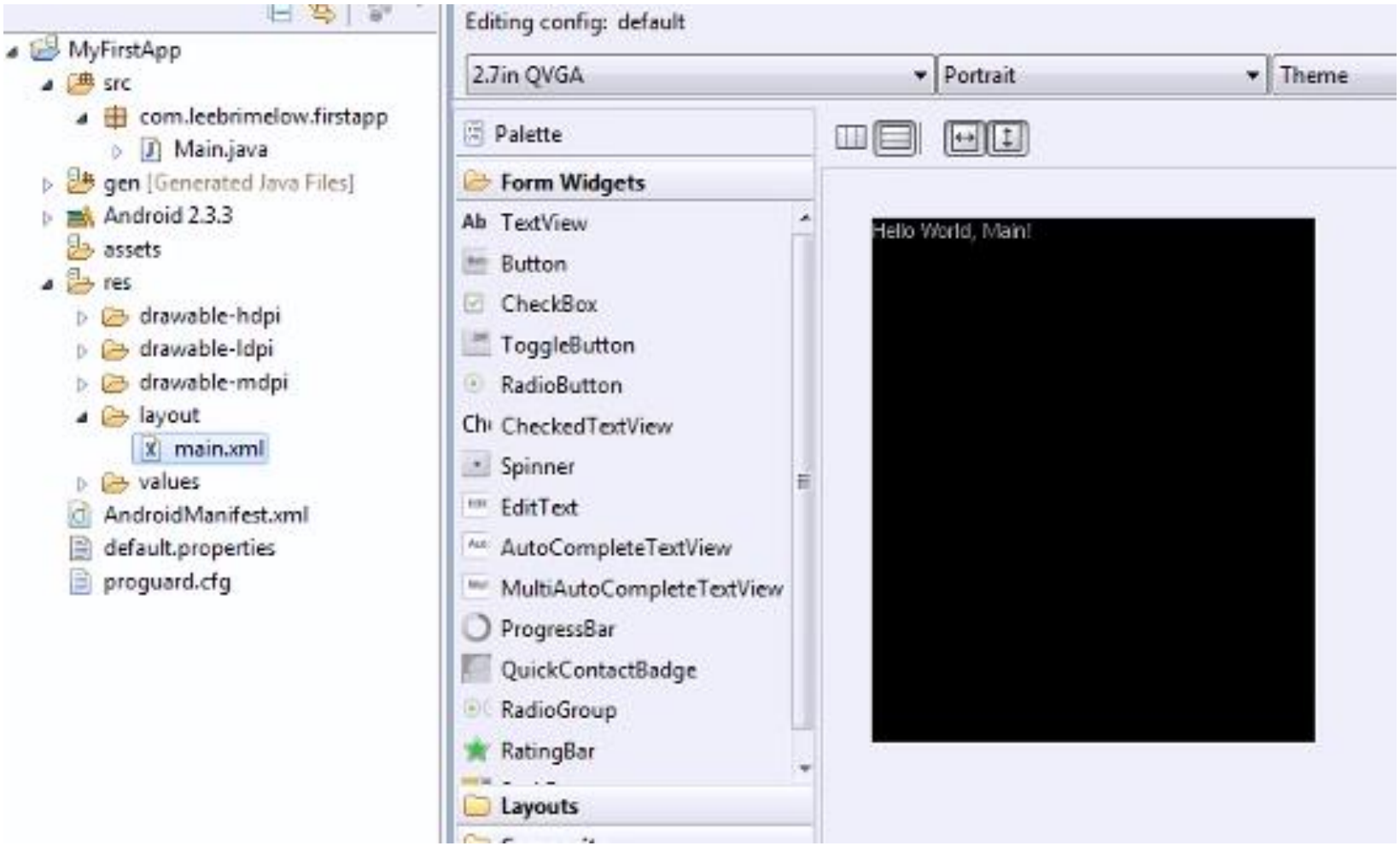
- Proje adı
- Application name
- Package Name reverse
- Create activity
- Min SDK version
- finish





- OnPause()
- OnStart()
  - Başlangıçta bir hata verebiliyor bir zaman vermek gerekiyor.
- setContentView(R.layout.main

Main.xml

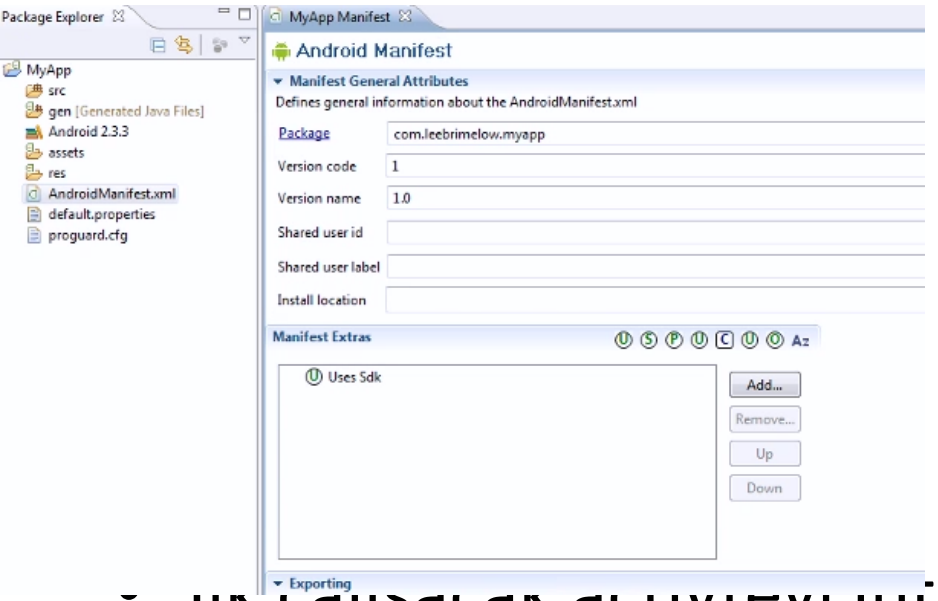

MyFirstApp
src
com.leebrimelow.firstapp
Main.java
gen [Generated Java Files]
Android 2.3.3
assets
res
drawable-hdpi
drawable-ldpi
drawable-mdpi
layout
main.xml
values
AndroidManifest.xml
default.properties
proguard.cfg
Editing config: default
2.7in QVGA
Portrait
Theme
Palette
Form Widgets
TextView
Button
CheckBox
ToggleButton
RadioButton
CheckedTextView
Spinner
EditText
AutoCompleteTextView
MultiAutoCompleteTextView
ProgressBar
QuickContactBadge
RadioGroup
RatingBar
Layouts
Hello World, Main!

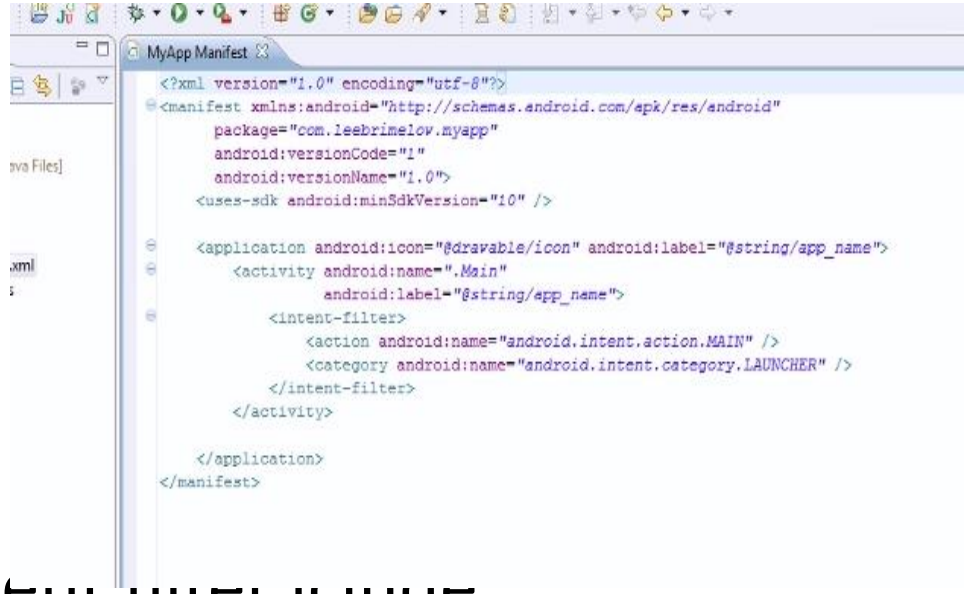
Folders

- Gen
  - OYNAMAYIN otomatik üretilir
- Android x.x.x gerçek SDK
  - Burası ile de OYNAMAYIN
- Res
  - Resimler
  - Layoutlar
  - valueler

AndroidManifest.xml

- Uygulamamızın Android sistemine anlatan dosyadır.





