Modulo 3

Ecosistema Android

3.1

Introduzione ad Android

3.1 Cos'è Android

- Sistema operativo open-source inizialmente pensato per dispositivi mobili e poi evolutosi.
- Attualmente leader del mercato

  ...e lo sarà per molto tempo, secondo le stime.

3.1.1 Storia

- Android Inc. fondata nel 2003 da Andy Rubin con l'obiettivo di creare:

  "smarter mobile devices that are more aware of its owner's location and preferences"

- Inizialmente sviluppato in segreto e già con problemi di budget nel primo anno di vita
- Passa poco tempo e Google acquisisce la società (17 agosto 2005)
  - Rubin comincia a lavorare con kernel Linux

3.1.1 Storia (2)

- L'OHA *(Open Handset Alliance)* si svela al mondo il 5 novembre 2007
  - vi facevano parte diversi operatori di telefonia, di dispositivi e di SoC
  - ufficialmente il consorzio sviluppa standard aperti da utilizzare nel mondo mobile
- Quello stesso giorno viene annunciato il progetto Android
  - Framework per dispositivi mobili
  - Linux kernel 2.6

3.1.1 Storia (3)

- L'anno successivo viene presentato il primo prodotto, l'HTC Dream (22 ottobre 2008)
  - Android 1.0 "Apple Pie"
  - Android Market
  - Notifiche
  - Google Apps
  - Media Player
  - Camera
  - WiFi
  - Bluetoon

3.1.1 Storia (4)

- Da quel momento lo sviluppo è prorompente
- Prime versioni con una vera diffusione sul mercato
  - 1.5 *"Cupcake"* (13 aprile 2009)
  - 1.6 *"Donut"* (16 settembre 2009)

- Ogni 6 mesi circa una nuova versione
  - attualmente 4.2 *"Jelly Bean"*

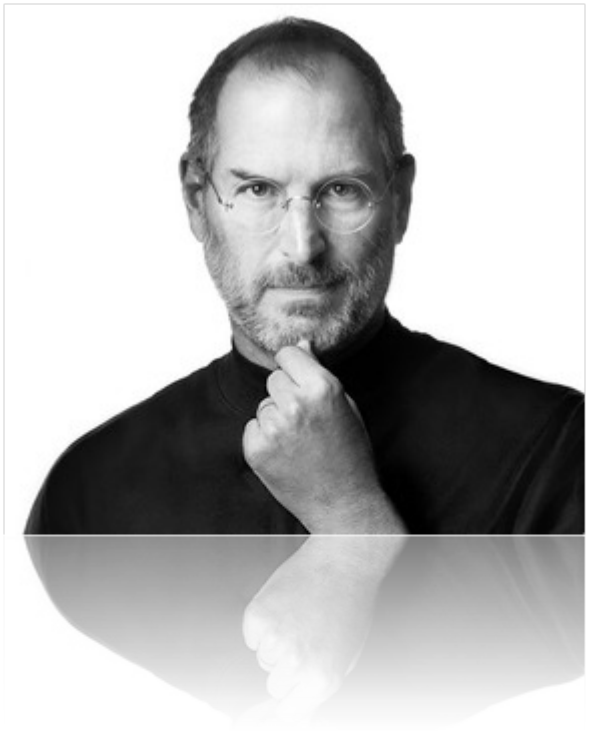
3.1.1 Storia (5)

3.1.1 Storia (6)

- Dal 2010 Google ha lanciato una sua serie di dispositivi Nexus sviluppati da partner e con una esperienza d'uso Android pura
  - HTC One
  - Samsung Nexus S
  - Samsung Galaxy Nexus
  - Asus Nexus 7
  - LG Nexus 4
  - Samsung Nexus 10

3.1.2 Progetto open-source

- Ottobre 2010, **Steve Jobs**:



„Google likes to characterize Android as open and iOS as closed. We think this is disingenuous. […] The real difference is integrated versus fragmented."

3.1.2 Progetto open-source

@Arubin
Andy Rubin

the definition of open: "mkdir android ; cd android ; repo init -u git://android.git.kernel.org/platform/manifest.git ; repo sync ; make"

32 minutes ago via web ☆ Favorite ↻ Retweet ↩ Reply

3.1.2 Progetto open-source

- Il codice di Android è disponibile sotto licenze "*free*" e "*open source*".

- Linux kernel (Open Handset Alliance)
  - GNU GPL v2, pubblico.

- Android system (Google)
  - Rete, telefonia, market, apps.
  - Apache License v2.0, privato.

3.1.2 Progetto open-source

- Google Play disponibile solo su licenza.
- Honeycomb incident (2011).
- Licenza incompatibile con GPL (FSF).

