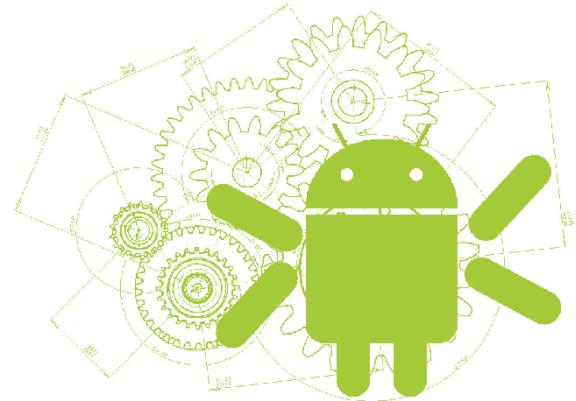


POZITRON®

Android Developer Days



Can Elmas

Android Team Leader

Ismail Ceylan

Android Developer

Mehmet Arıkan

Android Developer

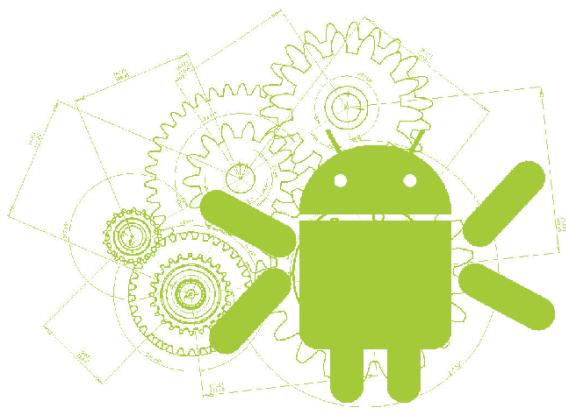


Agenda

- Android Fundemantals
- Sample Projects
- Contest!

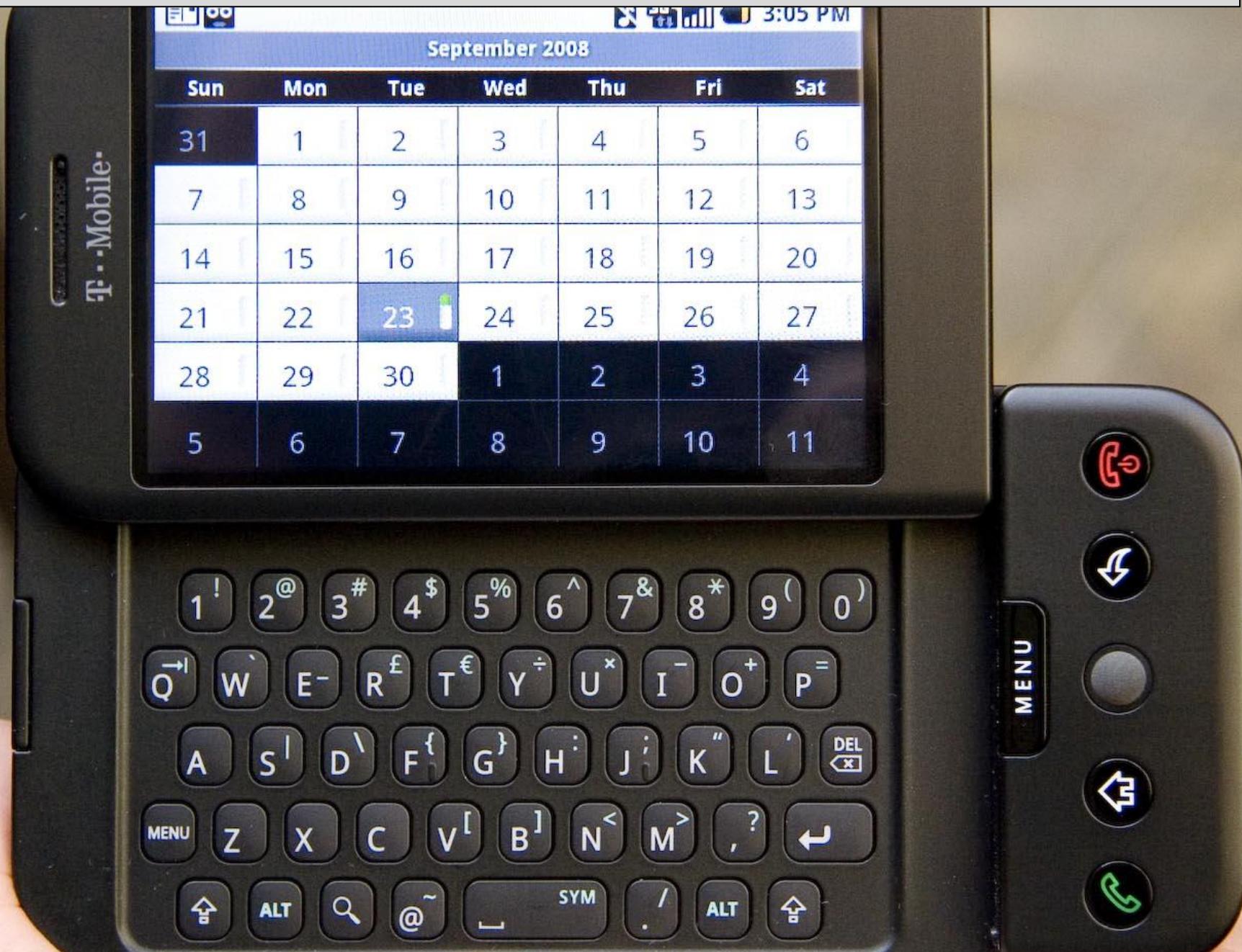


- Leader in Mobile App Development
- Many native apps in Turkey, and also in other markets: Middle East, Europe, Russia, USA
- 60% of Pozitron = Computer Sci./Eng. Grads
- IsCep, YKB, THY, Bilyoner, HepsiBurada, Biletix, DohaBank (Middle East), Milo (USA), Brands4Friends (Germany) and more...



Introduction to Android Platform

T-Mobile G1 / HTC Dream - 2008



What is Android ?

Android is an **open-source** software stack **for mobile devices**, and a corresponding **open-source project** led by Google.

<http://source.android.com>

- you can get the code
- you can modify if you want
- you can build and deploy to your device

Android Software Stack

Android Application

Application Framework

Activity M. Window M. Content P. View System
Package M. Tel. Manager Resource M. Location M. Notification

Java
Dalvik VM

Develop
using SDK

Android RT

Core Libraries

Android RT Dalvik VM

Libraries

Media SQLite
Surface M. SGL
Freetype Webkit
OpenGL/ libc
SSL

C/C++
Native

Develop
using NDK

Linux Kernel

Kernel
Root Level

Android Architecture

- A multi user Linux system. Each application is a different user : Kernel level Application Sandbox
- Every app run in its own Linux process, each process has its own VM
- Permissions set at the install time
- Complete isolation

For more on Security,

<http://source.android.com/tech/security/index.html>

What is Dalvik Virtual Machine (VM) ?

- Software that runs the android apps
- **java > class > dex**
javac dx
- More suitable in terms of memory and processor speed

class is run by Java Virtual Machine

dex is run by Dalvik Virtual Machine

For more on DVM, http://davidehringer.com/software/android/The_Dalvik_Virtual_Machine.pdf

Developer Tools

Download

Installing the
SDK

Exploring the SDK

NDK

Workflow

Tools Help

Revisions

Extras

Samples

ADK



Get the Android SDK

The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android.

**SDK is for Java targeting Dalvik VM.
NDK is for C/C++ targeting native environment.
*You will probably need only SDK.***

Platform	Package	Size	MD5 Checksum
Windows	android-sdk_r20.0.3-windows.zip	90379469 bytes	cd895c79201f7f02507eb3c3868a1c5e
	installer_r20.0.3-windows.exe (Recommended)	70495456 bytes	cf23b95d0c9cd57fac3c3be253171af4
Mac OS X (intel)	android-sdk_r20.0.3-macosx.zip	58218455 bytes	07dc88ba2c0817ef178a665d002831bf
Linux (i386)	android-sdk_r20.0.3-linux.tgz	82616305 bytes	0d53c2c31d6b5d0cf7385bccd0b06c27

Android Developer Documentation

Extensive documentation at

<http://developer.android.com>

Check also Google IO Conference

<https://developers.google.com/events/io/>

API Levels

Android versions are mapped to API Levels. e.g. Android 4.1 is API Level 16

API features are available or not for a specific API Level. Some features are back-ported with Android Support Library.

4.0+ (API Level 14+) is 22%

2.3.3 - 2.3.7 (API Level 10) is 57%

API Levels

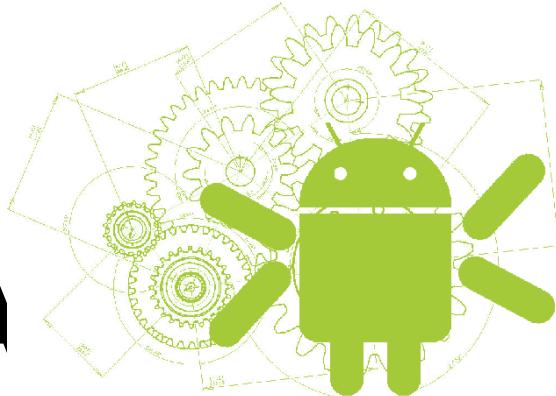
- Cupcake (1.5 - API Level 3)
- Donut
- Eclair
- Froyo
- Gingerbread
- Honeycomb (3.0 - API Level 11)
- ICS (4.0 - API Level 14)
- JellyBean (4.1 - API Level 16)
- Key Lime Pie ???