- İlk çalışacak aktiviteyi intent filter içinde gösterebiliriz
- Tüm activiteler burada tanımlanmış olmalı

Activity Temelleri

- LearnActivity Projesi
  - Amaç iki aktivite ile çalışmak
- Key Steps
  - XML layout dosyası ekle (second)
    - TextView ekle ve ikinci (metnini değiştir)
    - Buton ekle (Adını değiştir)
  - İkinci bir sınıf ekle Activity sınıfından olsun
    - **protected void onCreate(Bundle savedInstanceState) ekle**

- İlk aktiviteye Buton Ekle Oncreate kısmında

- Button b= (Button) findViewById(R.id.*btn1);*
- b.setOnClickListener(**new OnClickListener() {**
- **public void onClick(View v) {**
- // **TODO Auto-generated method stub**
- startActivity(**new Intent(ilk.this, ikinci.class));**
- }
- });
  - Manifest dosyasına ikinci aktiveteyi ekle
  - İkinci aktiviteye onCreate metoduna
    - setContentView(R.layout.*second); XML layout ekle*

İntent

- Bir önceki örnekte görüldüğü gibi kullanılan intent bir şeyi aktif hale getirmek için kullandık
- İki tip intent vardır
  - Explicit
    - Özel bir, belirli bir activiteyi çağırmak için kullanırız.
      - Hey işte «bu» aktiviteyi çalıştır.
  - Implicit
    - Ben bir eylem yapmak istiyorum diye Android sitemine duyurulur ve Bu eylemi yapabilecek tüm uygulamalar arasında seçim yapılır. Çalıştıran ne çalıştıracağını bilmez

İki activite arasında bilgi Gönderme

- Birinci Aktivitede
  - **final EditText et= (EditText) findViewById(R.id.*editText1);***
  - Intent myIntent = **new Intent(AndTest04Activity.this, Second.class);**
  - myIntent.putExtra("TheText", et.getText().toString());
  - startActivity(myIntent);
- İkinci Aktivitede
  - TextView tv = (TextView) findViewById(R.id.*textView1);*
  - tv.setText(getIntent().getExtras().getString("TheText"));

İmplicit Intents

- MANIFEST File
- <intent-filter>
- <action android:name="*android.intent.action.SEND*" />
- <category android:name="*android.intent.category.DEFAULT*" />
- <data android:mimeType="*image/**"></data>
- </intent-filter>

- **try{**
- ImageView iv= (ImageView) findViewById(R.id.*imageView1);*
- iv.setImageURI((Uri) getIntent().getExtras().get(Intent.*EXTRA_STREAM));*
- }
- **catch(Exception ex)**
- { }

Resources

- Res dizininde saklanır
  - Layout UI için
  - Values, strings.xml
    - Globalization, uluslararsılaştırma
- Developer.android.com/guide/index.html
  - Application Resources/ Providing Resource
- Asset kalsörü
  - Bir ID üretilmez. Res dizinine koyulması tavsiye edilir.
- Projeye eklenmesi
  - Sürükle bırak
  - Hdpi, mdip, ldip Sistem otomatik olarak gerekli olanı alacak.

Gen Dizini

- R sınıfı ile oynamaz, otomatik üretilen bir sınıftır.
  - İçinde kaynakların bir çeşit adresleri var

```
/* AUTO-GENERATED FILE.  DO NOT MODIFY.
 *
 */

package com.leebrimelow.resources;

public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int amsterdam=0x7f020000;
        public static final int icon=0x7f020001;
        public static final int lima=0x7f020002;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class string {
        public static final int app_name=0x7f040001;
        public static final int hello=0x7f040000;
    }
}
```

Resource kullanmak

- ImageView
  - Src kısmından drawable dan gelir.
  - XML değişir.
    - @drawable/resim
- Java kodu ile yapılması
  - ImageView iv= (ImageView) findViewByID(R.id.imageview1)
  - iv.setImageResource(R.id.drawable.resim)
- Id yi elle vermek
  - Android:id=«@+id/BileşenAdı»

Permissions/İzinler

- Apk kurulurken bizden bazı izinler ister
  - İnternete ulaşma, kamera kullanma , contactlara erişme vb.
- Örnek olarak Wifi açıkmı bu bir servistir
  - ConnectivityManager conman= (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE)
  - conman.getNetworkInfo(ConnectivityManager .Type_WIFI).isConnectedOrConnecting();

Manifest e Permission Koymak

- Log Cat e gidilince
  - Security istisnası oluşur
- Manifest.xml
  - Add Uses Permission
    - android.permission.ACCESS_NETWORK_STATE

```
<uses-sdk android:minSdkVersion="10" />

<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"></uses-permission>

<application android:icon="@drawable/icon" android:label="@string/app_name">
    <activity android:name=".Main"
```

Main.Xml unit &d layout

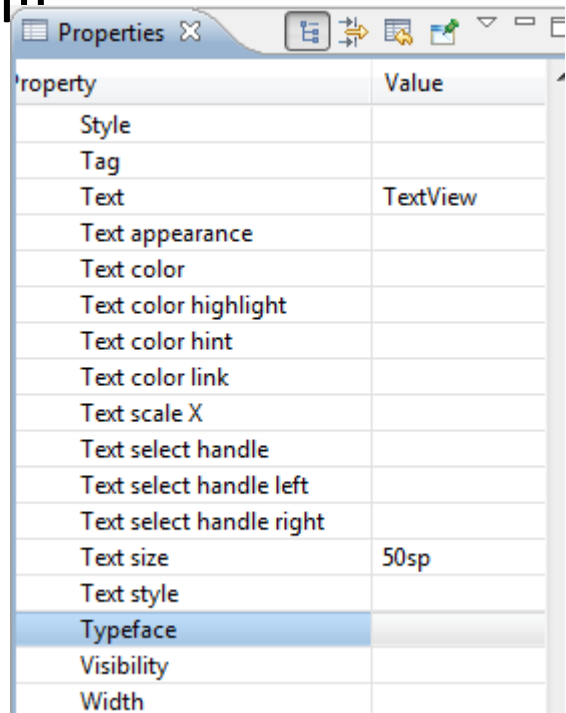
- İki görünümü var
  - Graphical Layout
  - Main.xml
    - LinearLayout
    - Her bir layout ın mutlaka layout_width ve layout_height iolmalı
      - fill_parent= tamamını kaplayacak
      - wrap_content= içerik kadar büyük olacak
- Bir buton eklenirse bu en ve boy özellikleri incelenebilir.
  - Properties view
    - Window/Show view/ other
      - General - properties

Properties

- Toggle fill, toggle height
  - Bazen properties da değeri hemen güncellenmez
  - Bu durumda, nesne üzerinde sağ tıkla, Show in – properties seç
    - Layout height/width
      - Match_parent (gingerbread) fill_parent, wrap_content
      - 200px (string olmalı) tavsiye edilmez.
      - 200dip device indipendent pixel dp

Text Size?

- Text Size =50dp device indipendent pixel
  - User preferences dikkate alınmıştır. Belki görme bozukluğu olan birisi metinleri büyük göstermek istiyordur.
  - Bu durumda Scale pixel sp kullanılabilir
- Text Size=50sp



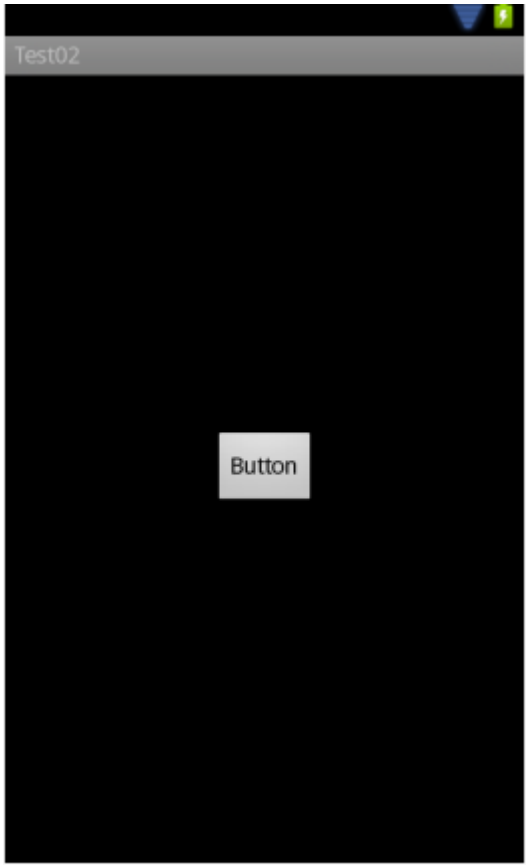
Layouts

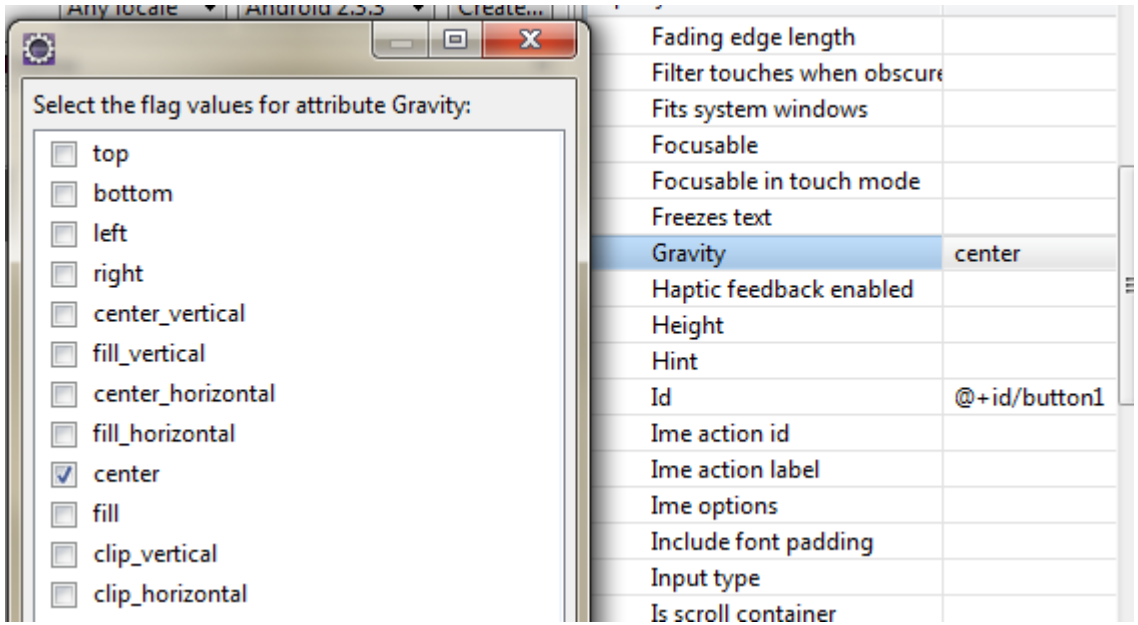
- Mutlaka bir layout olmalı
  - Linear Layout
  - Horizantal Layout
    - Wrap_content
  - RelativeLayout
    - Diğer itemlara bağlı olarak ayarlayacak
    - Hizalamaları Gravity ile yaparız