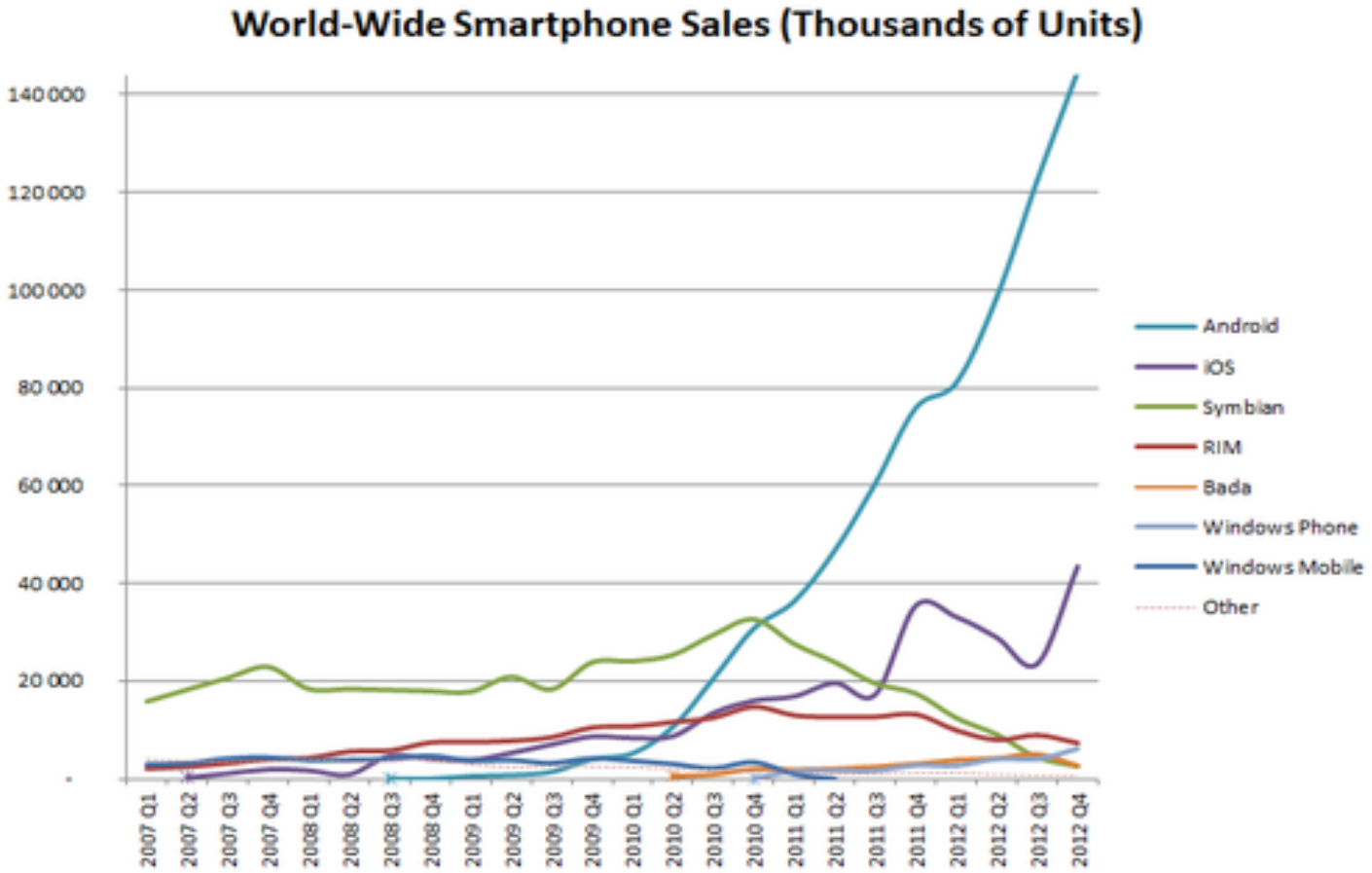
3.1.3 Diffusione

- Esclusi Apple (per ovvi motivi) e Nokia (a stretto contatto con Microsoft), tutti i principali produttori di smartphone si affidano ad Android, dalla fascia bassa ai modelli di punta.

- Ma nel mercato il riscontro sarà stato effettivamente così positivo?

3.1.3 Diffusione

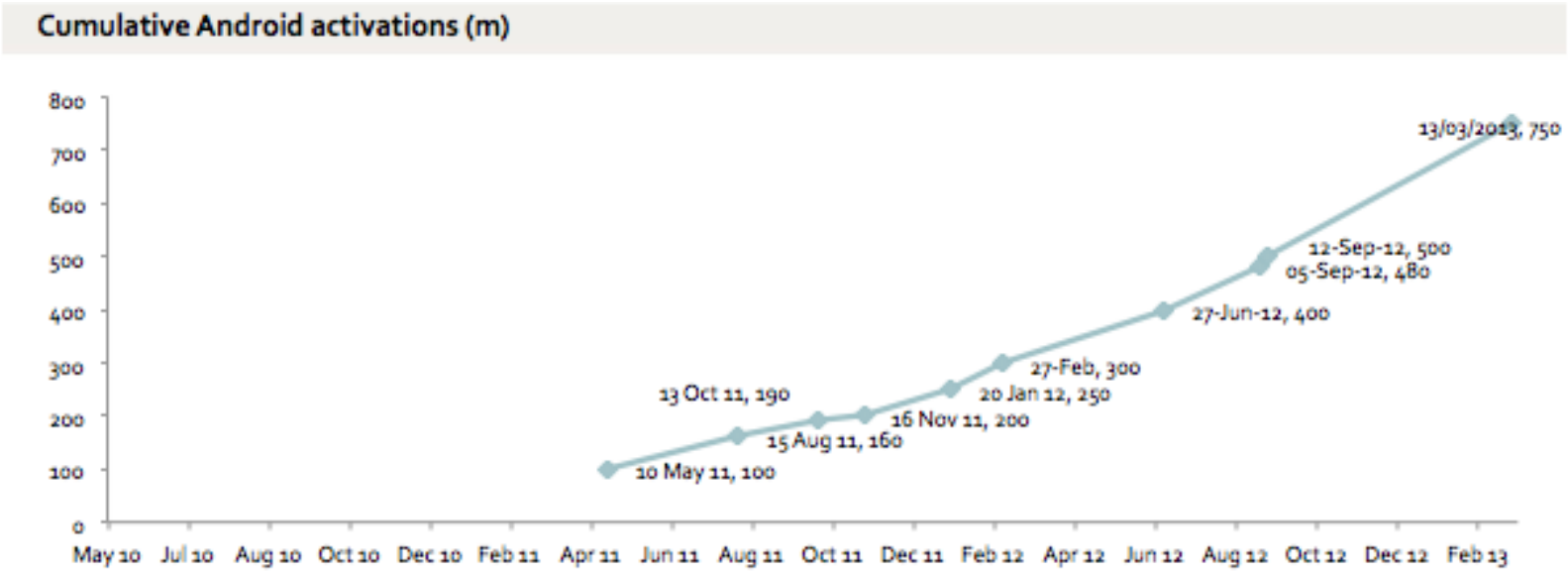
A quanto pare sì.



3.1.3 Diffusione

Benedict Evans
@BenedictEvans

Steady, stable growth of 'official' Android: now 750m or so cumulative activations.



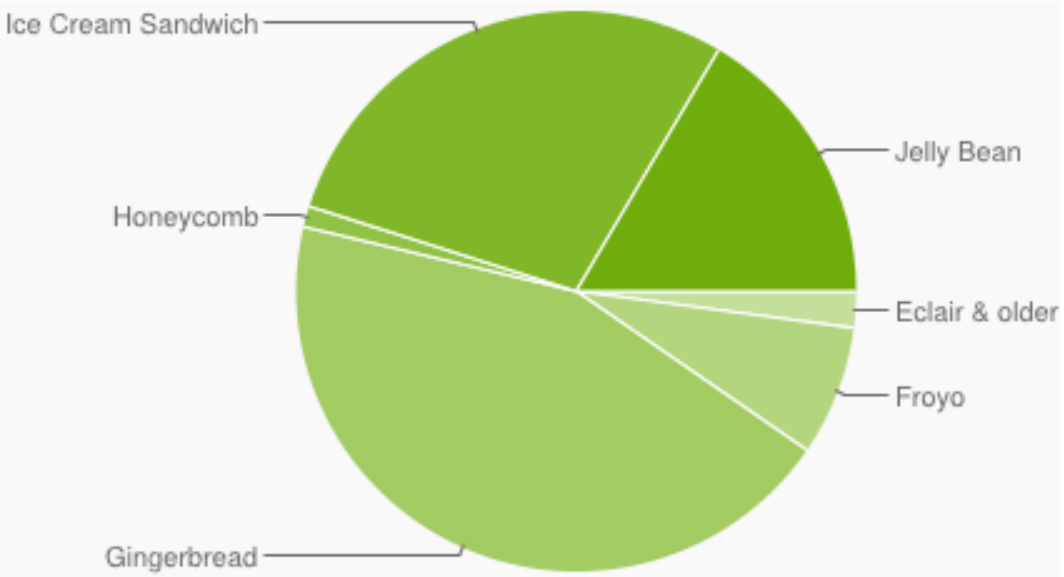
[Source: Google, Enders Analysis]