Android Development Workflow

- Setup development environment (SDK)
- Build / Debug / Test (in your favorite IDE)
 - Virtual Device Setup / Physical Device
 - DDMS (Dalvik Debug Monitor Server)
 - LogCat
 - Lint
- Publish (to Google Play)

First Android Project

- Folders
- Layouts and Views
- R class
- Android Manifest
- Activity
- Intent



A n **Fundamentals**

Activities
Intents and Intent Filters
Services
Content Providers
Broadcast Receivers
Fragments
Android Manifest - Permissions

Activity

- * Typically correspond to one screen in a UI
- * But, they can:
 - be faceless
 - return a value
- * An application with User Interface, consist of one or more Activities
- * Managed as an *activity stack (LIFO)*

Activity

The screenshot shows the Twitter mobile application interface. At the top, there's a blue header bar with the Twitter logo and a compose button. Below it is a navigation bar with four tabs: Home, Connect, Discover, and Me. The main area displays a feed of tweets from four users:

- Nick Fisher @Nick** 2m ago
Love being reminded of the power and magic of #space
- Rob Chiswa @RobChiz** 2m ago
Hilarious. Cats in #Space:
omgcatsinspace.tumblr.com
- Lisa Wang @ldubs** 4m ago
"Somewhere, something incredible is waiting to be known."
Carl Sagan #space
pic.twitter.com/qbjx26r
- Coleen Baik @colbay** 5m ago
Incredible, hi-res Images of Earth from #space - thanks NASA satellite! is.gd/PFm9b9

The screenshot shows the Biletix mobile application interface. At the top, there's a header bar with the Biletix logo and a location indicator for Marmara. Below it is a navigation bar with three tabs: Hot Tickets, Bu Hafta Sonu (selected), and Yakınımda. The main area displays a list of events with small images and details:

- Jurassic Land (Aralık 2011 sürekli etkinlidir)**
Jurassic Land
26 Ara 2011 - 31 Ara 2011
- Orphans / Öksüzler**
Dot / Dotmarsta Salonu
29 Ara 2011 - 30 Ara 2011
- Paris Sirki**
Ümraniye CarrefourSA
30 Ara 2011
- Genç Yetenekler Buluşması**
Akbank Sanat
30 Ara 2011
- Fenerbahçe Ülker - Galatasaray MP**
Sinan Erdem Spor S.
30 Ara 2011
- Atlantis Sirki**
Bursa Anatolian AVM

```

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

    @Override
    protected void onStart() {
        super.onStart();

        // YOU ARE ON THE RUN LIST
        // YOU CAN START EXECUTION ON ANY MOMENT
    }

    @Override
    protected void onResume() {
        super.onResume();

        // YOU NOW HAVE FOCUS
        // YOU ARE THE TOP LEVEL APPLICATION
    }

    @Override
    protected void onPause() {
        super.onPause();

        // YOU ARE NO LONGER THE TOP OF THE STACK
        // AND NO LONGER HAVE FOCUS
    }

    @Override
    protected void onStop() {
        super.onStop();

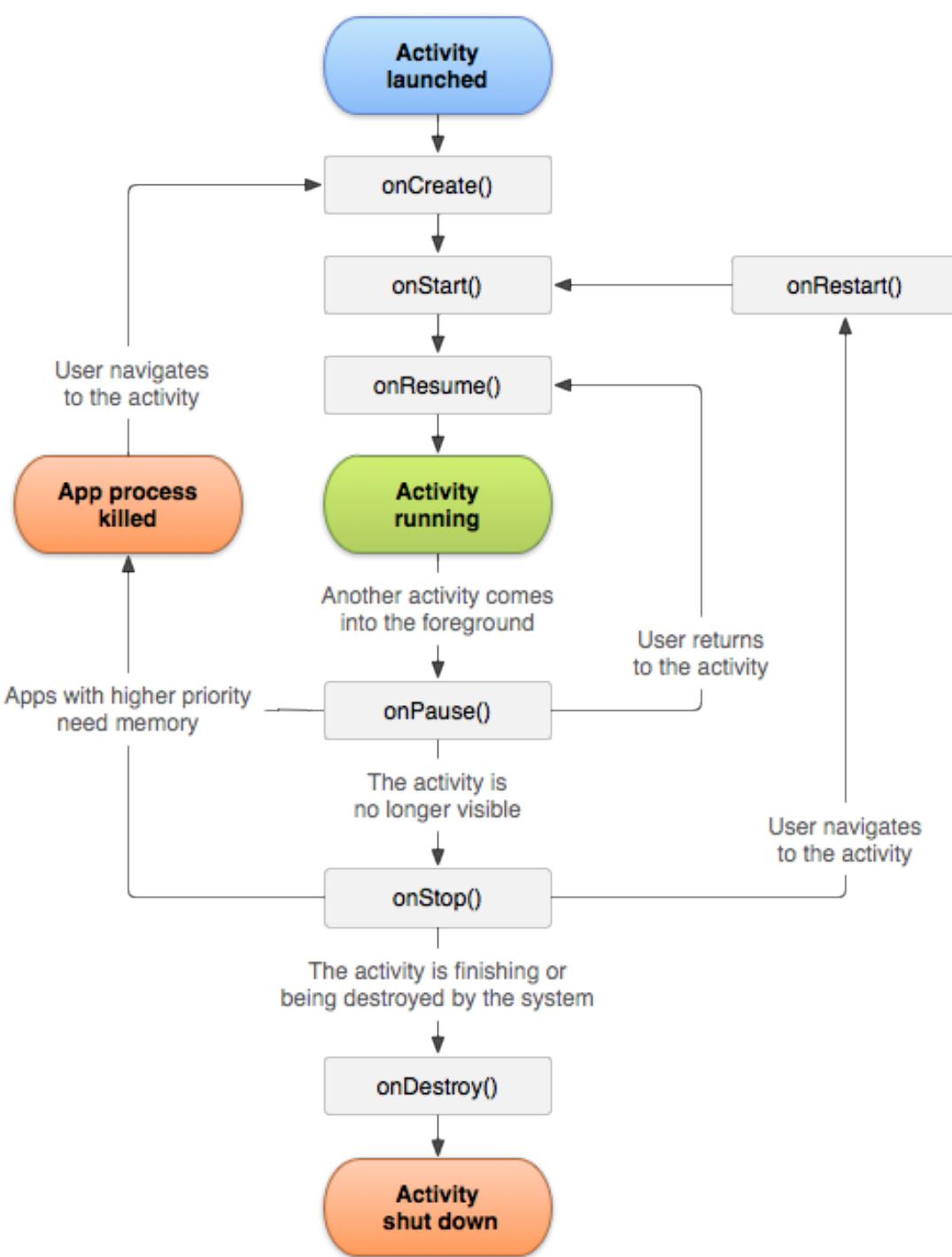
        // NO LONGER RUN IN FUNCTIONS
    }

    @Override
    protected void onRestart() {
        super.onRestart();

        // YOU ARE BACK ON THE RUN LIST
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
    }
}

```



Activity

- Activity States managed by the Android system
- Declaration in the android manifest file
- "Main Activity" must be set

<http://developer.android.com/guide/components/activities.html>

Activity

- Activity with UI content

```
public class MainActivity extends Activity {  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.main_act);  
    }  
}
```

Intent

- Intention
- Passive data structure holding an abstract description of an operation to be performed
- A message to the OS that you want to do something

Intent

- Primary Attributes :
 - action : General action to be performed (ACTION_VIEW, ACTION_MAIN etc..)
 - data : Data to operate on, expressed as a URI
- Depending on how Intent is constructed, Android system determines what to do e.g finds the appropriate activity, service, or set of broadcast receivers

Intent - Explicit

- Used to navigate from one Activity to another in app
- Data transfer between Activities

```
Intent intent = new Intent(ActHome.this, BrowserActivity.class);  
intent.putExtra("url", "http://www.google.com");  
startActivity(intent);
```

Intent - Implicit

- Used to open other applications
 - Ex : Open a specific URL in a user's preferred browser

```
String url = "http://www.example.com";
Intent i = new Intent(Intent.ACTION_VIEW);
i.setData(Uri.parse(url));
startActivity(i);
```

Intent

- IPC mechanism
- Depending on how Intent is constructed, Android system finds the appropriate activity, service, or set of broadcast receivers to respond the intent

Intent Filter

- Description of what intent an activity is capable of handling
- "Main" Activity :

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

    <application android:icon="@drawable/icon" android:label="@string/app_name">

        <activity android:name=".activities.MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>