*main.xml strings.xml

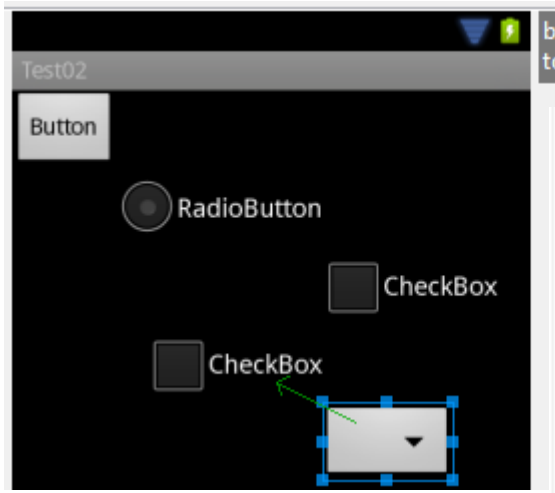
```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

</RelativeLayout>
```

Gravity






```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:gravity="center" >


    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:text="Button" />

</RelativeLayout>
```

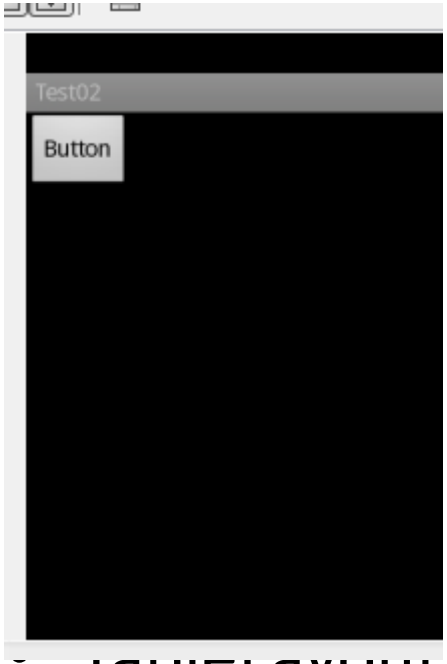



```
    <RadioButton
        android:id="@+id/radioButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/button1"
        android:layout_toRightOf="@+id/button1"
        android:text="RadioButton" />

    <CheckBox
        android:id="@+id/checkBox1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/radioButton1"
        android:layout_toRightOf="@+id/radioButton1"
        android:text="CheckBox" />

    <CheckBox
        android:id="@+id/checkBox2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignRight="@+id/radioButton1"
        android:layout_below="@+id/checkBox1"
        android:text="CheckBox" />

    <Spinner
        android:id="@+id/spinner1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/checkBox2"
        android:layout_toRightOf="@+id/checkBox2" />


</RelativeLayout>
```

Frame Layout

- Tüm neslere birbirleri üzerine gelir.
  - Her bir item üzerinden gravity değerlerini ayarlanabilir.
  - Bu sitem bir çok nesnenin ortalanması ve gerekli oldukça görünür yapılması anlamında kullanılabilir.

TableLayout




```
<?xml version="1.0" encoding="utf-8"?>
<TableLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TableRow
        android:id="@+id/tableRow1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" >

        <Button
            android:id="@+id/button1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Button" />

    </TableRow>
```

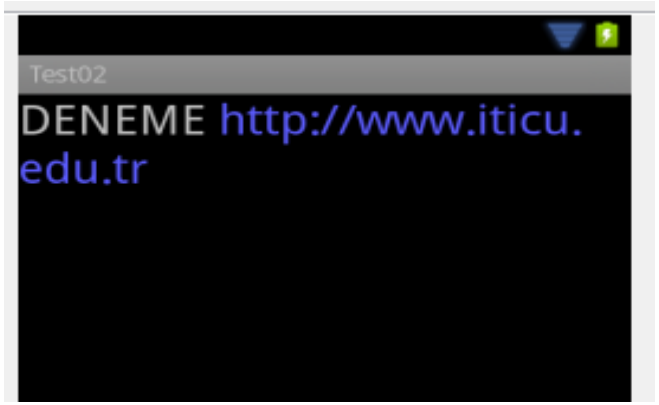
- TableLayout > TableRow eklenip içine elemanlar eklenir
- Form işlemleri yapılırken faydalı olacaktır.
- Layoutları olabildiğince az çeşit olacak şekilde kullanmak gereklidir.

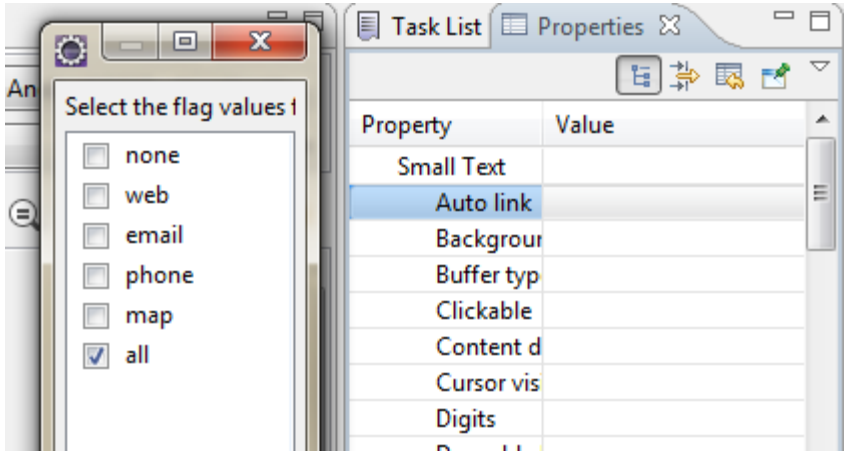
Text

- Properties
  - Text @strings/hello
    - Direk metin yazılabilir
  - TypeFace
  - TextSize
    - Pixel değerleri yerine scale indipendent
  - Text Color hex değer alır (RGB)
    - #FFFFFF (beyaz) #FF0000 tam kırmızı

Text

- Text özelliğine bir link verilmek istense
  - AutoLink ile içinde yazılı olan link biçimini kullanabiliriz
  - Textview aslında değişik çeşit metinler gösterebilmek için vardır.





Edit Text

- Text
  - Multiline default
  - Deafult olarak merkeze hizalanır
    - Gravity ile değiştirebiliriz
- Input type
  - Değişik tipte kullanıcı girişlerine izin verir keyboard
    - Password, number, none,
- Reading

  -Main.java

  EditText et = (EditText) findViewById(R.id.***editText1***);

  et.getText().toString()

  getText() metodu edit edilebilen bir alan getirir.

AutoCompleteTextView

- Text field
  - Bir dizi string değeri vererek otomatik doldurulmasını sağlayabiliriz
  - İlçeler
- MultiAutoCompleteTextview
  - Bir çok kelimeden word base kelime temelli olarak çalışır.

Buttons

- Button
- Toggle Button
  - On/Off
- Image button
  - Src/ Image
- Radio button
    - <RadioGroup
    - android:layout_width=*"fill_parent"*
    - android:layout_height=*"wrap_content"*
    - android:orientation=*"vertical"*>
- Checkbox button

OnClick Event REspond

```
8  public class Main extends Activity implements OnClickListener {
```

```
14          setContentView(R.layout.main);
15
16          Button b = (Button) findViewById(R.id.button1);
17          b.setOnClickListener(this);
18      }
19
20      @Override
21      public void onClick(View v) {
22          // TODO Auto-generated method stub
23
24      }
25  }
```

```
public void onClick(View v) {
    if(v.getId() == R.id.imageButton1) {
        Log.d("LEE", "Image button was clicked");
    }
```

List

- Res
  - Values
    - Country.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="countries">
        <item name="usa">United States</item>
        <item name="brazil">Brazil</item>
        <item name="russia">Russia</item>
        <item name="egypt">Egypt</item>
        <item name="japan">Japan</item>
    </string-array>
</resources>
```

- Eğer List elemanı kullanılacaksa
  - Activity yerine ListActivity kullanılabilir.

ListView Control

- Layout
  - Composite
  - ListView
    - Fill entire parent
    - ListActivitesinin bakabilmesi için özel bir id vermek gerekir «@android:id/list»
- JAVA
  - Bir adaptör e ihtiyacımız var

```
setListAdapter(new ArrayAdapter<String>(this,
        android.R.layout.simple_list_item_1,
        getResources().getStringArray(R.array.countries)));
```

Toast Message

```
Button b = (Button) findViewById(R.id.button1);
b.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View v) {
        Toast toast = Toast.makeText(Main.this, "This is some toast", 5000);
        toast.setGravity(Gravity.CENTER, 0, 0);
        toast.show();
    }
```

List view TIKLAMA

```
    }

    @Override
    protected void onListItemClick(ListView l, View v, int position, long id) {
        super.onListItemClick(l, v, position, id);

         Toast.makeText(this,(String) l.getItemAtPosition(position),Toast.LENGTH_LONG).show();

    }
```

Status Notification

- Notification service manager dan bir örnek alınır

```
Intent myInt= new Intent( packageContext: this,MainActivity.class);
PendingIntent pending = PendingIntent.getActivity( context: this,
        requestCode: 0,
        myInt,
        flags: 0);

//Notification.Action myAction=
Notification n  = new Notification.Builder( context: this)
        .setContentTitle("New mail from " + "test@gmail.com")
        .setContentText("Subject")
        .setSmallIcon(android.R.drawable.btn_plus)
        .setContentIntent(pending)
        .setAutoCancel(true)
        .addAction(android.R.drawable.ic_menu_compass,
                title: "Call", pending)
        .addAction(android.R.drawable.ic_dialog_email,
                title: "More", pending)
        .addAction(android.R.drawable.ic_menu_camera,
                title: "And more", pending).build();

NotificationManager nm = (NotificationManager)
        getSystemService(Context.NOTIFICATION_SERVICE);


nm.notify( id: 0,n);
```

BEEP

- Res klasörü içine
  - Raw klasörü açılır.ve içine audio dosyaları konulabilir.

Temel SQLite

- Bir çok yapısal veri kullanırız.
  - İlk önce bir örnek açılır (