3.1.3 Diffusione

- Il settore tablet è rimasto appannaggio Apple fino al Q2 2012 (circa)
  - secondo le previsioni della società IDC nel 2013 i tablet Android saranno il 48.8% del mercato contro il 46% di tablet Apple

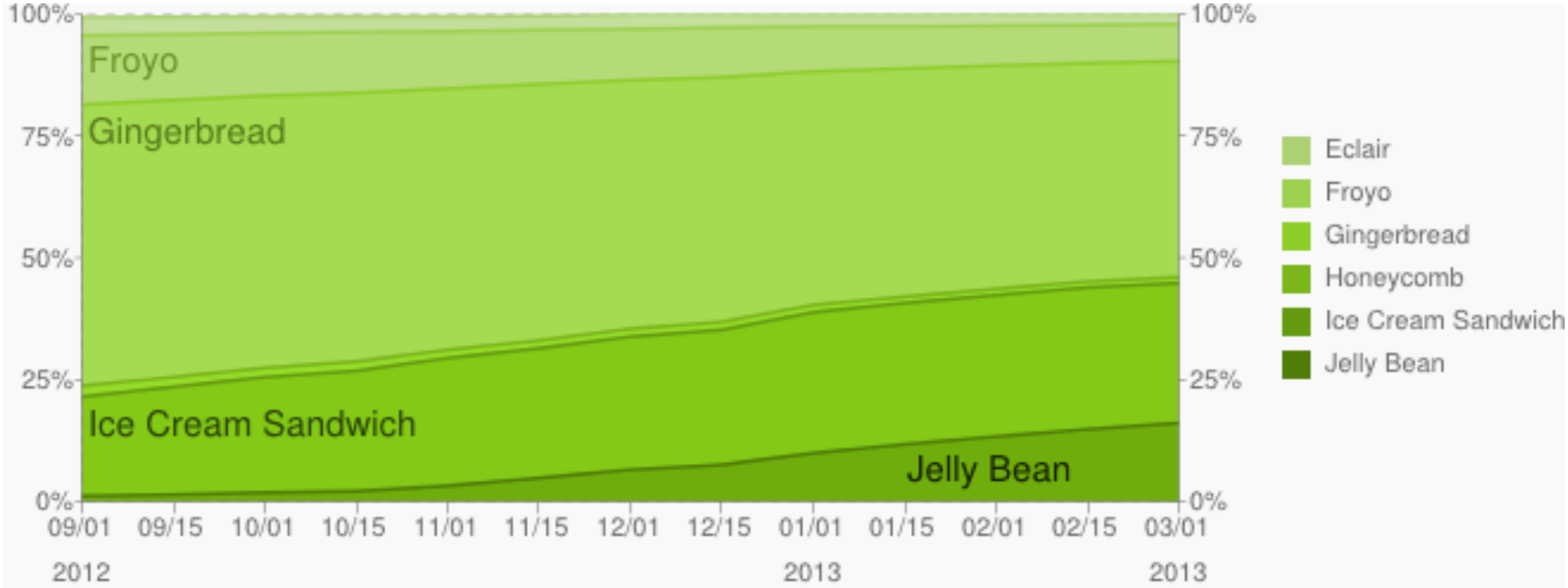
3.1.4 Frammentazione

Version	Codename	API	Distribution
1.6	Donut	4	0.2%
2.1	Eclair	7	1.9%
2.2	Froyo	8	7.5%
2.3 - 2.3.2	Gingerbread	9	0.2%
2.3.3 - 2.3.7		10	43.9%
3.1	Honeycomb	12	0.3%
3.2		13	0.9%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	28.6%
4.1	Jelly Bean	16	14.9%
4.2		17	1.6%