        <activity android:name=".activities.ActTwo"/>

    </application>

</manifest>
```

Intent Filter

- Depending on how Intent is constructed, Android system finds the appropriate activity, service, or set of broadcast receivers to respond the intent

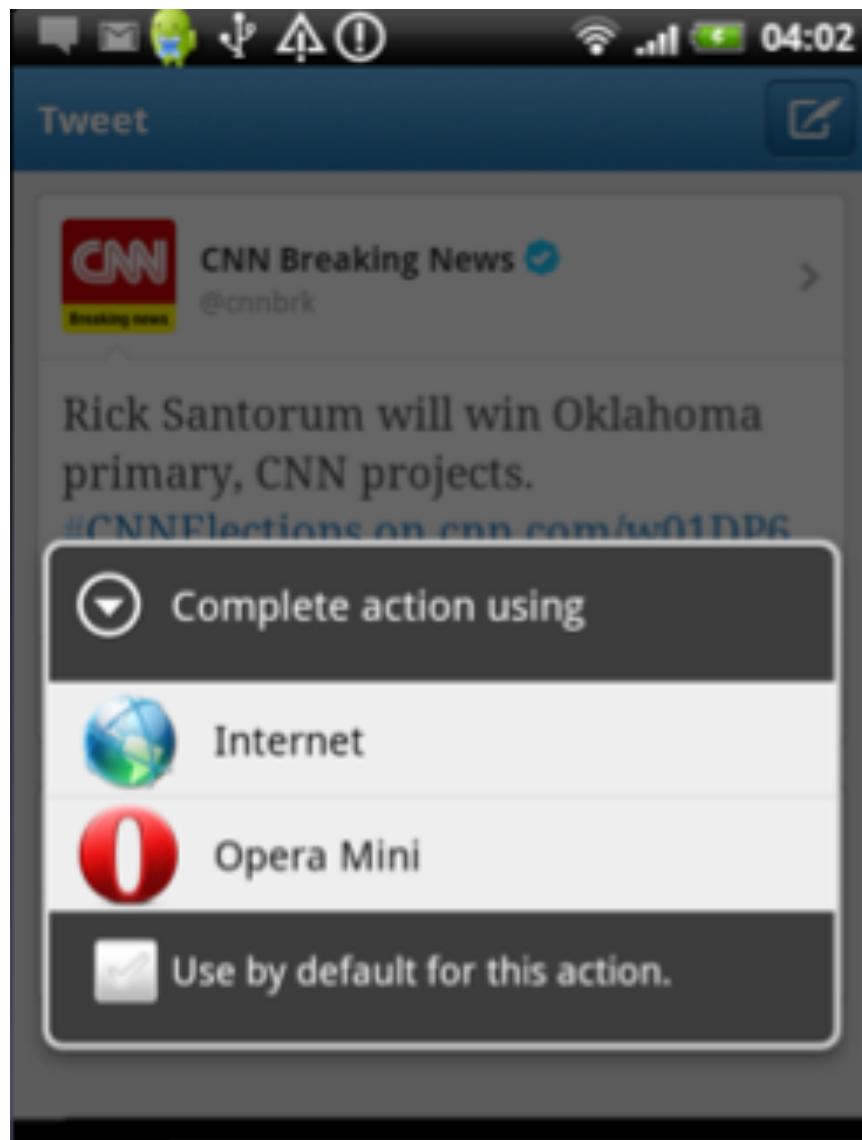
Intent Filter - Crappy Browser

- Register for intent

```
<activity android:name=".ui.activities.BrowserActivity">  
    <intent-filter>  
        <action android:name="android.intent.action.VIEW" />  
        <category android:name="android.intent.category.DEFAULT" />  
        <data android:scheme="http"/>  
    </intent-filter>  
</activity>
```

Intent Filter

- Receive Intent



Broadcast Receiver

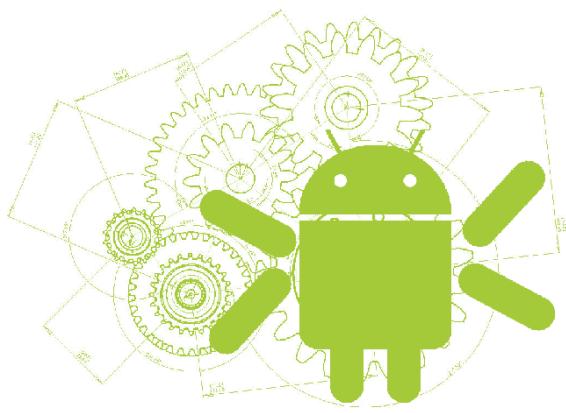
- Component which allows to register for system or application events
- You cannot trigger system Broadcasts, Android will prevent it; you can define your own actions
- System Broadcasts:
 - Intent.ACTION_BOOT_COMPLETED
 - Intent.ACTION_BATTERY_LOW
 - Intent.ACTION_POWER_DISCONNECTED
 - etc..

Services

- Faceless
- Runs in the background
- Pre-defined Services :
 - Context.LAYOUT_INFLATOR_SERVICE
 - Context.LOCATION_SERVICE
- Might run on its own thread or in activity

Content Provider

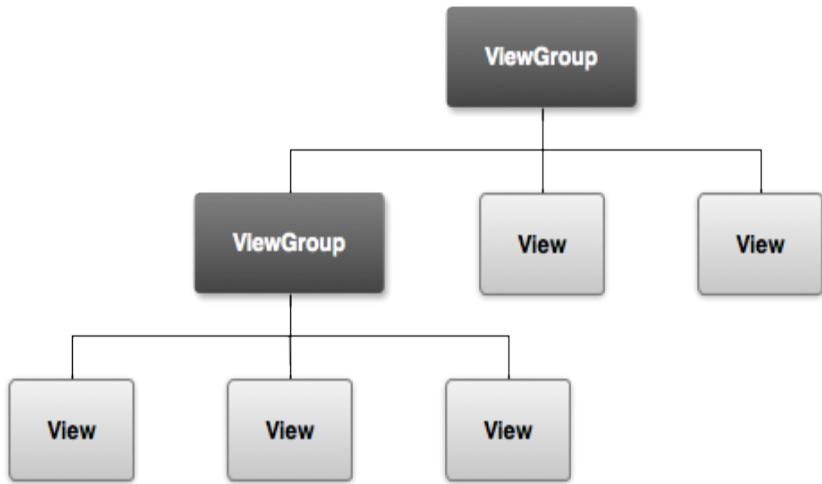
- Access to a central repository of data
- Intended to be used by other applications
- IPC
- Contacts, Calendar, your own custom provider etc.



User Interface @ Android

Overview

- * UI is defined by hierarchy of *View's* and *ViewGroup's*.
- * Views are highly customizable. You can even create your own custom views.
- * Layouts can be created from code programmatically or can be defined in XML files.
- * Both methods have advantages

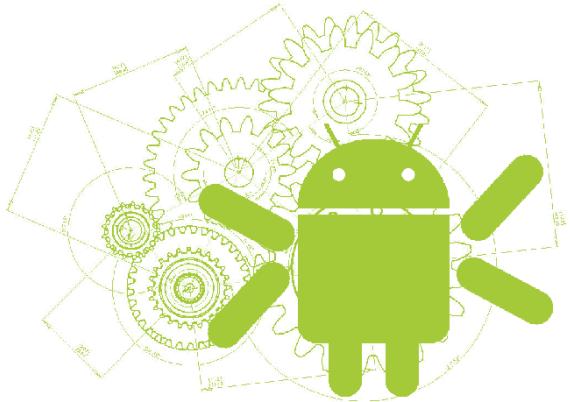


Overview

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="I am a TextView" />
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="I am a Button" />
</LinearLayout>
```

- * View's created in XML files can be referenced from code via ID's.
- * We should set content view if layout is defined in XML file:

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



User Interface Demo



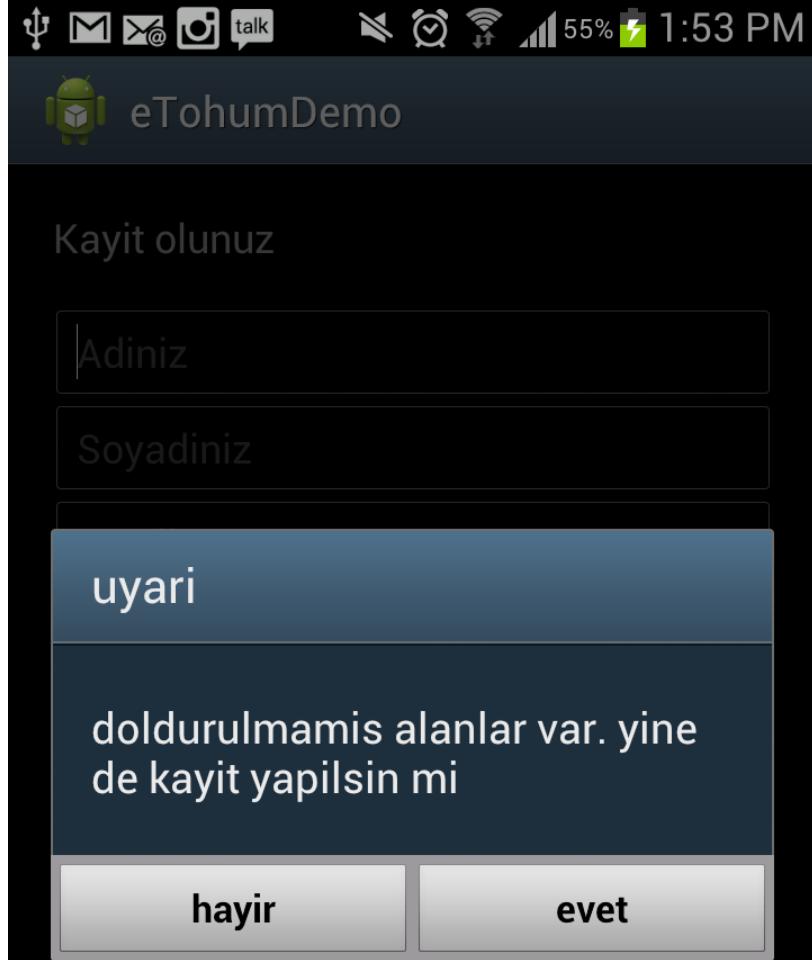
UI - Dialogs

- * Small windows used for notifying user or perform small actions(confirmation, login etc)

```
new AlertDialog.Builder(this)
    .setTitle ("uyari")
    .setMessage ("doldurulmamis alanlar var. yine de kayit yapilsin mi" )
    .setCancelable (false)
    .setPositiveButton ("evet", new DialogInterface.OnClickListener () {
        public void onClick(DialogInterface dialog, int id) {
            register();
        }
    })
    .setNegativeButton ("hayir", null).show();
```