```
SQLiteDatabase db = openOrCreateDatabase("MyDB", MODE_PRIVATE, null);
```

  - Ad
  - Mode : kim erişlebilir/ sadece bu program mı diğer prpgramlarda erişebilir
  - Null

SQLLite a veri yazıp okuma

- SQLiteDatabase db= openOrCreateDatabase(**"Test"**,***MODE_PRIVATE***,**null**);

```
// db.execSQL("Drop Table MyTable");
 db.execSQL("Create Table If not exists " +
        "MyTable(" +
        "Id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,"
+
        " Lastname varchar NOT NULL, " +
        " FirstName CHAR(50) NOT NULL,  " +
        " Age int(3) NOT NULL);");

 db.execSQL("insert  into MyTable(Lastname, FirstName, Age )
Values('kasapbasi', 'mustafa', 61);");

 db.close();
```

Veri Okuma

- Data >Data (packadge adı) içinde oluşturulan DB durur
  - Bir cursor nesnesi döndüren rowquery metodu çalıştırılır.
    - Cursor nesnesi liste/tablo halindeki veriler üzerinde

```
SQLiteDatabase db = openOrCreateDatabase("MyDB", MODE_PRIVATE, null);
Cursor c = db.rawQuery("SELECT * FROM MyTable", null);
```

ilk parametre SQL sorgu cümlecigi,

Cursor kullanmak

- **boolean requery()**
- moveToFirst(); move;movetonext
- **int getCount()**
- **String[] getColumnNames()**
- **String getColumnName(int columnIndex)**
- **int getColumnIndex (String columnName)**
- **int getColumnCount()**
- **getVeriTipi metotpları**
  - **getInt (kolon indeksi)**
  - **getString (kolon indeksi)**
  - **getBlob (kolon indeksi)**
  - **getLong (kolon indeksi)**

Tipik Kod

```
if (c != null ) {
    if  (c.moveToFirst()) {
        do {
            firstName = c.getString(c.getColumnIndex("FirstName"));
            int age = c.getInt(c.getColumnIndex("Age"));
            //results.add("Name: " + firstName + ",Age: " + age);
        }while (c.moveToNext());
    }
}
```

Network Erişimi

- Manifest dosyası
  - Uses Permission
    - Android.permission.INTERNET
- Layout

Java

- Bileşenlere referans ekle

```
final EditText et = (EditText) findViewById(R.id.editText1);
final Button b = (Button) findViewById(R.id.button1);
final TextView tv = (TextView) findViewById(R.id.textView1);
```

- OnClick

```
b.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View v) {
        try {

        } catch(Exception e) {

        }
    }
```

- URL nesnesi
  - URL url = new URL(string);
- URL Connection
  - URLConnection con= url.openConnection();
- Okumak için
  - BufferedReader reader= new BufferedReader(new InputStreamReader(con.getInputStream()));
- Satır satır okuyacağz

```
String line = "";
while((line = reader.readLine()) != null) {

}
```

Shared Preference

- Kullanıcı ayarları
- Kalması istenen bilgiler
- SHARED Preferences API

Örnek Proje

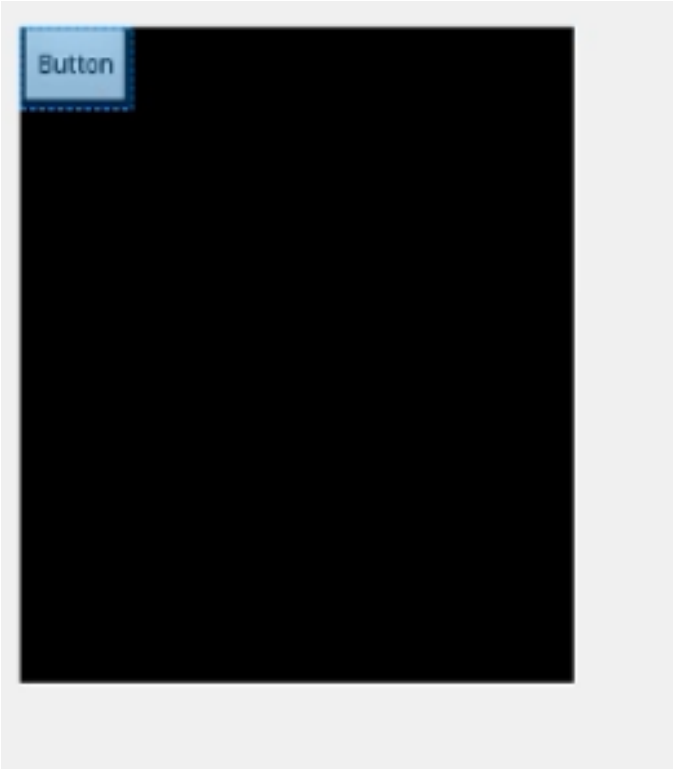
Bir EditText içindeki bilgi program kapatıldıktan sonra, tekrar açılınca gelecek

SharedPrefefences

```
public class Main extends Activity {

    private EditText et;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        et = (EditText) findViewById(R.id.editText1);

        SharedPreferences settings = getSharedPreferences("MYPREFS", 0);
        et.setText(settings.getString("tvalue", ""));
    }

    @Override
    protected void onStop() {
        super.onStop();
        SharedPreferences settings = getSharedPreferences("MYPREFS", 0);
        SharedPreferences.Editor editor = settings.edit();
        editor.putString("tvalue", et.getText().toString());
        editor.commit();
    }
```

Preference Screens

- Düğmeye basıldığında ara birim preference activitesi çıkacak ve oradan istediği ayar yapılabileicek



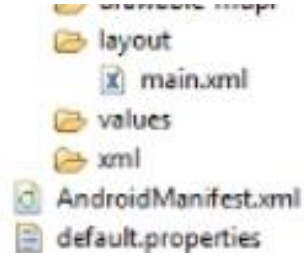
Preference Screen

- Yeni bir aktivite ekliyoruz
  - Üst sınıf olarak PrefereneActivity olacak
- Yeni layout buda özel bir XML olacak
  - XML eklede preference seçilir.

```
<?xml version="1.0" encoding="utf-8"?>
<PreferenceScreen
  xmlns:android="http://schemas.android.com/apk/res/android">

  <CheckBoxPreference android:key="first"
    android:title="First Option"
    android:summary="This is the first option." />

  <CheckBoxPreference android:key="second"
    android:title="Second Option"
    android:summary="This is the second option." />

</PreferenceScreen>
```

Preference Screen

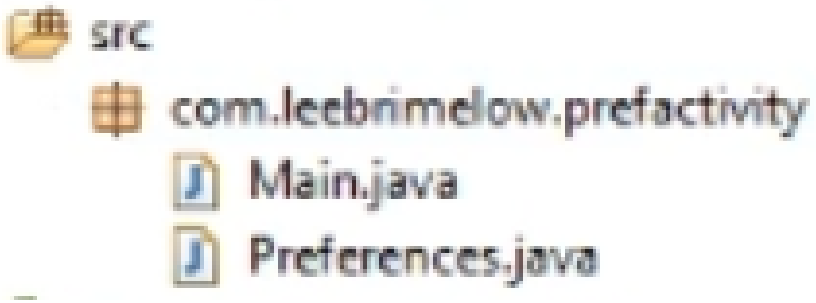
- Preference XML ini yüklemek için normal Activitelerde olduğu gibi
  - setContetView(R.layout.main) KULLANILMAZ
- Onun yerine bir metot kullanılır.




```
11          super.onCreate(savedInstanceState);
12          addPreferencesFromResource(R.xml.prefs);
13      }
14
15  }
```

Preference Screen

- Manifest Dosyasında activiteyi tanımlamak gerekir.
- Manifest dosyasına sadece name öz. Koymak yeterli oda JAVA sınıfının adı olmalı




```
<activity android:name=".Preferences" />
```

```
    Button b = (Button) findViewById(R.id.button1);
    b.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View v) {
            Intent intent = new Intent(Main.this, Preferences.class);
            startActivity(intent);
        }
    });
}
```

- Kendisi gerekli ayarları tutacaktır ve istenildiğinde okunaması sağlanacak

Shared Preferensları okumak