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

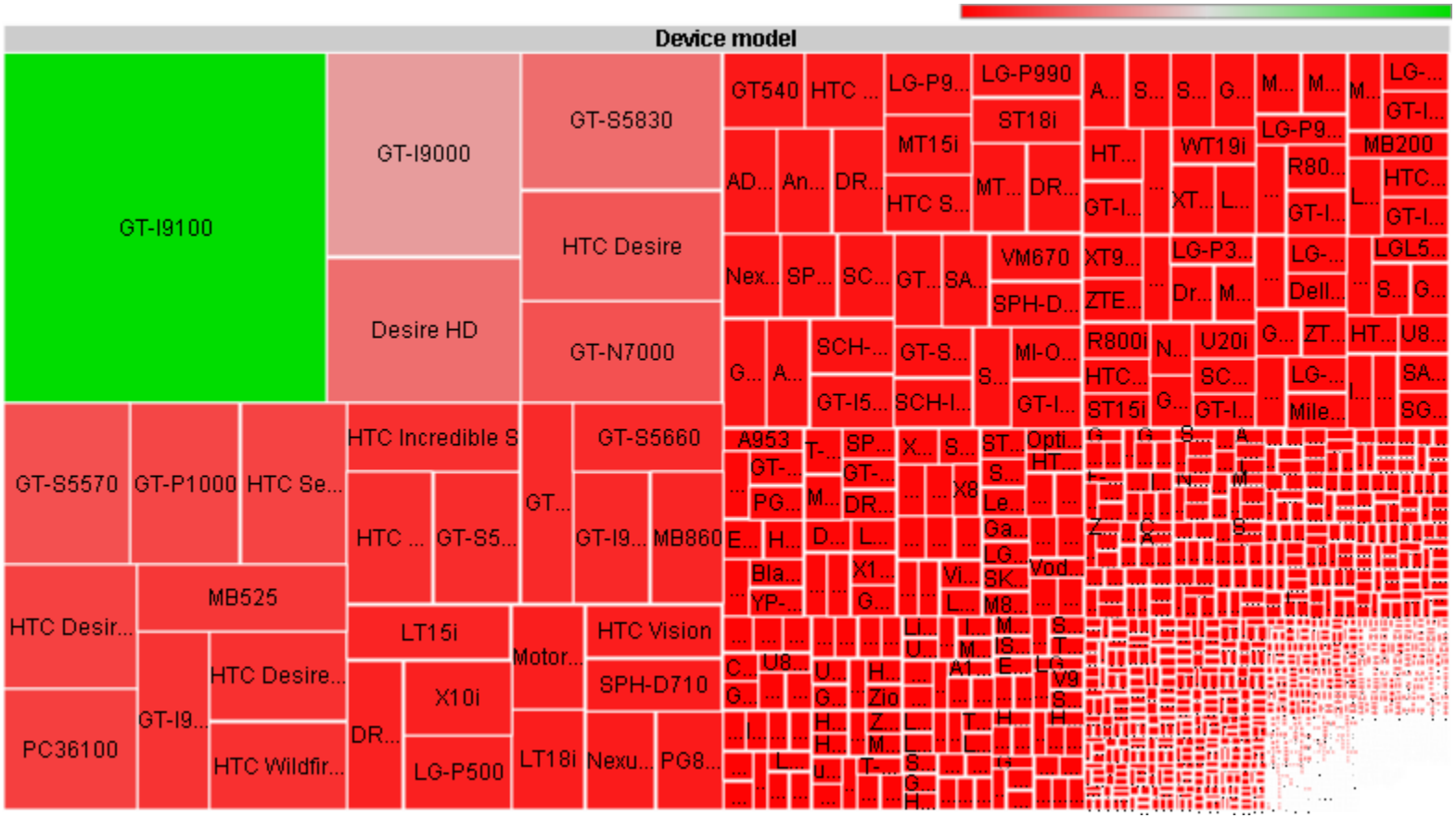
3.1.4 Frammentazione nel tempo



Last historical dataset collected during a 14-day period ending on March 1, 2013

3.1.4 Frammentazione - nello spazio

3.1.4 Frammentazione - infografica



3.1.4 Frammentazione - soluzione?

- Platform Development Kit (Google I/O 2012)
  - Insieme di tools offerti da Google ai produttori per velocizzare il porting dei propri firmware alle nuove versioni del sistema operativo
  - Consegnato 2 mesi prima della pubblicazione delle nuove versioni di Android
  - Politiche commerciali dei vendor?

3.1.5 Prospettive di crescita

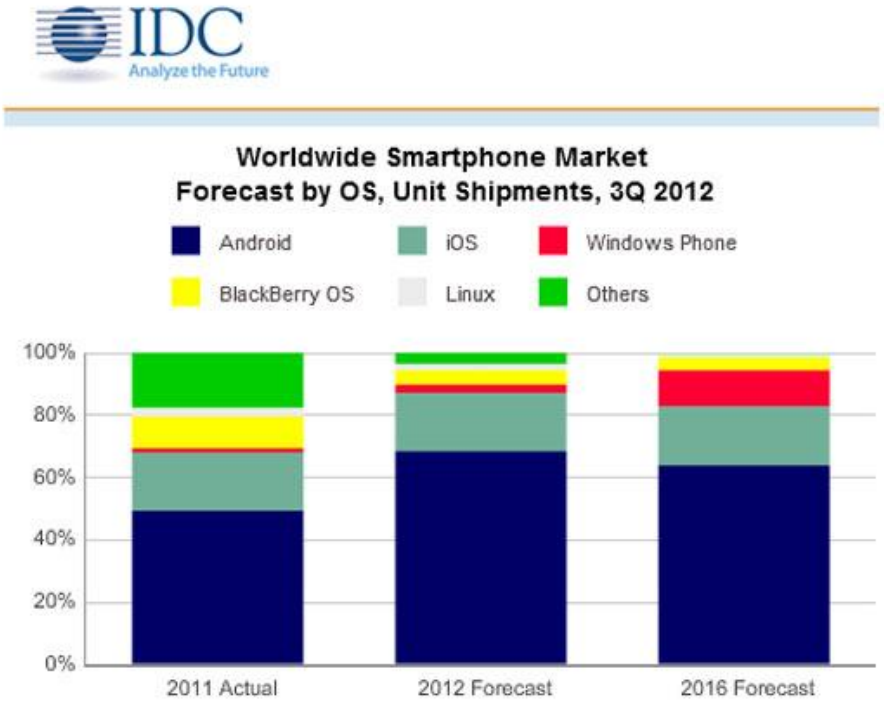
- Per quanto riguarda il settore tablet, sempre secondo IDC, le previsioni al 2017 vedono decrementi sia per iOS che per Android
  - A vantaggio di tablet Windows RT.

Sistema operativo	Quote di mercato 2013	Quote di mercato 2017
Android	48.8%	46.0%
iOS	46.0%	43.5%
Windows	2.8%	7.4%
Windows RT	1.9%	2.7%
Altri	0.6%	0.4%
Totale	100.0%	100.0%

3.1.5 Prospettive di crescita

- Passando al settore smartphone, sempre secondo IDC, le previsioni al 2016 vedono lo stesso scenario

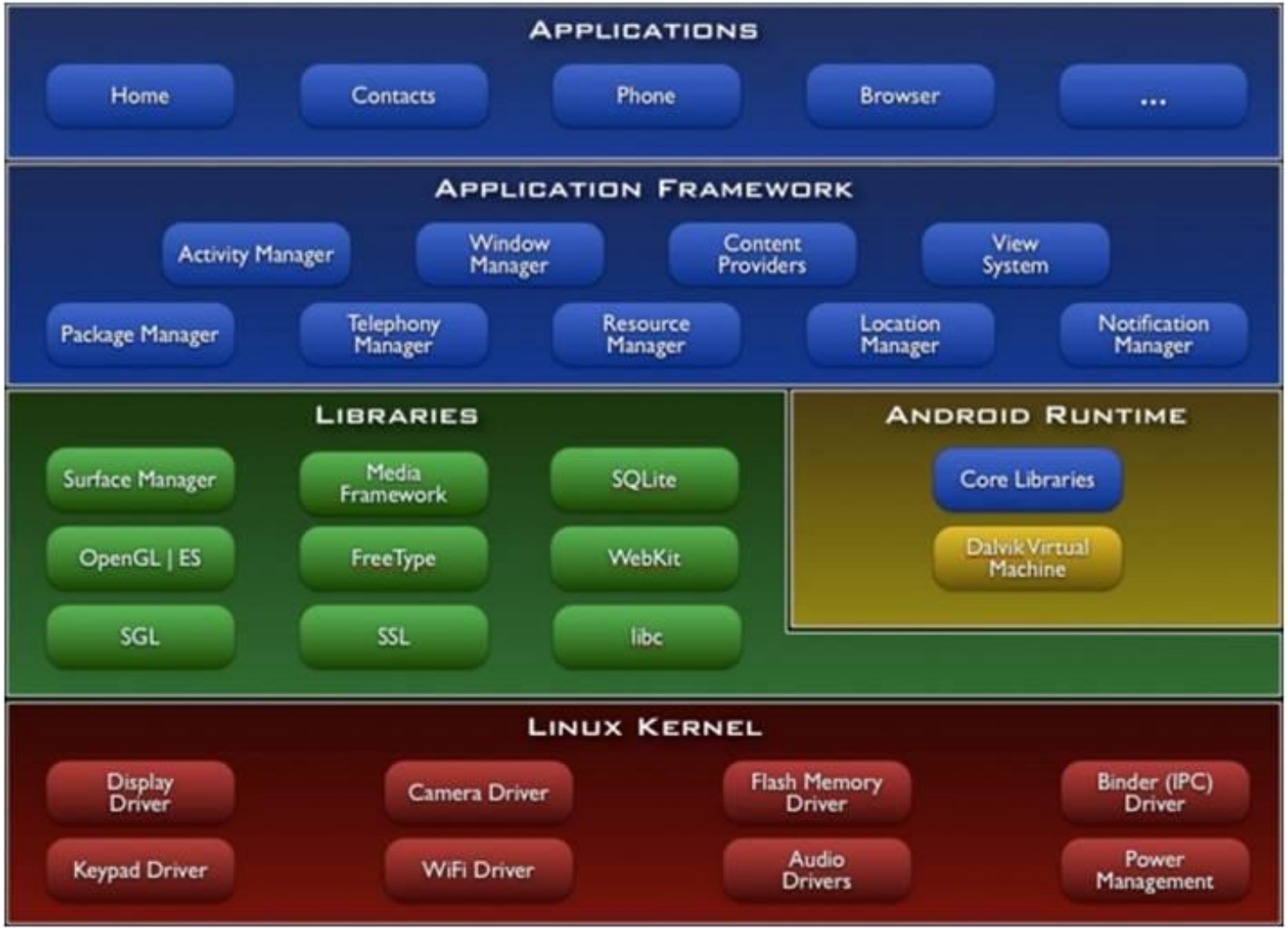
  - Android: 68.3% -> 63.8%
  - iOS: 18.8% -> 19.1%
  - BB: 4.7% -> 4.1%
  - WP: 2.6% -> 11.4%
  - Linux: 2.0% -> 1.5%
  - Altri: 3.6% -> 0.1%