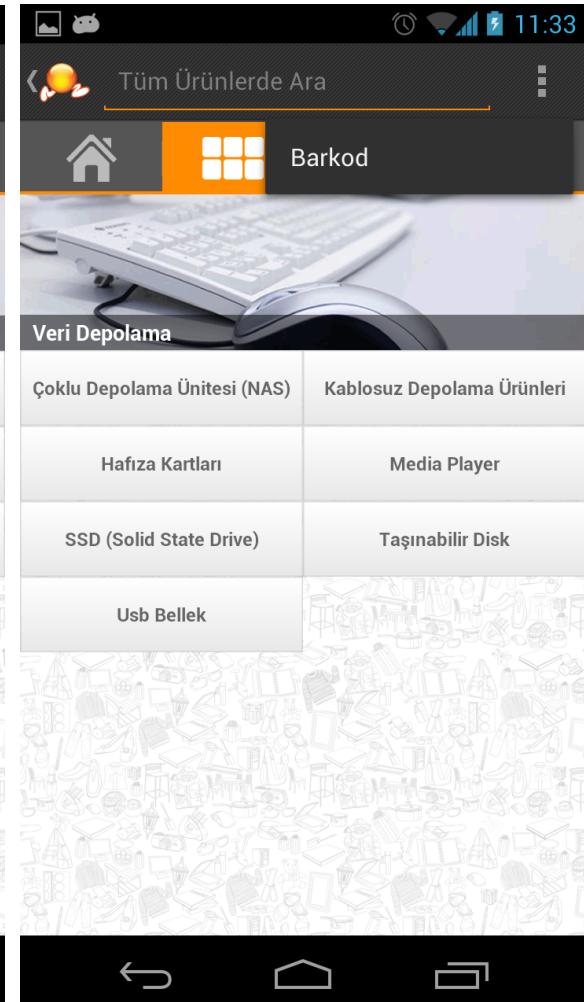
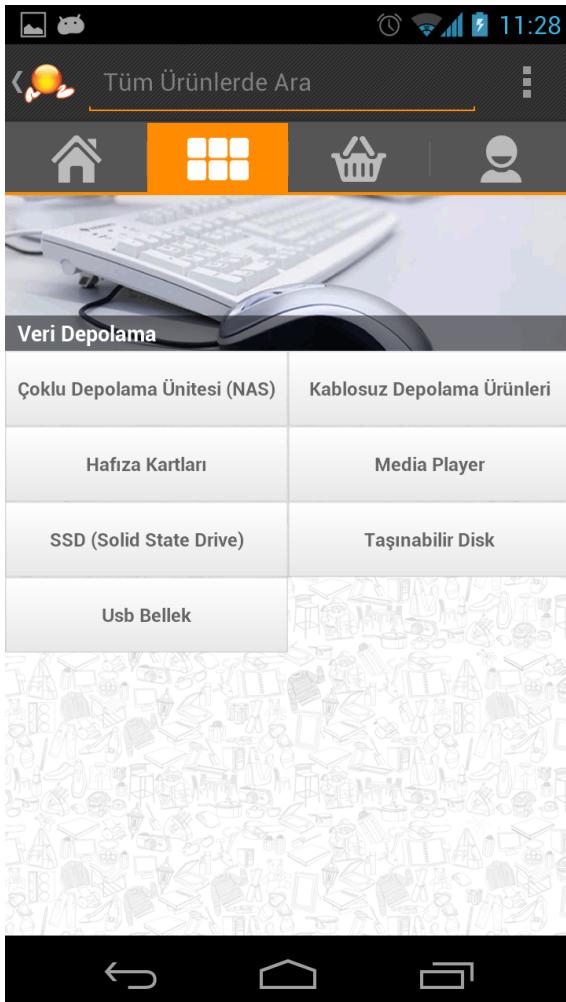
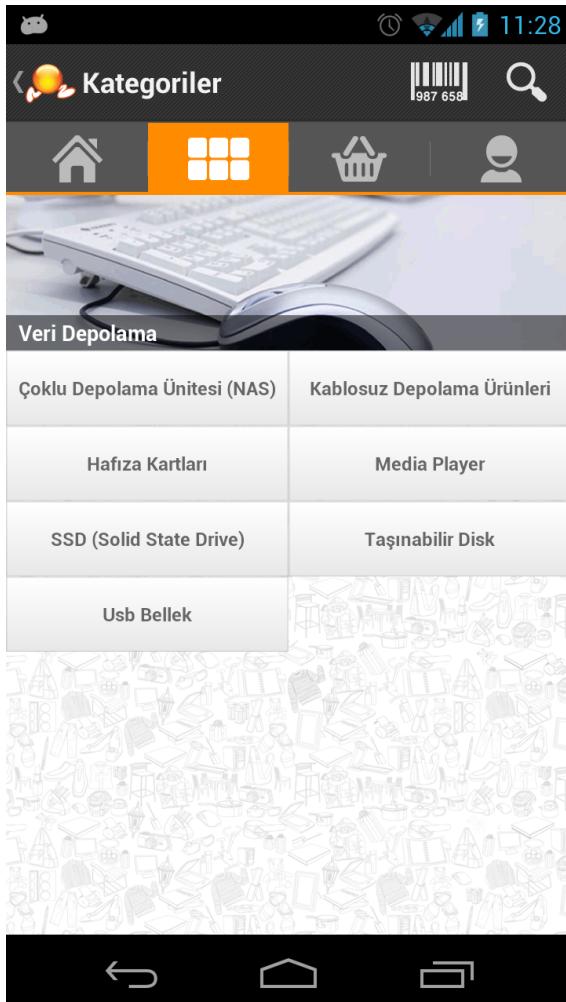
Dialogs Demo



UI - ActionBar

- * Introduced in Honeycomb (API 11)
- * Powerful widget used for navigation and perform user actions
- * Backward compatible via SherlockActionBar:
3rd party library developed by Jake Wharton
<http://actionbarsherlock.com/>
- * Enables standardization between applications.

UI - ActionBar





Notifications @ Android



eTohumDemo

Toast Notification

Simple non-persistent
popup text notifications to
inform user and doesn't
require user response.

```
Toast.makeText(context, context.getString(R.  
string.welcome) + " " + name, Toast.LENGTH_LONG).  
show();
```

Universite Listesi



Ozyegin Universitesi



Galatasaray Universites

BAHÇEŞEHİR
ÜNİVERSİTESİ

Bahcesehir Universitesi



Koc Universitesi

. Sabancı .
Üniversitesi

Sabanci Universitesi

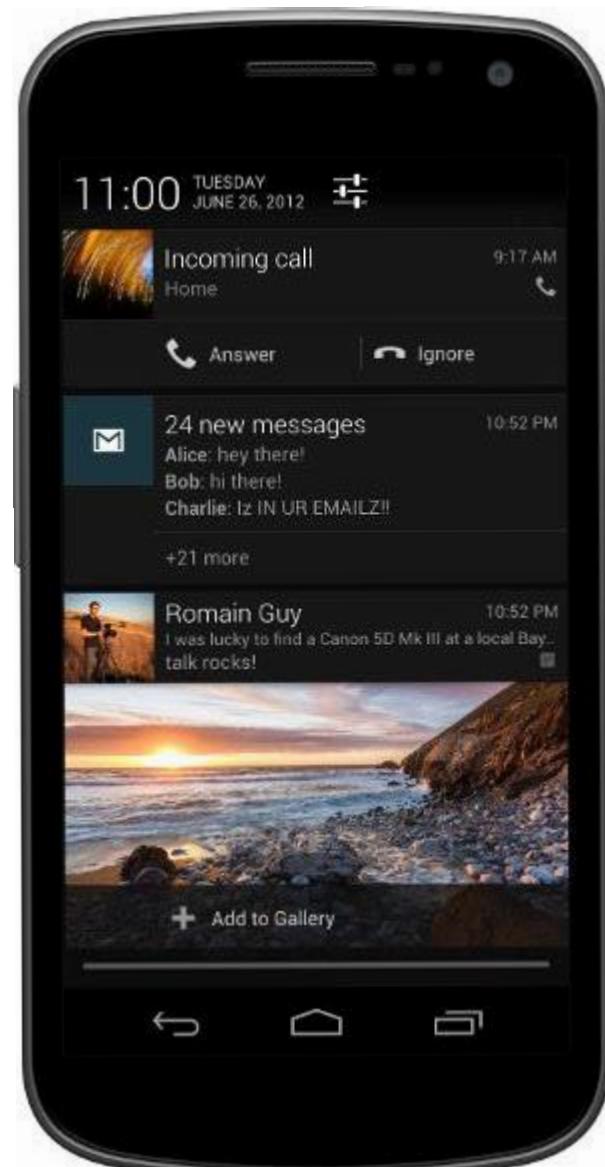
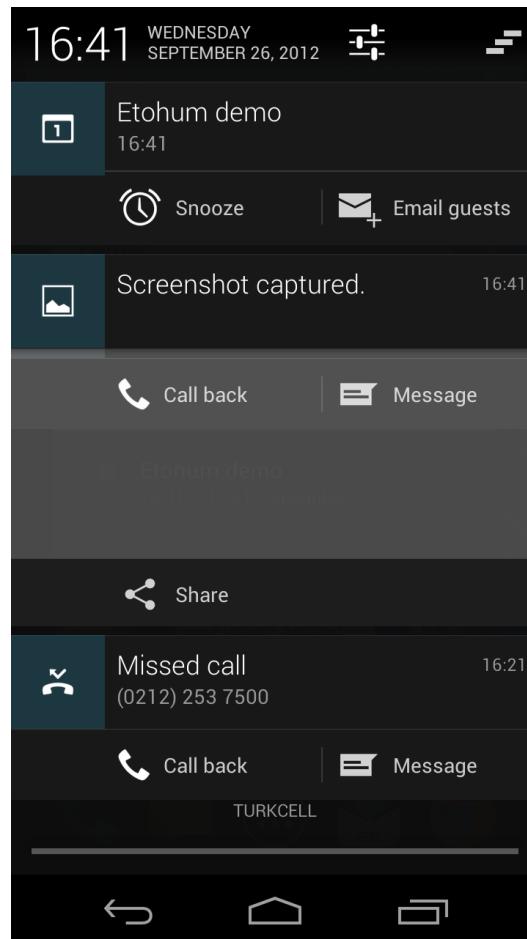
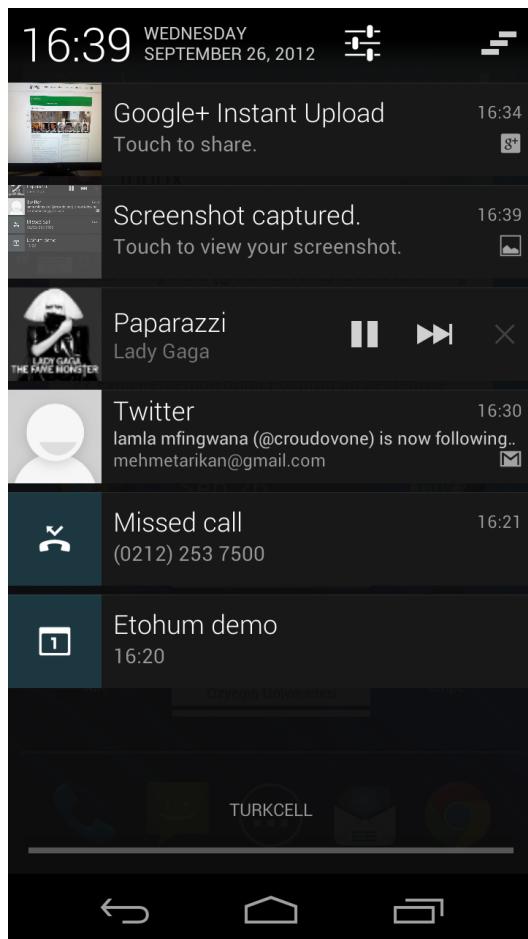
Welcome ismail ceylan