```
SharedPreferences settings = PreferenceManager.getDefaultSharedPreferenc
boolean first = settings.getBoolean("first", false);
```

Option Menus

- XML ile veya JAVA kod ile oluşturulur

mymenu.xml

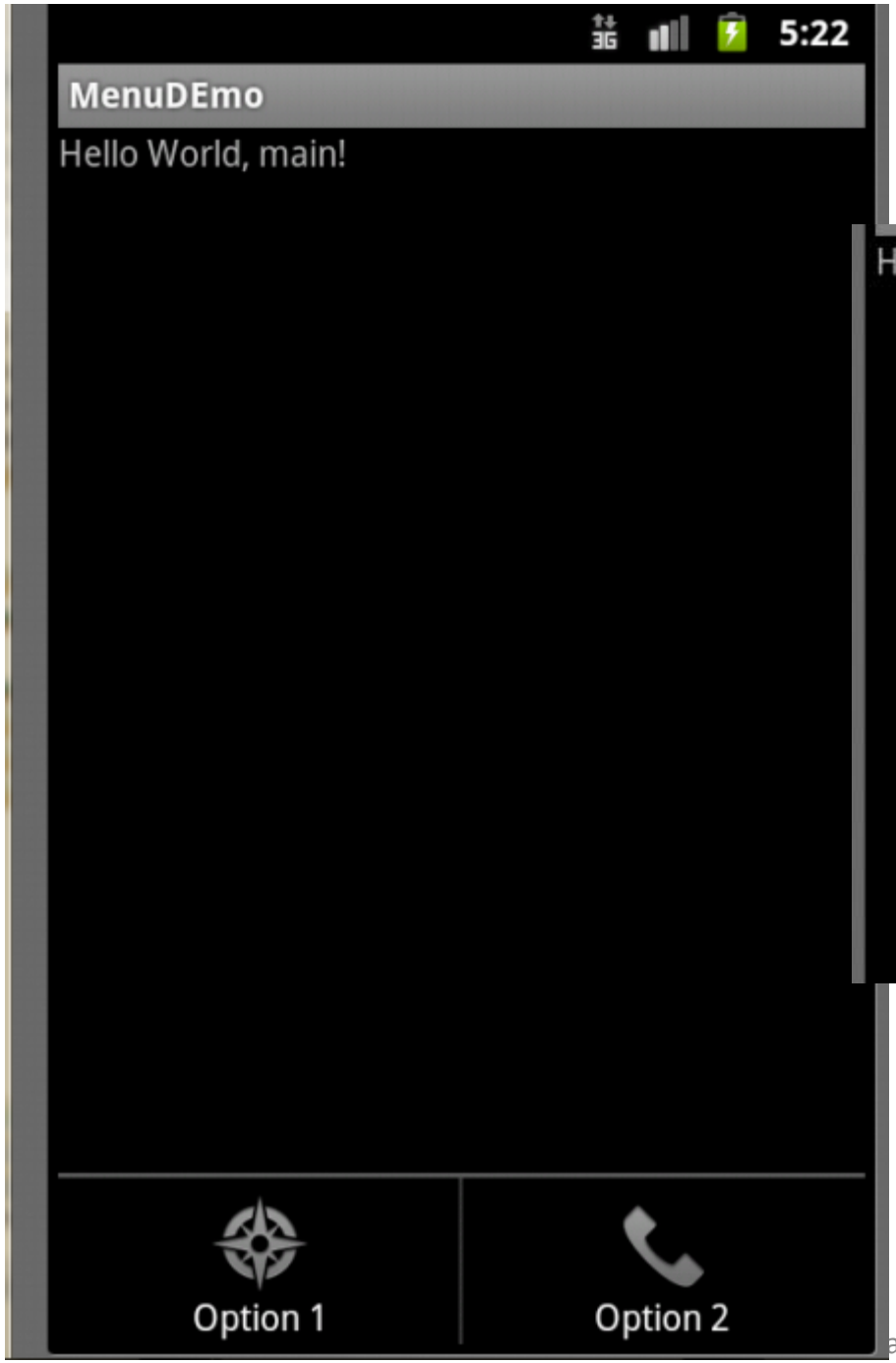
```
1 <?xml version="1.0" encoding="utf-8"?>
2 <menu xmlns:android="http://schemas.android.com/apk/res/android" >
3
4 <item android:id="@+id/item1" android:title="Option 1"
5     android:icon="@android:drawable/ic_menu_compass"></item>
6 </menu>
```

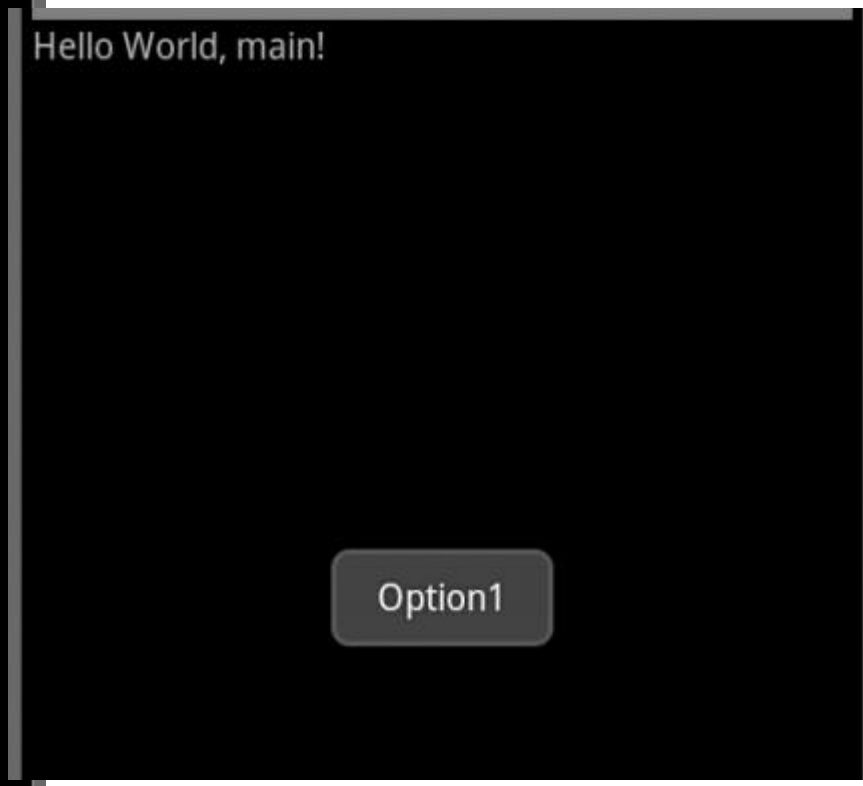
- Android iconları nın yeri
  - Android.jar
    - Resources
      - Drawable
      - ic ile başlayanlar

Option menu 2

```
import android.app.Activity;

public class main extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // TODO Auto-generated method stub
     MenuInflater infaltor = getMenuInflater();
     infaltor.inflate(R.menu.mymenu, menu);
     return true;


}
```

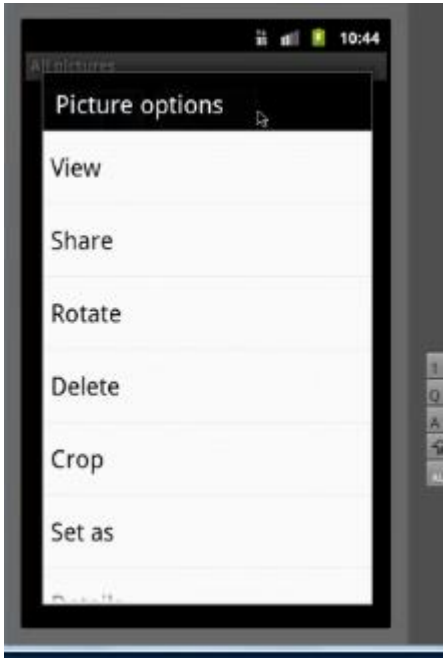
```
}
@Override
public boolean onOptionsItemSelected(MenuItem item) {

    if(item.getItemId()==R.id.item1)
    {
        Toast ts = Toast.makeText(this, "Option1", 5000);
        ts.setGravity(Gravity.CENTER, 0, 0);
        ts.show();

    }
    else if(item.getItemId()==R.id.item2)
    {
        Toast ts = Toast.makeText(this, "Option2", 5000);
        ts.setGravity(Gravity.CENTER, 0, 0);
        ts.show();

    }

    // TODO Auto-generated method stub
```


5:22
MenuDEmo
Hello World, main!
Option 1
Option 2


Hello World, main!
Option1

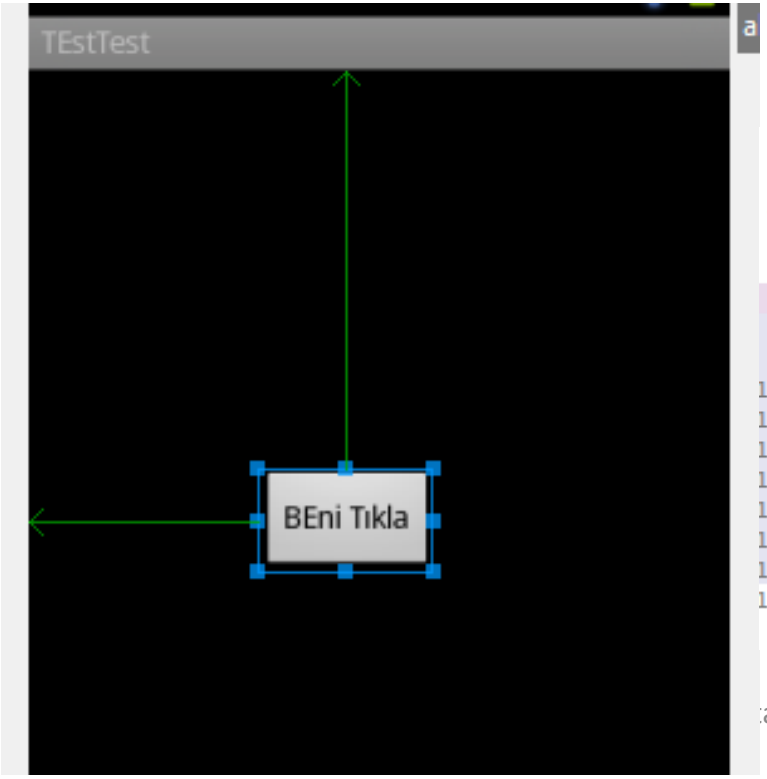
Context menu

- Bir nesneye uzun süre basıldığında ortaya çıkan menü
  - Galery resimler uzun süre basma



Context Menu Örneği

- Layout
  - Düğmeye uzun süre basıldığında Context menu oluşacak




```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/r
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentTop="true"
        android:layout_marginLeft="105dp"
        android:layout_marginTop="182dp"
        android:text="BEni Tıkla" />