3.1.6 Architettura

- Kernel Linux.
- Librerie native.
- Runtime Android.
  - Virtual Machine Dalvik.
  - Core Libraries.
- Application framework.
- Livello applicazioni.

3.1.6 Architettura



3.1.6 Kernel Linux

- *Kernel* 2.6+.

- Utilizzato per la gestione dei processi e dei *thread*, per la gestione a basso livello della memoria, per lo *stack* di rete, per la sicurezza, per il modello dei *driver*.

3.1.6 Librerie native

Librerie C/C++ eseguite sopra al *kernel* Linux ed utilizzate in diversi ambiti dal sistema Android.

Espongono inoltre funzionalità agli sviluppatori tramite l'*application framework*.

3.1.6 Runtime Android

- Ciò che rende un dispositivo *Android* diverso da un'implementazione mobile *Linux*.
- Tramite l'utilizzo combinato della *Dalvik Virtual Machine* e delle *Core Libraries* rende possibile il funzionamento delle applicazioni Android.

3.1.6 Virtual Machine

- Implementazione software di una macchina (con un certo *Instruction Set Architecture*) che esegue programmi come una macchina reale.
- Virtualization: simulazione di un ambiente in cui *Guest ISA* e *Host ISA* sono gli stessi.
- Emulation: simulazione di un ambiente in cui *Guest ISA* e *Host ISA* sono diversi.

3.1.6 Java Virtual Machine

- Macchina virtuale che esegue programmi scritti in byte-code: rappresentazione intermedia tra il linguaggio macchina e il codice sorgente Java.
- JVM è una specifica di Oracle (*ex* Sun).
- Principali implementazioni:
  - *HotSpot* – OpenJDK (GPL).
  - *HotSpot* – OracleJDK.

3.1.6 Dalvik Virtual Machine

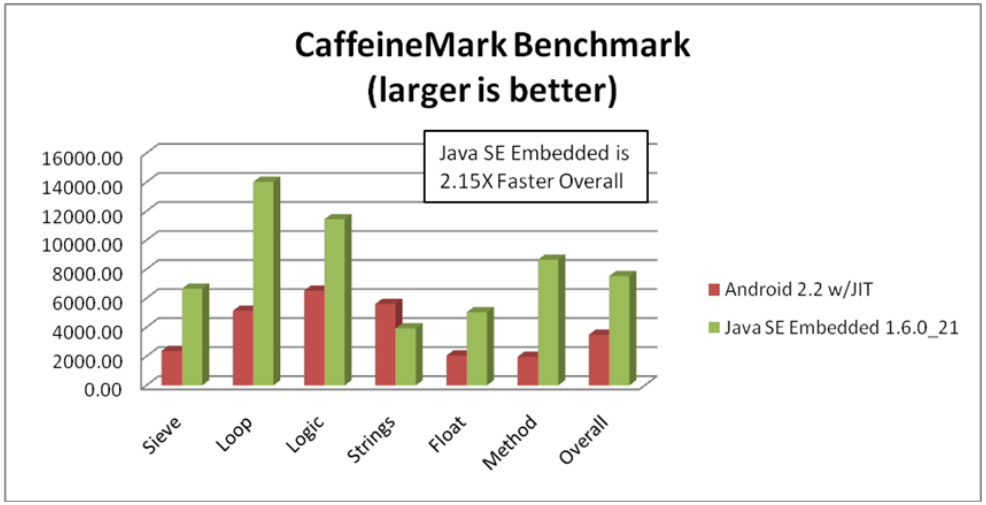
- Macchina virtuale che esegue file .dex (*Dalvik Executable*).
- Sorgente Java compilato in file .class e trasformato in formato .dex.
- Ogni applicazione Android viene eseguita in un processo separato con la propria istanza della Dalvik virtual machine.
- JIT compiler (versione 2.2).

3.1.6 Dalvik vs JVM

Dalvik < JVM.

Ma il problema è multidimensionale.

- Register-based.
- Efficiente con più VMs.
- Footprint minimo in memoria.
- Startup time ridotto.



3.1.6 Core libraries

- Dalvik *non* rispetta la specifica J2SE, né quella J2ME.
- Le core libraries forniscono la maggior parte delle funzionalità disponibili nelle librerie Java standard, così come librerie specifiche per Android.

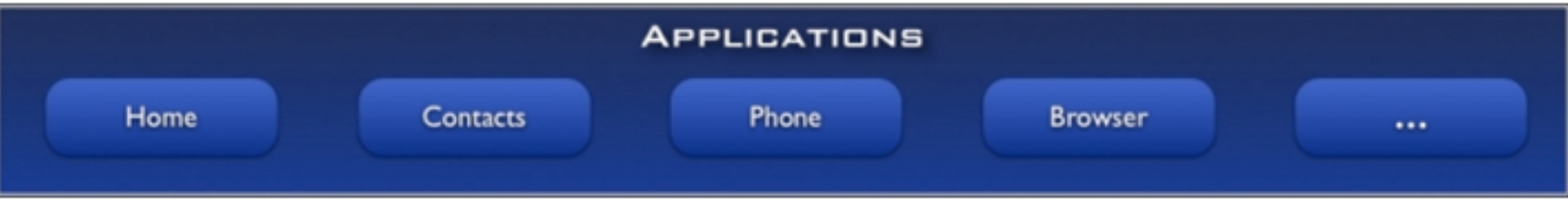
3.1.6 Application framework

- Fornisce servizi importanti, incapsulati in oggetti Java, alle applicazioni di livello sovrastante.
- Sono disponibili alla stessa maniera sia per applicazioni di sistema che per applicazioni di terze parti.