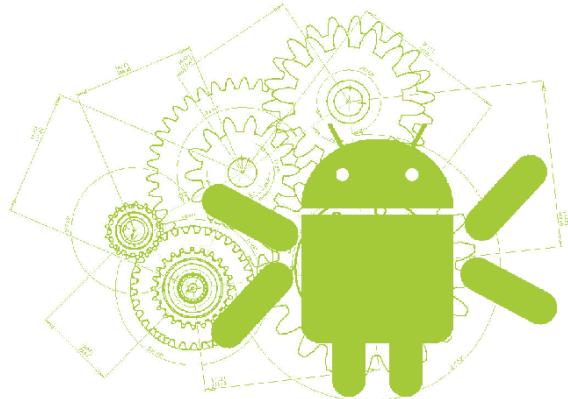
Bogazici Universitesi

Status Notification

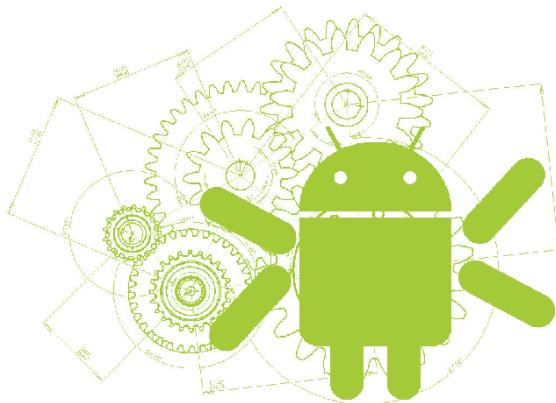
- Adds an icon and ticker message to the status bar.
- Shows a message in the notification window
- Can send intent to application, if user clicks.
- Can perform actions without opening application.

Status Notification





Toast Notification Demo



Supporting Different Devices @ Android

- Language
- Android Versions
- Screens (density, resolution, size)

Resources - Overview

- Externalization of resources (strings, images, animations) from source code.
- Improves maintainability of project.
- Allows easily support different device types

```
MyProject/
  src/
    MyActivity.java
  res/
    drawable/
      icon.png
    layout/
      main.xml
      info.xml
    values/
      strings.xml
```

Resources

- We can provide alternative resources for different devices.

```
res/
    drawable/
        icon.png
        background.png
    drawable-hdpi/
        icon.png
        background.png
```

- **Possible configurations:** screen width, screen height, screen density, screen size, screen orientation, API level, language and region etc...

```
drawable/
drawable-en/
drawable-fr-rCA/
drawable-en-port/
drawable-en-notouch-12key/
drawable-port-ldpi/
drawable-port-notouch-12key/
```

Accessing Resources

- On compile time a *R* class containing id's for resources is generated.
- Can be referenced from both xml and code.

```
<string name="hello">Hello!</string>
```

- **In code :** `R.string.hello`
- **IN XML :** `@string/hello`

```
EditText editText = (ImageView) findViewById (R.id.editText);
editText.setText (R.string.hello);

or

editText.setText (context.getResources.getString (R.string.hello));
```

```
<EditText xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/editText"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:textColor="@android:color/secondary_text_dark"
    android:text="@string/hello" />
```

Supporting Different Devices



Figure 1. Two different devices, each using the default layout (the app provides no alternative layouts).

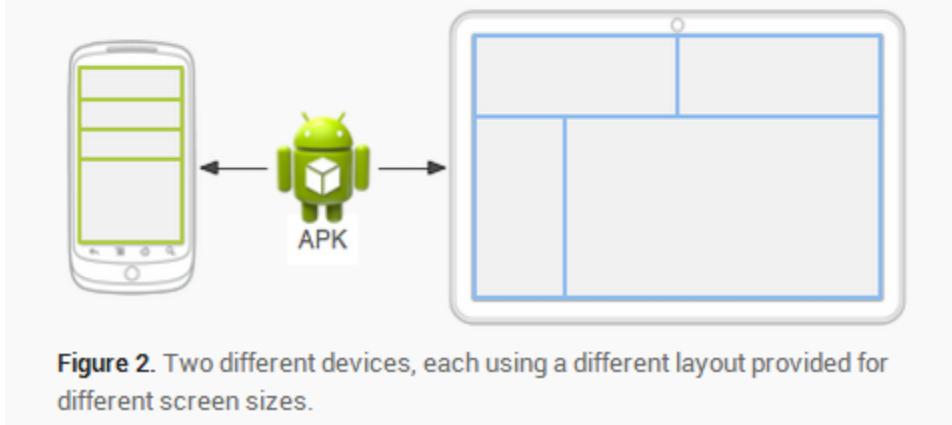


Figure 2. Two different devices, each using a different layout provided for different screen sizes.