</RelativeLayout>
```

Context Menu

- Menu XML oluşturma

main.xml | contextmenu.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:android="http://schemas.android.com/apk/res/android" >
3
4      <item android:id="@+id/item1"  android:title="Item 1"></item>
5      <item android:id="@+id/item2"  android:title="Item 2"></item>
6      <item android:id="@+id/item3"  android:title="Item 3"></item>
7  </menu>
```

```
15      @Override
16      public void onCreate(Bundle savedInstanceState) {
17          super.onCreate(savedInstanceState);
18          setContentView(R.layout.main);
19      Button b =(Button) findViewById(R.id.button1);
20      registerForContextMenu(b);
21
22      }
```

```
    @Override
    public void onCreateContextMenu(ContextMenu menu, View v,
            ContextMenuInfo menuInfo) {
        // TODO Auto-generated method stub
        super.onCreateContextMenu(menu, v, menuInfo);
        MenuInflater inf = getMenuInflater();
        inf.inflate(R.menu.contextmenu, menu);

    }
```

```
35      @Override
36      public boolean onContextItemSelected(MenuItem item) {
37          // TODO Auto-generated method stub
38
39          Toast ts;
40      if(item.getItemId()==R.id.item1)
41       {
42        ts =Toast.makeText(this, item.getTitle()+" bir", 5000);
43       ts.show();
44
45       }
46       else if(item.getItemId()==R.id.item2)
47       {
48              ts =Toast.makeText(this, item.getTitle()+" iki", 5000);
49           ts.show();
50
51       }
52       else
53       {
54             ts =Toast.makeText(this, item.getTitle()+ " üç", 5000);
55           ts.show();
56
57       }
58
59      return super.onContextItemSelected(item);
60       }
```

main.xml | contextmenu.xml | main.java | main.java

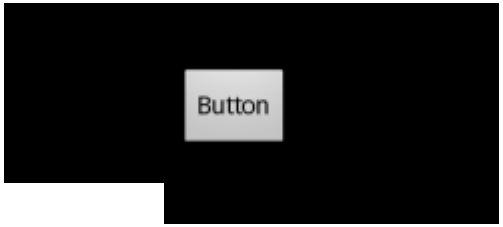
```xml
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
   <group android:checkableBehavior="single">
    <item android:id="@+id/item1"  android:title="Item 1"></item>
    <item android:id="@+id/item2"  android:title="Item 2"></item>
    </group>
    <group android:checkableBehavior="all" >
    <item android:id="@+id/item3"  android:title="Item 3"></item>
    <item android:id="@+id/item4"  android:title="Item 4"></item>
    </group>
 </menu>
```

Alert Kutucuğu

```
    Button b =(Button) findViewById(R.id.button1);
    registerForContextMenu(b);

    b.setOnClickListener( new OnClickListener() {

        public void onClick(View v) {
            // TODO Auto-generated method stub
            AlertDialog.Builder bld= new AlertDialog.Builder(main.this);
            bld.setCancelable(false);
            bld.setMessage("çıkayım mı abi");
```

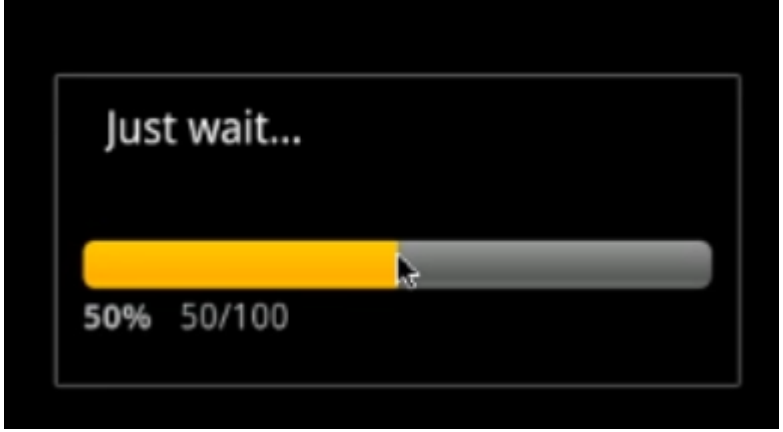
```
  bld.setPositiveButton("ÇIK", new DialogInterface.OnClickListener() {

      public void onClick(DialogInterface dialog, int which) {
          // TODO Auto-generated method stub
      main.this.finish();
      }
  });

  bld.setNegativeButton("İptal", new  DialogInterface.OnClickListener() {

      public void onClick(DialogInterface dialog, int which) {
          // TODO Auto-generated method stub

          dialog.cancel();

      }
  });
});

AlertDialog alrt = bld.create();
alrt.show();
```

Progress Dialog




```
final ProgressDialog pd = new ProgressDialog(this);
pd.setProgressStyle(ProgressDialog.STYLE_SPINNER);
pd.setMessage("Just wait...");
pd.setIndeterminate(true);
pd.setCancelable(true);

Button b = (Button) findViewById(R.id.button1);
b.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View arg0) {
        pd.show();
    }
});
```

- ağlar
- SetIndetermined: ne kadar süreceği belli mi değil mi
- setcancelable: back tuşu ile iptal edilebilir mi



- Style_Horizontal
- pd.setProgress (yüzde cinsinden ilerleme)

```
final ProgressDialog pd = new ProgressDialog(this);
pd.setProgressStyle(ProgressDialog.STYLE_HORIZONTAL);
pd.setMessage("Just wait...");
pd.setIndeterminate(false);
pd.setCancelable(true);

Button b = (Button) findViewById(R.id.button1);
b.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View arg0) {
        pd.show();
        pd.setProgress(50);
    }
```

Custom dialogs

- Custom layout
  - Xml layout oluşturulur. Tasarlanır
  - Paddingler ayarlanır
  - Çalıştırılır

```
Button b = (Button) findViewById(R.id.button1);
b.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View arg0) {
        Dialog d = new Dialog(Main.this);
        d.setContentView(R.layout.dialog);
        d.setTitle("This is important");
        d.show();
    }
});
```

Sound

- Media Player sınıfı
- Projede raw adlı klasörde medya dosyaları sakanır.

```
Button b = (Button) findViewById(R.id.button1);
b.setOnClickListener(new OnClickListener() {

    @Override
    public void onClick(View v) {
        MediaPlayer mp = MediaPlayer.create(Main.this, R.raw.beep
        mp.start();
    }
});
```

Video

- Test i gerçek bir cihaz üzerinde test etmeniz önerilir.
  - Mp4, h264
- Layout.. VideoView tüm layout kaplar

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    VideoView v = (VideoView) findViewById(R.id.videoView1);
    v.setVideoPath("/sdcard/vid.mp4");
    v.setMediaController(new MediaController(this));
    v.start();
    v.requestFocus();
}
```

Kamera ile çalışmak

- İzin alınması gerekir
- İmagaview
- Intent (android.provider.MediaStore.Actio

```
    iv = (ImageView) findViewById(R.id.imageView1);

    Button b = (Button) findViewById(R.id.button1);
    b.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View v) {
            Intent intent = new Intent(android.provider.MediaStore.ACTION_IMAG
            startActivityForResult(intent, 0);
        }
    });
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
    // TODO Auto-generated method stub
    super.onActivityResult(requestCode, resultCode, data);

    Bitmap bm = (Bitmap) data.getExtras().get("data");
    iv.setImageBitmap(bm);
}
```

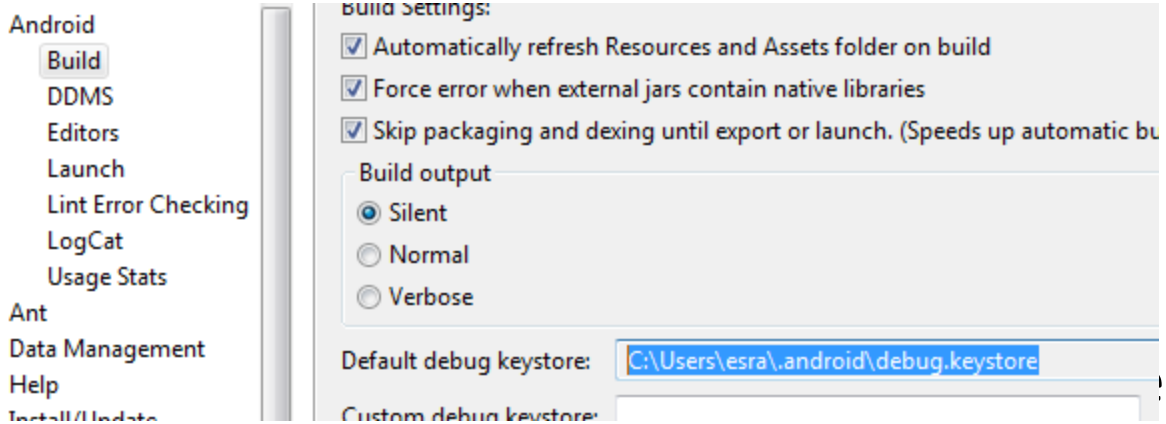
Google MAP

- Özel bir SDK in yuklendiğine emin olmalıyız
  - GoogleAPI yüklü SDK yüklenmeli
- Bu SDK yi kullanan bir Sanal makine de oluşturulmalı
  - SD card alanı belirlenmeli
  - Snapshoot a izin ver
- Yeni proje alçıldığında, build target kısmında google API lı SDK seçilmeli

Google API KEY alma

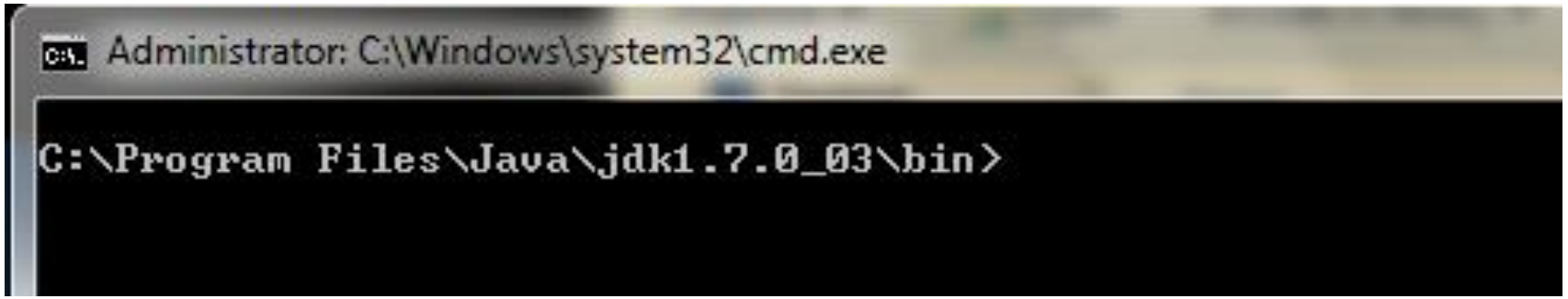
- http://code.google.com/android/add-ons/google-apis
- Maps external Library
  - Get a Maps API KEY
  - http://code.google.com/android/maps-api-signup.html
  - Java key tool ile MD5

- Şimdiye kadar hep debug KEY kullandık
  - Debug certificası nerede ile uyulamaları imzalarız



keytool var

  - Oraya git



- Keytool -list -alias androiddebugkey -keystore «C:\Users\esra\.android\debug.keystore» -storepass android –keypass android -v
- 91:13:23:0E:73:B5:98:DA:86:74:01:4B:87:88:8F:04
- Bunu web sayfasına yapıştıracağız