3.1.6 Livello applicazioni

- Applicazioni a livello utente scritte in linguaggio Java.

App di sistema
=
App terze parti



3.1.7 Evoluzione

Anno	Versione	Nome in codice
2008	1.0	Apple Pie
2009	1.1	Petit Four
	1.5	Cupcake
	1.6	Donut
	2.0	Eclair
2010	2.2	Froyo (Frozen Yogour)
	2.3	GingerBread
2011	3.0	HoneyComb
	4.0	IceCream Sandwich
2012	4.1	JellyBean
	4.2	JellyBean
2013	5.0	KeyLime Pie?

3.1.7 Cupcake 1.5 API 3

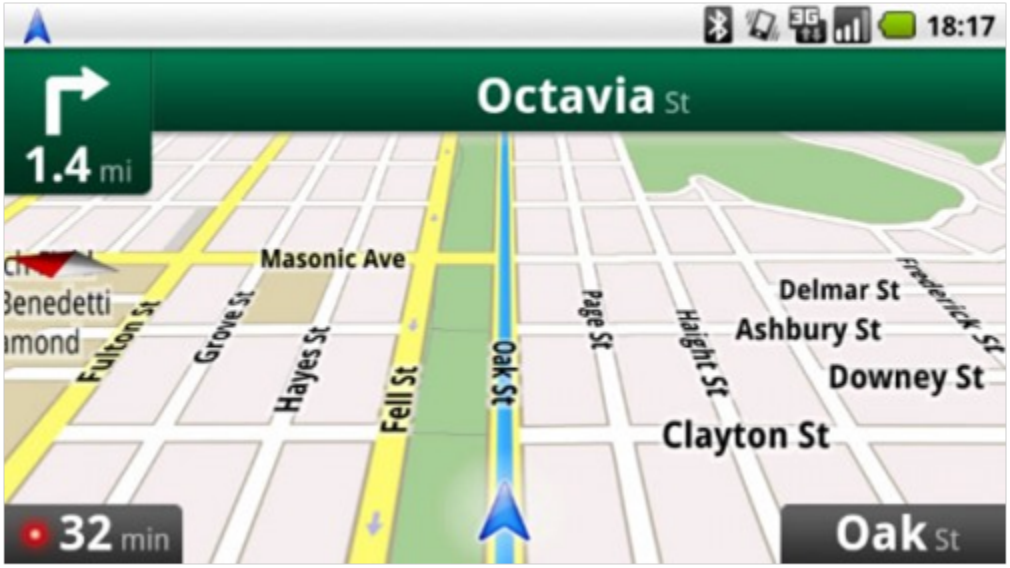
- Riproduzione e cattura audio/video.
- Riconoscimento vocale.
- Soft keyboard
- Widget.
- Notifiche.
- WebView.

3.1.7 Donut 1.6 API 4

- Gestures.
- Sintesi vocale.
- Supporto CDMA.
- Supporto per dimensioni e densità di schermi variabili.

3.1.7 Eclair 2.1 API 7

- Mappe.
- Navigazione
- Supporto Bluetooth 2.1.
- Live wallpapers.



3.1.7 Froyo 2.2 API 8

- JIT compiler.
- Supporto installazione in SD card.
- Miglioramento del riconoscimento vocale ed apertura a terze parti.
- Supporto alla tecnologia Flash.

3.1.7 GingerBread 2.3 API 9/10

- Supporto nativo a Voip (SIP).
- Supporto a NFC.
- Garbage Collector concorrente.
- Supporto ad altri formati multimediali.

3.1.7 HoneyComb 3.0 API 11

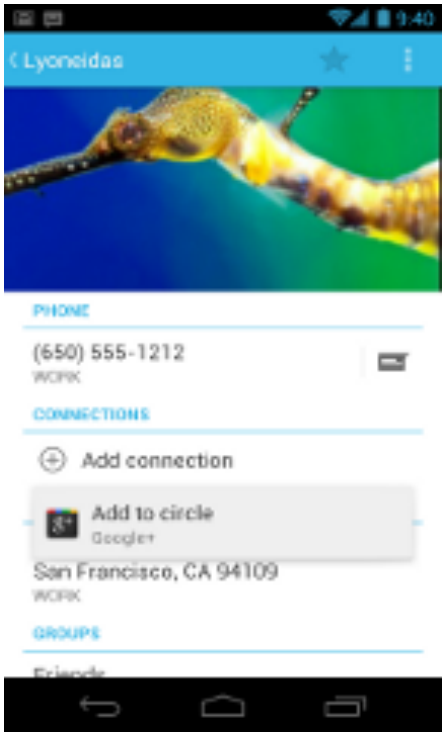
- Fragments.
- ActionBar.
- Nuove API per il drag and drop.
- Supporto per l'accelerazione HW 2D.
- Supporto multimediale migliorato.
- Stile digitale.

3.1.7 HoneyComb 3.1 API 12

- Supporto a periferiche USB.
- Widget ridimensionabili.
- Supporto multimediale migliorato.

3.1.7 IceCream Sandwich 4.0 API 14

- Merge tra versione tablet e smartphone.
- Linux kernel 3.x.
- UI riprogettata (+coerenza).
- Notifiche più potenti.
- Supporto per riconoscimento vocale continuo.
- API sociali.
- API di basso livello streaming.



3.1.7 JellyBean 4.1 API 16

- Project Butter.
- Notifiche espandibili.
- Social API migliorate.
- Riconoscimento vocale avanzato.
- Google Now.
- Abbandono della tecnologia Flash.