Views - XML

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="I am a TextView" />
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="I am a Button" />
</LinearLayout>
```

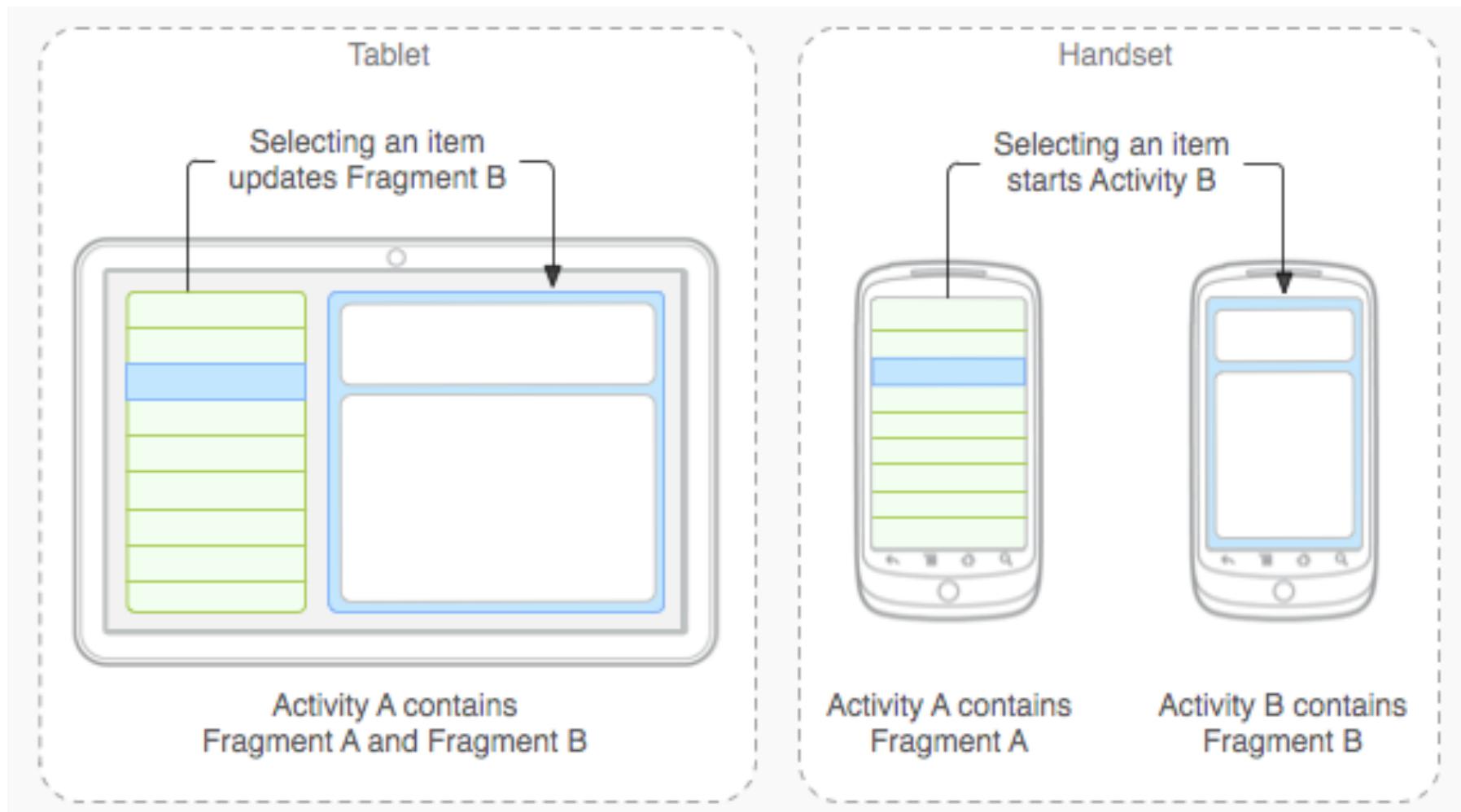
Views

- Common Layouts:
 - Linear Layout
 - Relative Layout
 - List View
 - Grid View

Fragments

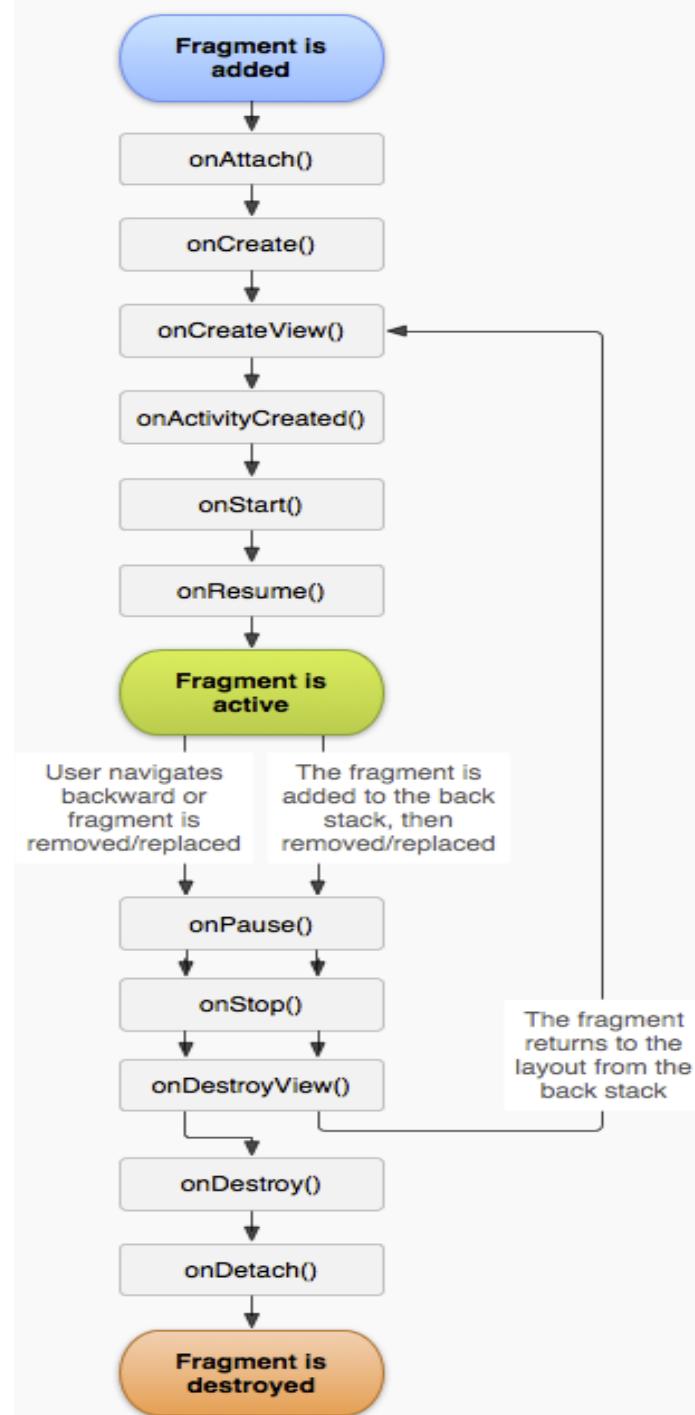
- Reusable UI modules
- Avoid switching between activities; use Fragments instead
- Introduced with Honeycomb
- Dynamic and Flexible UI
- Backward Compatible via Support Library

Fragments



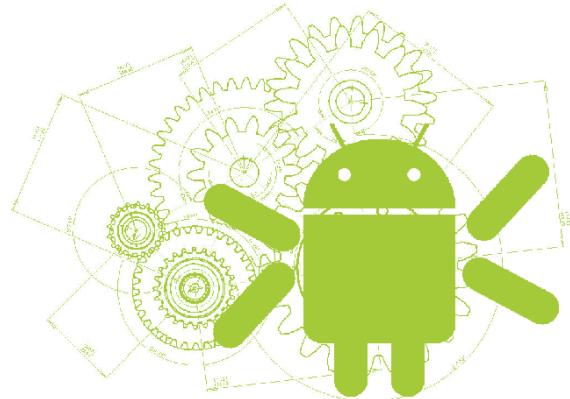
Fragments

- Has its own lifecycle, consumes input events
- Embedded in activities and lifecycle directly affected by the host activity's lifecycle
- Back stack managed by the activity



Android Manifest

- * Presents essential information about the application to the Android system
 - Names the Java package
 - Describes the components:
 - * activities, services, content providers
 - Declares which permission the application must have, the minimum level of the Android API, libraries etc...
 - Declares hardware and software features used



Application Fundamentals Demo

Localization

- Provide resources for each language you want to support. Make sure there is *default* resources as well.

```
res/
    values/
        strings.xml
            <string name="hello">Merhaba!</string>
            <string name="goodbye">Güle güle!</string>
    arrays.xml
values-en/
    strings.xml
        <string name="hello">Hello!</string>
        <string name="welcome">Welcome!</string>
```

Localization

- Uses device's region.
- For custom language :

```
Locale locale = new Locale("en");
Locale.setDefault(locale);
Configuration config = new Configuration();
config.locale = locale;

context.getResources().updateConfiguration(config, context.getResources()
    .getDisplayMetrics());
```

UI - Styling and Theming

- * **Style:** collection of properties specify the look and format of a View.
- * Styles are defined in a separate file from layouts.
- * Can be reused for different views.
- * Can be inherited from other styles
- * Reduces code duplication.

Theme: A Style applied to entire application or activity.

UI - Styles

- * Create an XML file under /res/values directory.
- * Each style should be defined via <style> tag.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="CodeFont" parent="@android:style/TextAppearance.Medium">
        <item name="android:layout_width">fill_parent</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:textColor">#00FF00</item>
        <item name="android:typeface">monospace</item>
    </style>
</resources>

<style name="GreenText" parent="@android:style/TextAppearance">
    <item name="android:textColor">#00FF00</item>
</style>
<style name="CodeFont.Red">
    <item name="android:textColor">#FF0000</item>
</style>
```

UI - Styles

- * Now we can use our styles.

```
<TextView  
    android:layout_width="fill_parent"  
    android:layout_height="wrap_content"  
    android:textColor="#00FF00"  
    android:typeface="monospace"  
    android:text="@string/hello" />
```

is reduced to

```
<TextView  
    style="@style/CodeFont"  
    android:text="@string/hello" />
```

and can be applied to other TextView's.

UI - Theme

- * Similar to styles

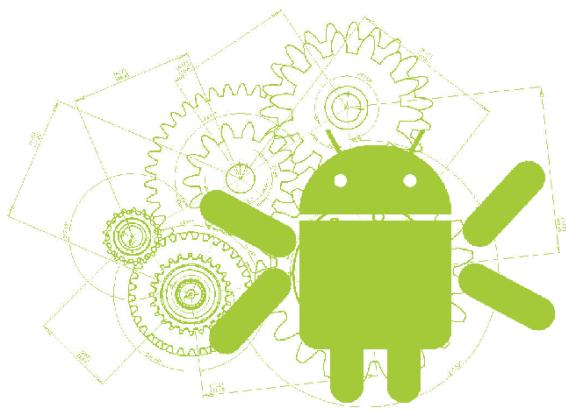
```
<color name="custom_theme_color">#b0b0ff</color>
<style name="CustomTheme" parent="android:Theme.Light">
    <item name="android:windowBackground">@color/custom_theme_color</item>
    <item name="android:colorBackground">@color/custom_theme_color</item>
</style>
```

but should be applied to application or activity
in manifest file

```
<application android:theme="@style/CustomTheme">
```

or

```
<activity android:theme="@android:style/Theme.Translucent">
```



Threads @ Android

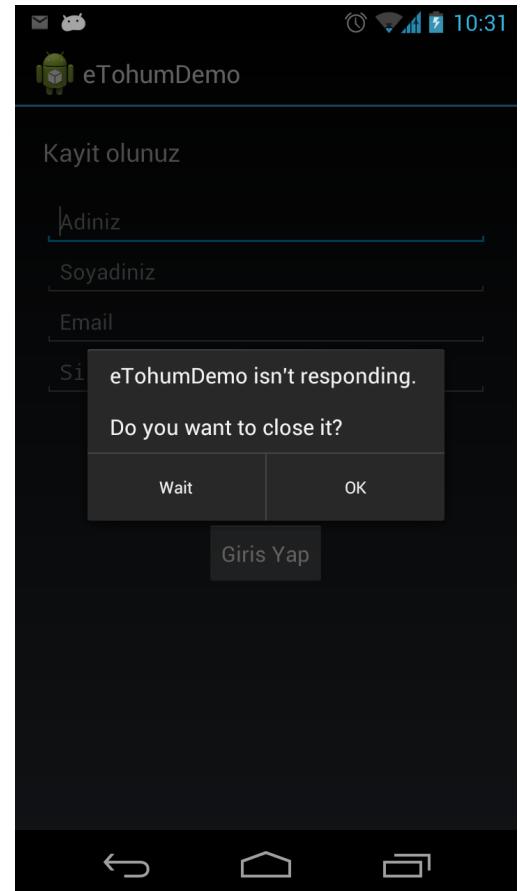
Asynchronous Tasks

Do **NOT** block UI (main) thread

- * ANR(Application not responding) :
 - No response to user input within 5 seconds
 - A BroadcastReciever hasn't finished executing in 10 seconds

- * Conventional Solution: Threads
- * Android approach: **AsyncTasks**

- * Runs on background and notifies you when you need to update the UI.