```
0QKkdU[redacted]hooy4JKJMKuu8j-Uwvmhd-7cts8A
```

This key is good for all apps signed with your certificate whose fingerprint is:

```
91:13:23:0E:73:B5:98:DA:86:74:01:4B:[redacted]
```

Here is an example xml layout to get you started on your way to mapping glory:

```
<com.google.android.maps.MapView
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:apiKey="0QKkdU[redacted]hooy4JKJMKuu8j-Uwvmhd-7cts8A"
   />
```

- Bu kodu Layout.xml e gömeriz
- İd tanımlanır. @+id/themap
- android:clickable=«true»

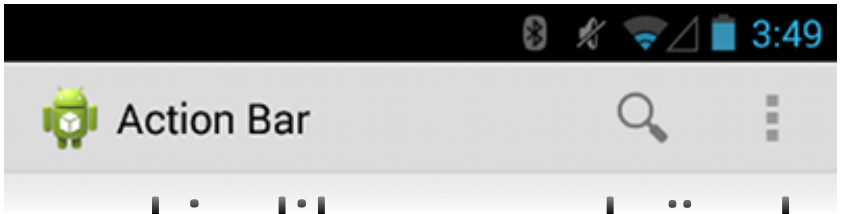
Manifest izinleri

- Uses Permission
  - Android.permission.INTERNET
  - Android.permission.Fine
- Application içinde
  - Uses-Library
    android:name=com.google.android.maps

Map Activity

- M

Action BAR



- Uygulamanıza kimlik verecek özel olarak ayrılmış kullanıcı lokasyonunu belirten özel olarak ayrılmış alan
- Uygulamadaki önemli fonk. Tahmin edilebilir şekilde erişmek için (arama gibi)
- Uygumala içinde gezinti ve görüntü değiştirmek için (Tab veya drop down list)
- Android 3.0 ile geldi API 11 ama API 7 (2.1)Support library ile erişim sağlanır.

Action Bar Cont.

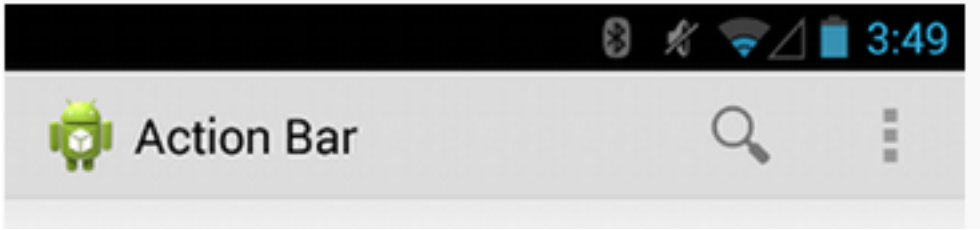
- import android.support.v7.app.ActionBar
- import android.app.ActionBar

```
<manifest ... >
    <uses-sdk android:minSdkVersion="11" ... />
    ...
</manifest>
```

- The action bar is included in all activities that use the Theme.Holo theme (or one of its descendants)

Action Button

- search actıon button nu ve actıon overflow



- Tüm butunlar Bir XML de tanımlanırlar menu resource olan bir XML olamlıdır.
- Bir **<item>** elementi ile istenildiği kadar eleman eklenir.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <!-- Search, should appear as action button -->
    <item android:id="@+id/action_search"
          android:icon="@drawable/ic_action_search"
          android:title="@string/action_search"
          android:showAsAction="ifRoom" />
    <!-- Settings, should always be in the overflow -->
    <item android:id="@+id/action_settings"
          android:title="@string/action_settings"
          android:showAsAction="never" />
</menu>
```

- action_search action butunu yer varsa gösterilecektir.
- action_settings adlı action buttonu

Add Action to Action Bar

Handling single and multi touch on Android – Tutorial Vogella

The Android standard View class support touch events. You can react to touch events in your custom views and your activities. Android supports multiple pointers, e.g. fingers which are interacting with the screen.

To react to touch events you override the onTouchEvent() method.

The motion Event Class ::: the number of pointers, the X/Y coordinates and size and pressure of each pointer.

Single touch

Via the `getAction()` method you receive the action which was performed. The `MotionEvent` class provides the following constants to determine the action which was performed.

Table 1. Touch Events

Event	Description
MotionEvent.ACTION_DOWN	New touch started
MotionEvent.ACTION_MOVE	Finger is moving
MotionEvent.ACTION_UP	Finger went up
MotionEvent.ACTION_CANCEL	Current event has been canceled, something else took control of the touch event
MotionEvent.ACTION_POINTER_DOWN	Pointer down (multi-touch)
MotionEvent.ACTION_POINTER_UP	Pointer up (multi-touch)

```
import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Path;
import android.util.AttributeSet;
import android.view.MotionEvent;
import android.view.View;

public class SingleTouchEventView extends View {
  private Paint paint = new Paint();
  private Path path = new Path();

  public SingleTouchEventView(Context context, AttributeSet attrs) {
    super(context, attrs);

    paint.setAntiAlias(true);
    paint.setStrokeWidth(6f);
    paint.setColor(Color.BLACK);
    paint.setStyle(Paint.Style.STROKE);
    paint.setStrokeJoin(Paint.Join.ROUND);
  }

  @Override
  protected void onDraw(Canvas canvas) {
    canvas.drawPath(path, paint);
  }
```

```
  @Override
  public boolean onTouchEvent(MotionEvent event) {
    float eventX = event.getX();
    float eventY = event.getY();

    switch (event.getAction()) {
    case MotionEvent.ACTION_DOWN:
      path.moveTo(eventX, eventY);
      return true;
    case MotionEvent.ACTION_MOVE:
      path.lineTo(eventX, eventY);
      break;
    case MotionEvent.ACTION_UP:
      // nothing to do
      break;
    default:
      return false;
    }

    // Schedules a repaint.
    invalidate();
    return true;
  }
```

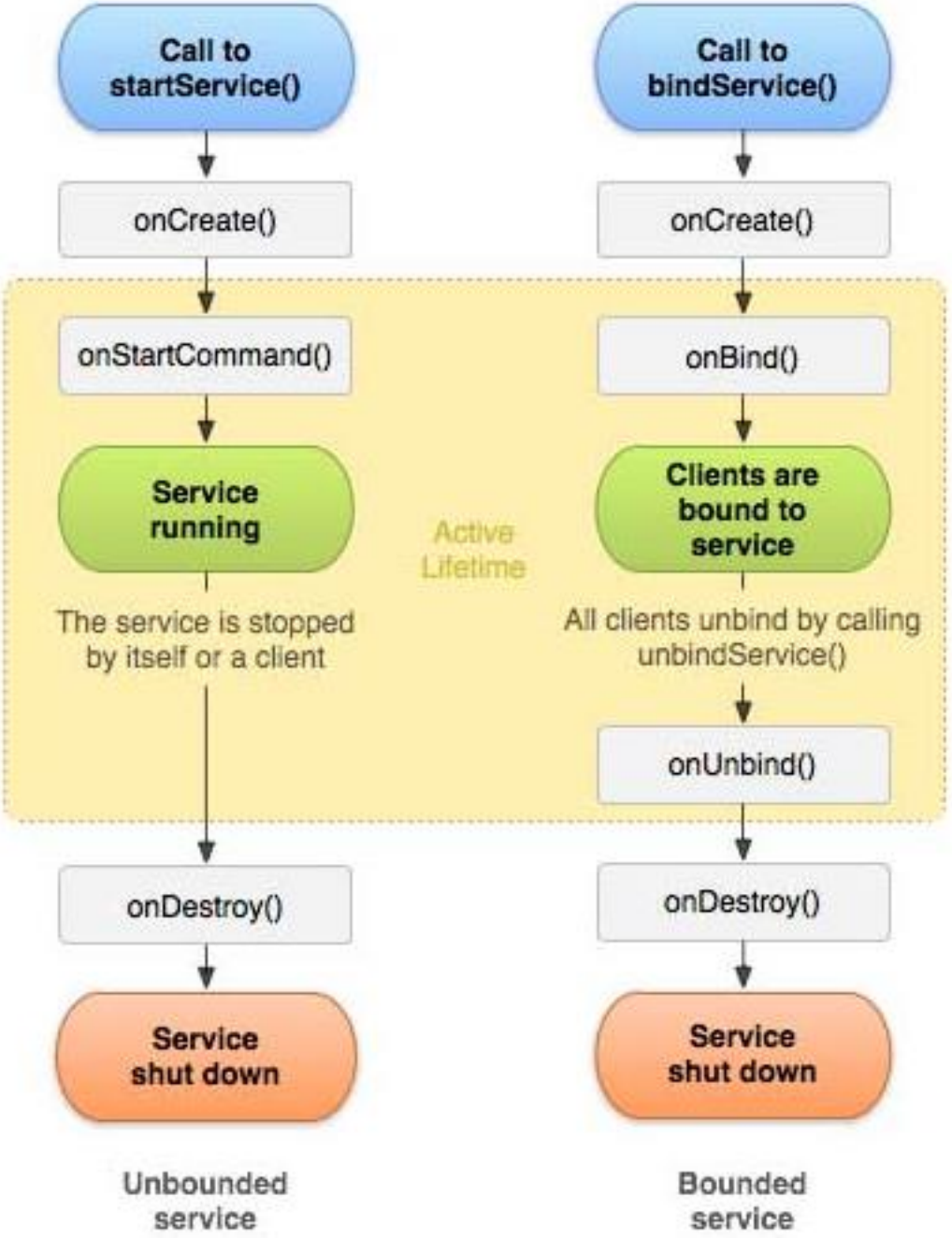
```
public class SingleTouchActivity extends Activity {
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(new SingleTouchEventView(this, null));
  }
}
```

Tracking

SERVICES (www.tutorialspoint.com)

- A service is a component that runs in the background to perform long-running operations without needing to interact with the user.
- A service can essentially take two states:

State	Description
Started	A service is **started** when an application component, such as an activity, starts it by calling *startService()*. Once started, a service can run in the background indefinitely, even if the component that started it is destroyed.
Bound	A service is **bound** when an application component binds to it by calling *bindService()*. A bound service offers a client-server interface that allows components to interact with the service, send requests, get results, and even do so across processes with interprocess communication (IPC).