3.1.7 JellyBean 4.2 API 17

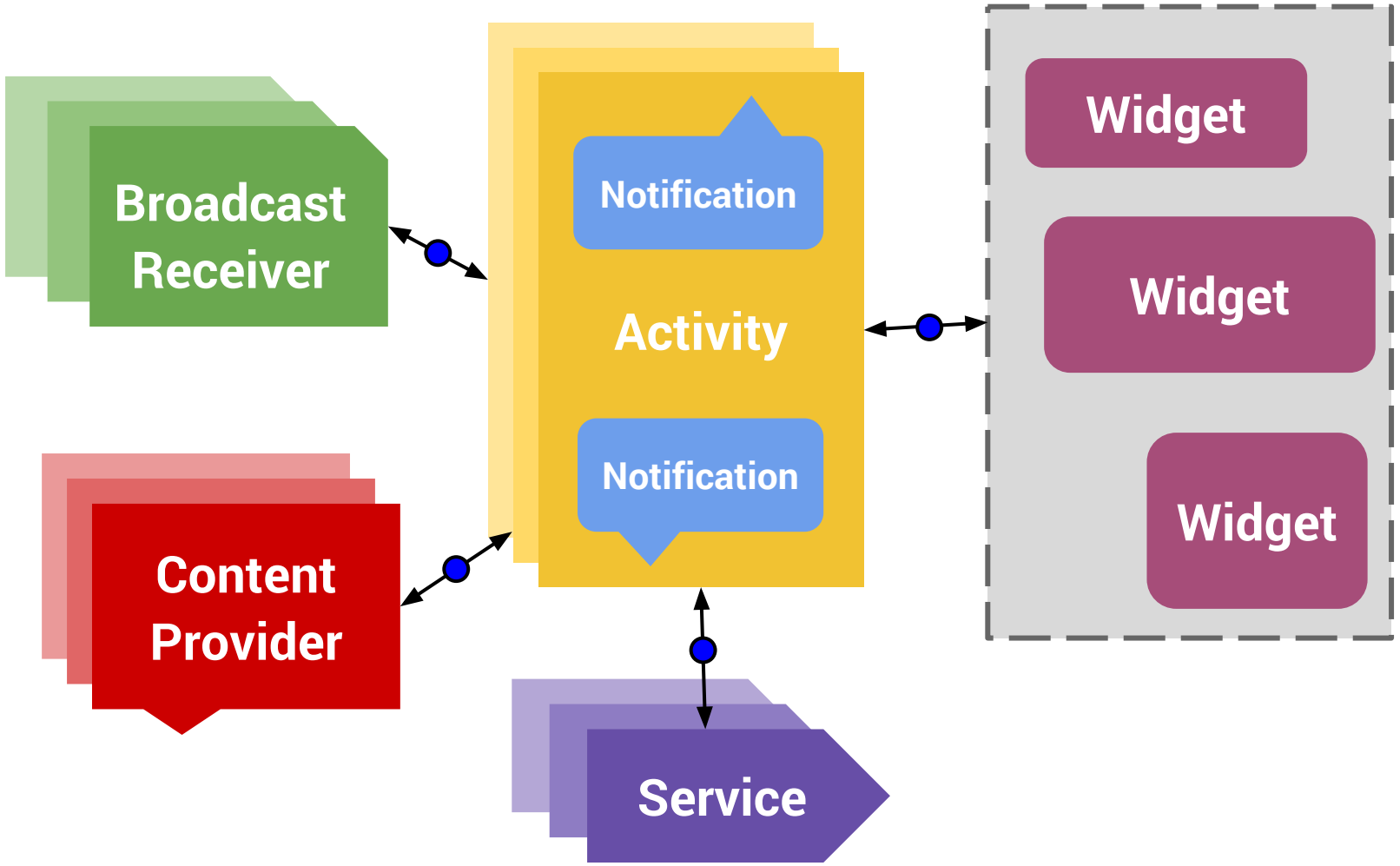
- DayDream.
- Display secondari.
- Lockscreen widgets.
- Supporto per utenti multipli.
- Fragments annidati.