Asynchronous Tasks

```
private class DownloadFilesTask extends AsyncTask<URL, Integer, Long> {
    protected Long doInBackground(URL... urls) {
        int count = urls.length;
        long totalSize = 0;
        for (int i = 0; i < count; i++) {
            totalSize += Downloader.downloadFile(urls[i]);
            publishProgress((int) ((i / (float) count) * 100));
            // Escape early if cancel() is called
            if (isCancelled()) break;
        }
        return totalSize;
    }

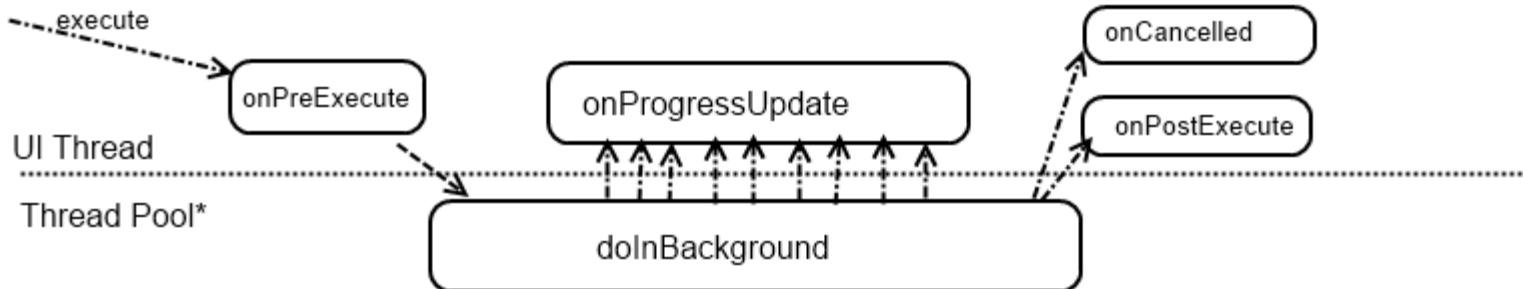
    protected void onProgressUpdate(Integer... progress) {
        setProgressPercent(progress[0]);
    }

    protected void onPostExecute(Long result) {
        showDialog("Downloaded " + result + " bytes");
    }
}

new DownloadFilesTask().execute(url1, url2, url3);
```

Asynchronous Tasks

- Workflow of AsyncTask :



* When first introduced, `AsyncTasks` were executed serially on a single background thread. Starting with [DONUT](#), this was changed to a pool of threads allowing multiple tasks to operate in parallel. Starting with [HONEYCOMB](#), tasks are executed on a single thread to avoid common application errors caused by parallel execution.

* For parallel execution `executeOnExecutor(java.util.concurrent.Executor, Object[])` with [`THREAD_POOL_EXECUTOR`](#).

Network Connection

```
try {
    URL url = new URL(myurl);
    HttpURLConnection conn = (HttpURLConnection) url.openConnection();
    conn.setReadTimeout(10000 /* milliseconds */);
    conn.setConnectTimeout(15000 /* milliseconds */);
    conn.setRequestMethod("GET");
    conn.setDoInput(true);
    // Starts the query
    conn.connect();
    int response = conn.getResponseCode();
    Log.d(DEBUG_TAG, "The response is: " + response);
    is = conn.getInputStream();

    // Convert the InputStream into a string
    String contentAsString = readIt(is, len);
    return contentAsString;

    // Makes sure that the InputStream is closed after the app is
    // finished using it.
} finally {
    if (is != null) {
        is.close();
    }
}
```



Lists and Adapters

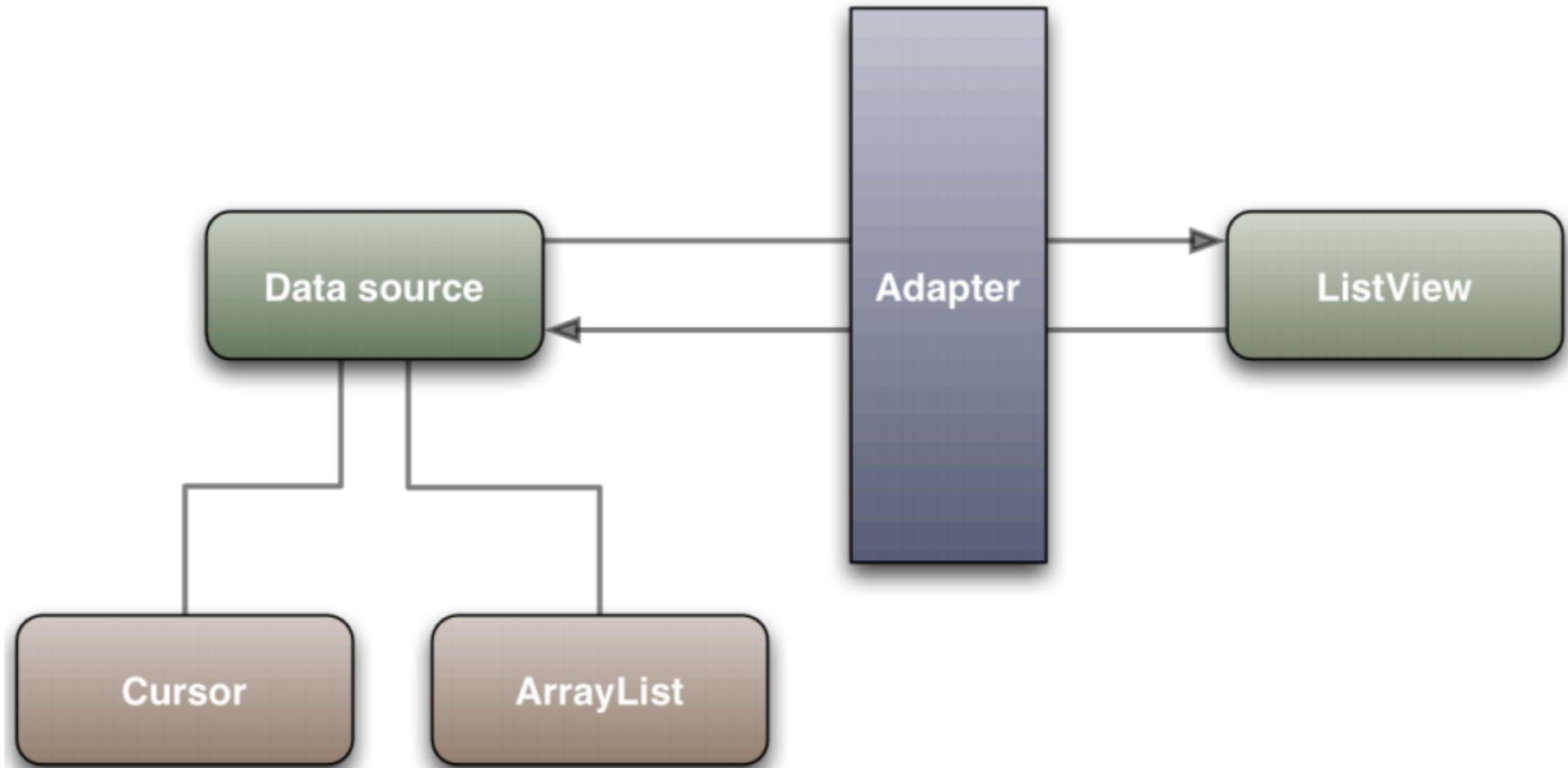
ListView
Adapters

ListView

- * View group to display a list of scrollable items
- * Large data set - performance
- * Handling data changes
- * *¿ Smooth scrolling*
- * *¿ Fetching data efficiently*



Adapters - man in the middle



Adapters

- * Terminology
 - index: Child views
 - position: Data in Adapter
 - id: Unique identifier for data

- Local view cache
- Accessing views from the adapter
- Change convertView's structure
- Assumptions about getView calls