Call to startService()
onCreate()
onStartCommand()
Service running
The service is stopped by itself or a client
onDestroy()
Service shut down
Unbounded service
Active Lifetime
Call to bindService()
onCreate()
onBind()
Clients are bound to service
All clients unbind by calling unbindService()
onUnbind()
onDestroy()
Service shut down
Bounded service

Create A Service

- To create an service, you create a Java class that extends the Service base class or one of its existing subclasses.

Callback methods

Callback	Description
onStartCommand()	The system calls this method when another component, such as an activity, requests that the service be started, by calling *startService()*. If you implement this method, it is your responsibility to stop the service when its work is done, by calling *stopSelf()* or *stopService()* methods.
onBind()	The system calls this method when another component wants to bind with the service by calling *bindService()*. If you implement this method, you must provide an interface that clients use to communicate with the service, by returning an *IBinder* object. You must always implement this method, but if you don't want to allow binding, then you should return *null*.
onUnbind()	The system calls this method when all clients have disconnected from a particular interface published by the service.
onRebind()	The system calls this method when new clients have connected to the service, after it had previously been notified that all had disconnected in its *onUnbind(Intent)*.
onCreate()	The system calls this method when the service is first created using *onStartCommand()* or *onBind()*. This call is required to perform one-time setup.
onDestroy()	The system calls this method when the service is no longer used and is being destroyed. Your service should implement this to clean up any resources such as threads, registered listeners, receivers, etc.

```
package com.tutorialspoint;

import android.app.Service;
import android.os.IBinder;
import android.content.Intent;
import android.os.Bundle;

public class HelloService extends Service {

  /** indicates how to behave if the service is killed */
  int mStartMode;
  /** interface for clients that bind */
  IBinder mBinder;
  /** indicates whether onRebind should be used */
  boolean mAllowRebind;

  /** Called when the service is being created. */
  @Override
  public void onCreate() {

  }

  /** The service is starting, due to a call to startService() */
  @Override
  public int onStartCommand(Intent intent, int flags, int
startId) {
     return mStartMode;
  }

  /** A client is binding to the service with bindService() */
  @Override
  public IBinder onBind(Intent intent) {
    return mBinder;
  }

  /** Called when all clients have unbound with
unbindService() */
  @Override
  public boolean onUnbind(Intent intent) {
    return mAllowRebind;
  }

  /** Called when a client is binding to the service with
bindService()*/
  @Override
  public void onRebind(Intent intent) {

  }

  /** Called when The service is no longer used and is being
destroyed */
  @Override
  public void onDestroy() {

  }
}
```

Service Example

Step	Description
1	You will use Eclipse IDE to create an Android application and name it as *HelloWorld* under a package *com.example.helloworld* as explained in the *Hello World Example* chapter.
2	Modify main activity file *MainActivity.java* to add *startService()* and *stopService()* methods.
3	Create a new java file *MyService.java* under the package *com.example.helloworld*. This file will have implementation of Android service related methods.
4	Define your service in *AndroidManifest.xml* file using <service.../> tag. An application can have one or more services without any restrictions.
5	Modify the detault content of *res/layout/activity_main.xml* file to include two buttons in linear layout.
6	Define two constants *start_service* and *stop_service* in *res/values/strings.xml* file
7	Run the application to launch Android emulator and verify the result of the changes done in the aplication.

Step1-2

```
    // Method to start the service
    public void startService(View view) {
        startService(new Intent(getBaseContext(), MyService.class));
    }

    // Method to stop the service
    public void stopService(View view) {
        stopService(new Intent(getBaseContext(), MyService.class));
    }
}
```

Step 3

```
package com.example.helloworld;

import android.app.Service;
import android.content.Intent;
import android.os.IBinder;
import android.widget.Toast;

public class MyService extends Service {
   @Override
   public IBinder onBind(Intent arg0) {
      return null;
   }

   @Override
   public int onStartCommand(Intent intent, int flags, int startId) {
      // Let it continue running until it is stopped.
      Toast.makeText(this, "Service Started", Toast.LENGTH_LONG).show();
      return START_STICKY;
   }
   @Override
   public void onDestroy() {
      super.onDestroy();
      Toast.makeText(this, "Service Destroyed", Toast.LENGTH_LONG).show();
   }
}
```

Step 4-5-6

```
        </activity>
        <service android:name=".MyService" />
    </application>
</manifest>
```

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <Button android:id="@+id/btnStartService"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/start_service"
    android:onClick="startService"/>

    <Button android:id="@+id/btnStopService"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/stop_service"
    android:onClick="stopService" />
```

```
<resources>

    <string name="app_name">HelloWorld</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">MainActivity</string>
    <string name="start_service">Start Service</string>
    <string name="stop_service">Stop Service</string>

</resources>
```

DialogFragmant

- Dialog→DialogFragmant→
  - AlertDialog
  - ProgressDialog
  - DatePicker
  - TimePicker

Services -Vogella

- A *service* is a component which runs in the background without direct interaction with the user. As the service has no user interface, it is not bound to the lifecycle of an activity
- Services are used for repetitive and potentially long running operations, i.e., Internet downloads, checking for new data, data processing, updating content providers and the like.

Services

- Services run with a higher priority than inactive or invisible activities and therefore it is less likely that the Android system terminates them. Services can also be configured to be restarted if they get terminated by the Android system once sufficient system resources are available again.

Permissions API 23 Vogella

- **1. Security and permissions**
- **1.1. Security concept in Android**
- The Android system installs every Android application with a unique user and group ID. Each application file is private to this generated user, e.g., other applications cannot access these files. In addition, each Android application is started in its own process.
- Therefore, by means of the underlying Linux kernel, every Android application is isolated from other running applications.
- If data should be shared, the application must do this explicitly via an Android component which handles the sharing of the data, e.g., via a service or a content provider.

- **1.2. Permission concept in Android**
- Android contains a permission system and predefined permissions for certain tasks. Every application can request required permissions. For example, an application may declare that it requires network access. It can also define new permissions.
- System permissions have different levels, e.g., protection levels.
- The permission concept has changed since API 23. Before API level 23 the user was asked during installation, after API level the user is asked during runtime.
- An Android application declares the required permissions in its Android manifest. It can also define additional permissions which it can use to restrict access to certain components.

- The two most important protection levels are normal and dangerous permissions:
- **Normal permissions** are permissions which are deemed harmless for the users privacy or the operation of other applications. For example, the permission to set the time zone. Normal permission are automatically granted to the application. See Normal permissions for a complete list.
- **Dangerous permissions** affect the users private information, or could potentially affect his data or the operation of other application. For example, the ability to read the users contact data. Dangerous permissions must be granted by the user at runtime to the app.

Callender Check

```
int permissionCheck = checkSelfPermission(this,
Manifest.permission.WRITE_CALENDAR);
if (!permissionCheck == PackageManager.PERMISSION_GRANTED) {
  // User may have declined earlier, ask Android if we should show him a
reason
    if (shouldShowRequestPermissionRationale(thisActivity,
Manifest.permission.WRITE_CALENDAR)) {
        // show an explanation to the user
        // Good practise: don't block thread after the user sees the
explanation, try again to request the permission.
    } else {
        // request the permission.
        // CALLBACK_NUMBER is a integer constants
        requestPermissions(thisActivity,  new String[]
{Manifest.permission.WRITE_CALENDAR}, CALLBACK_NUMBER);
        // The callback method gets the result of the request.
    }
} else {
// got permission use it
}
```

Using Toolbar

- Entries in the toolbar are typically called *actions*. While it is possible to create entries in the action bar via code, it is typically defined in an XML resource file.
- Each menu definition is contained in a separate file in the ***res/menu*** folder. The Android tooling automatically creates a reference to menu item entries in the R file, so that the menu resource can be accessed.
- An activity adds entries to the action bar in its **onCreateOptionsMenu()** method.
- The **showAsAction** attribute allows you to define how the action is displayed. For example, the **ifRoom** attribute defines that the action is only displayed in the action bar if there is sufficient screen space available.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >

    <item
        android:id="@+id/action_refresh"
        android:orderInCategory="100"
        android:showAsAction="always"
        android:icon="@drawable/ic_action_refresh"
        android:title="Refresh"/>
    <item
        android:id="@+id/action_settings"
        android:title="Settings">
    </item>

</menu>
```

```
@Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.mainmenu, menu);
        return true;
    }
```

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
  switch (item.getItemId()) {
  // action with ID action_refresh was selected
  case R.id.action_refresh:
    Toast.makeText(this, "Refresh selected", Toast.LENGTH_SHORT)
        .show();
    break;
  // action with ID action_settings was selected
  case R.id.action_settings:
    Toast.makeText(this, "Settings selected", Toast.LENGTH_SHORT)
        .show();
    break;
  default:
    break;
  }

  return true;
}
```

- Searching
- To search for a menu item in a menu you can use the findItem() method of the Menuclass. This method allows to search by id.
- **Changing the menu**
- The onCreateOptionsMenu() method is only called once. If you want to change the menu later, you have to call the invalidateOptionsMenu() method. Afterwards this onCreateOptionsMenu() method is called again.