Adapters

`getView(int position, View convertView,
ViewGroup parent)`

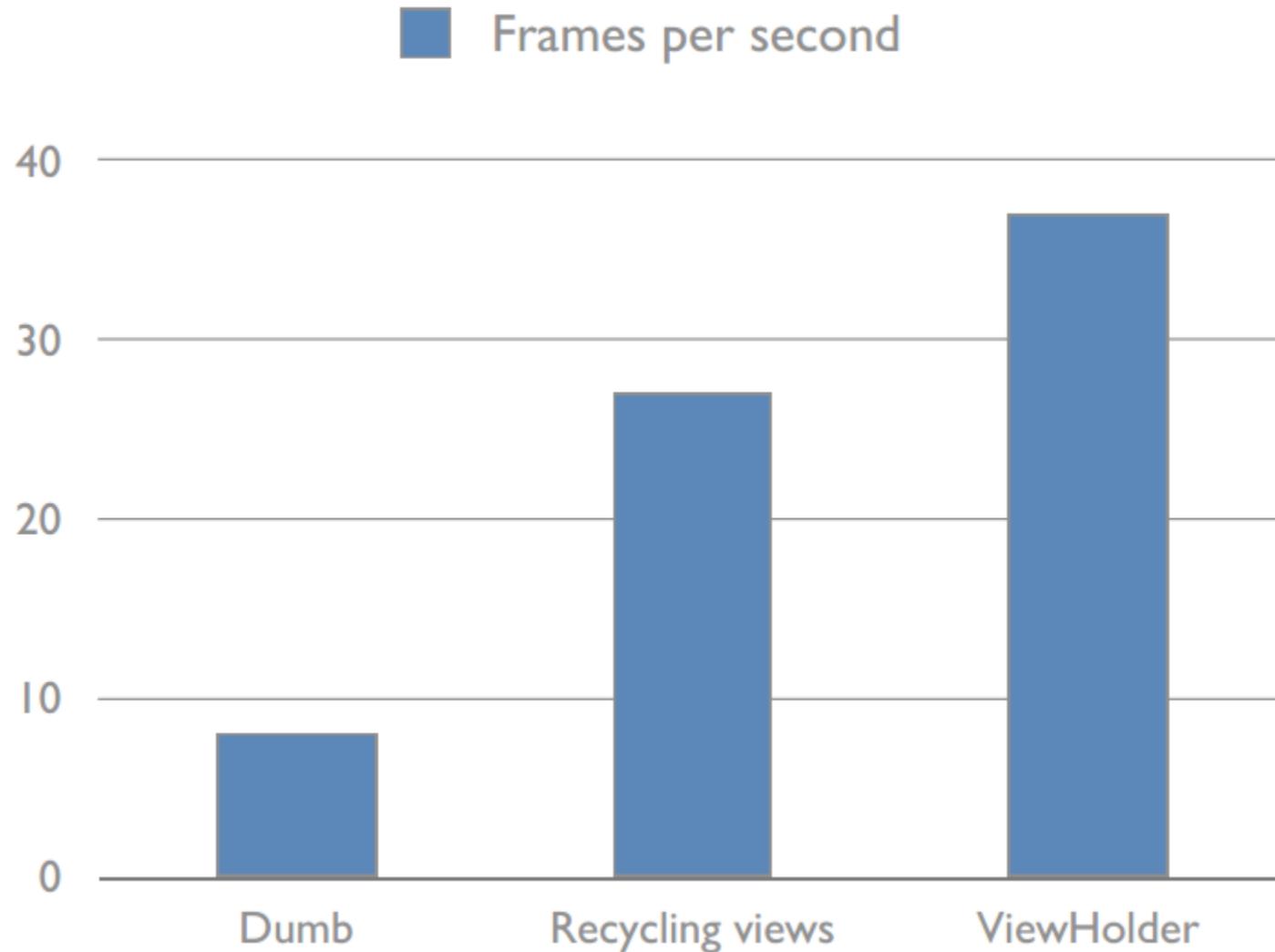
- Full data presentation control
- Optimization

- * `ListView` is smart
- * `convertView`
 - Supplied by `ListView`
 - Matches item types
- Reuse it

Adapters - DON'T

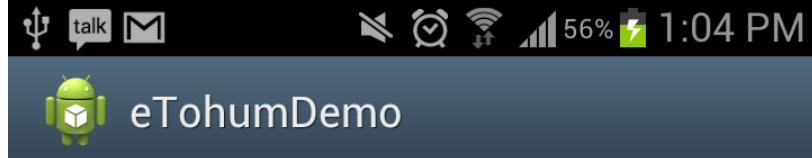
```
1 public View getView(int position, View convertView, ViewGroup parent) {  
2     if (convertView == null) {  
3         convertView = mInflater.inflate(R.layout.item, parent, false);  
4     }  
  
5     ((TextView) convertView.findViewById(R.id.text)).setText(DATA[position]);  
6     ((ImageView) convertView.findViewById(R.id.icon)).setImageBitmap(  
7             (position & 1) == 1 ? mIcon1 : mIcon2);  
  
8     return convertView;  
9 }
```

Performance





ListView Demo



Universite Listesi



Ozyegin Universitesi



Galatasaray Universitesi



Bahcesehir Universitesi

Koc Universitesi



Sabanci Universitesi

Bogazici Universitesi



Location @ Android

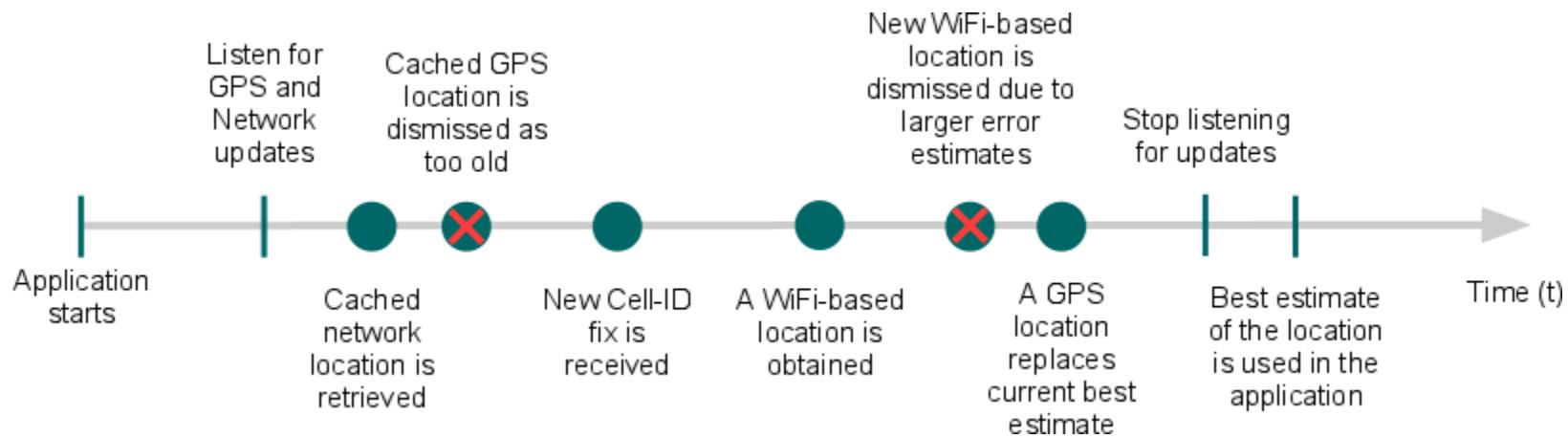
Location Strategies

- * GPS
 - *pros:* most accurate
 - *cons:* only works outdoors, consumes battery, and slow

- * Android Network Location Provider
 - *pros:* fast, less battery consumption
 - *cons:* uses cell towers and Wi-Fi signals

Challenges !

- * Multitude of location sources
 - trade-offs (gps, cell-id, Wi-Fi)
- * User movement
 - people walks, drives....
- * Varying accuracy



Requesting Location Updates

```
// Acquire a reference to the system Location Manager
LocationManager locationManager = (LocationManager) this.getSystemService
(Context.LOCATION_SERVICE);

// Define a listener that responds to location updates
LocationListener locationListener = new LocationListener() {
    public void onLocationChanged(Location location) {
        // Called when a new location is found by the network location provider.
        makeUseOfNewLocation(location);
    }

    public void onStatusChanged(String provider, int status, Bundle extras) {}

    public void onProviderEnabled(String provider) {}

    public void onProviderDisabled(String provider) {}
};

// Register the listener with the Location Manager to receive location updates
locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0, 0,
locationListener);
```

lastKnownLocation:

```
LocationProvider locationProvider = LocationManager.NETWORK_PROVIDER;
// Or use LocationManager.GPS_PROVIDER

Location lastKnownLocation = locationManager.getLastKnownLocation(locationProvider);
```



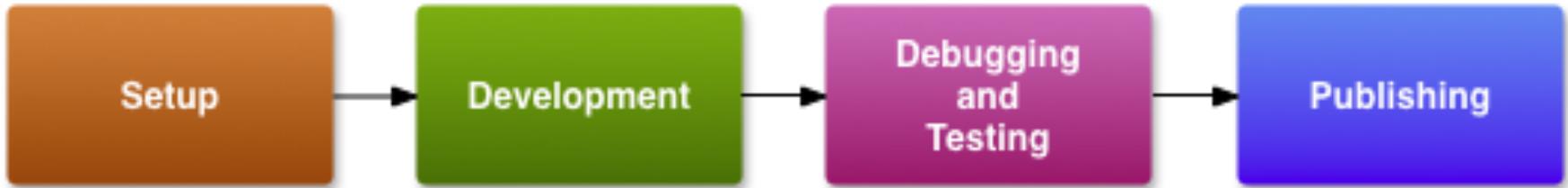
Data Storage @ Android

Data Storage

- Shared Preferences
 - Store private primitive key-value pairs
- Internal/External Storage
 - Store private data in internal storage
 - Public data in external storage
- SQLite
 - Private database
 - SQLite 3
- Data Backup
 - Backup data to cloud and restore when application is reinstalled



Publishing the Application



Publishing the Application

- All applications must be signed
- Special debug key created by SDK
- Self signed certificates; no certificate authority needed



We are hiring !
ik@pozitron.com

**Android / Java
iOS / Objective-C
Windows Phone / C#**

**Backend / Java, Scala, Python
Web Frontend / HTML5, JavaScript**



You can download
slides and demos @

[https://github.
com/cnlms/SabanciWorksho](https://github.com/cnlms/SabanciWorkshop)

p

Rotten Android Project

- Movie stuff (reviews, cast info, ratings, critics etc.)
- Accessible over a public API
- <http://developer.rottentomatoes.com/docs>