3.2

Elementi principali di un'applicazione Android

3.2 Elementi principali di un'app

3.2 Elementi principali di un'app



3.2 Elementi principali di un'app

Activity

Service

Content Provider

Broadcast Receiver

Intent

Widget

Notification

3.2.1 Elementi principali di un'app

Activity

Service

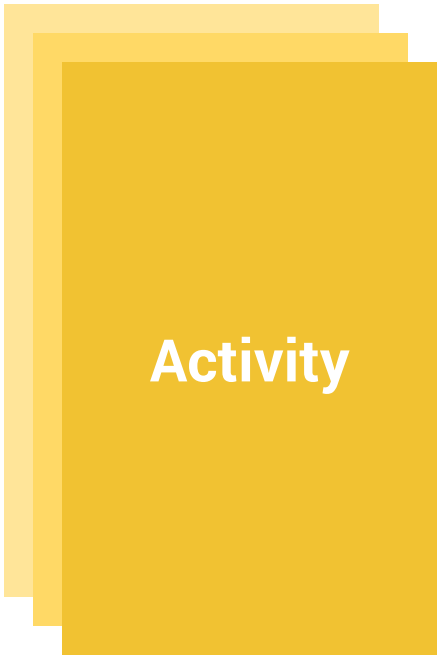
Content Provider

Broadcast Receiver

Intent

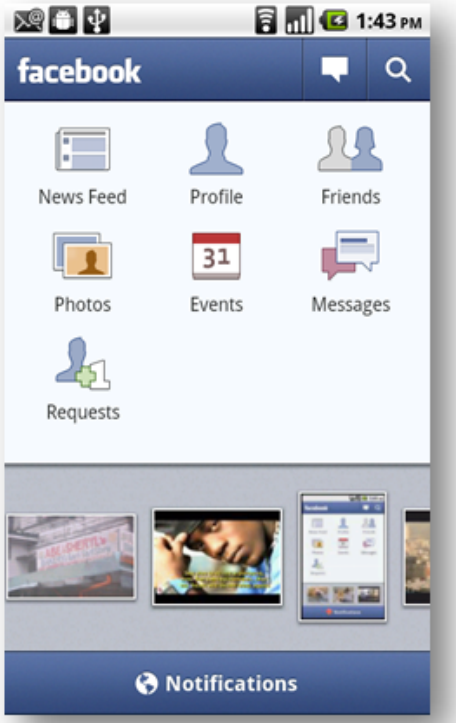
Widget

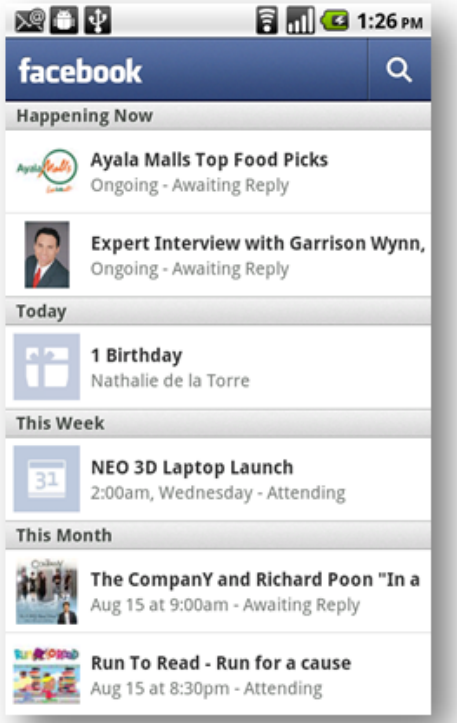
Notification

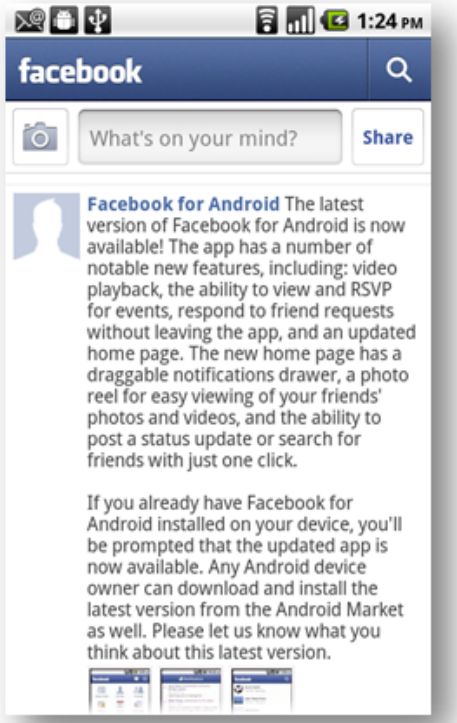


3.2.1 Activity

Ogni Activity rappresenta una **schermata** di una applicazione.

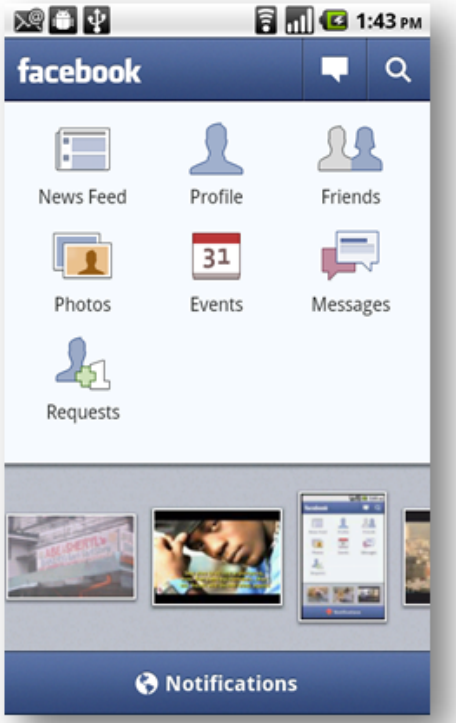


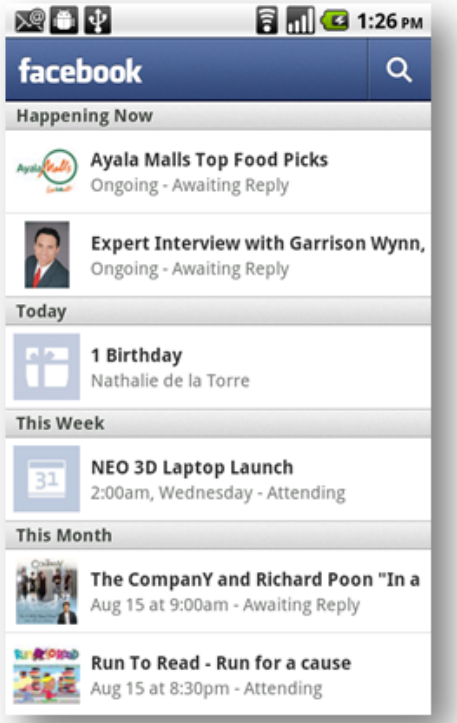


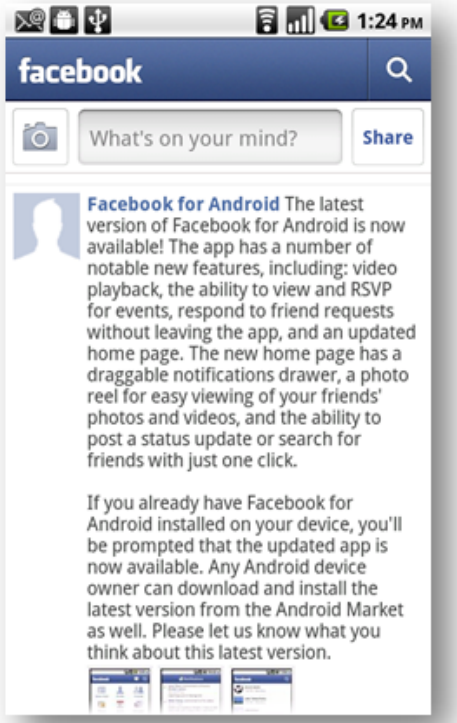


3.2.1 Activity

Ad ogni Activity corrisponde (idealmente) **un'attività atomica**.

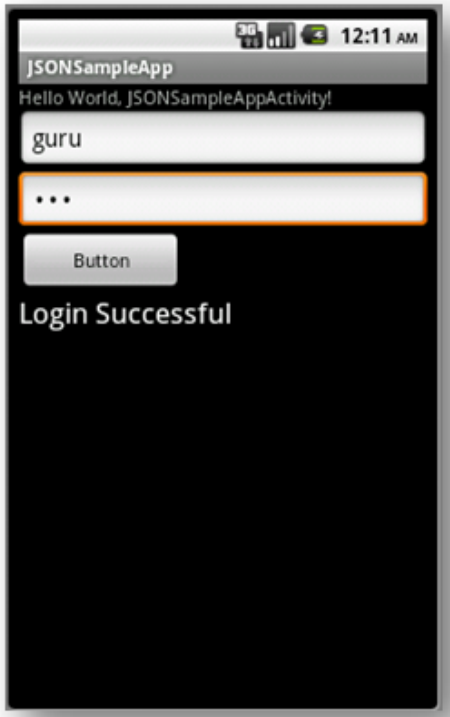


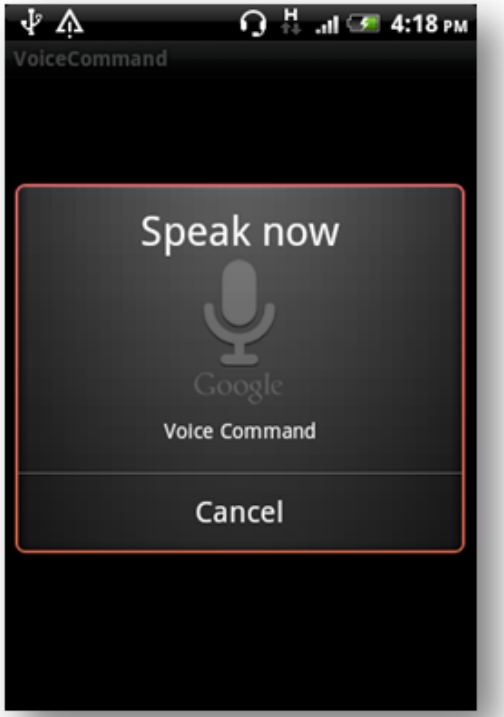




3.2.1 Activity

Le Activity possono avere un valore di ritorno.





3.2.2 Elementi principali di un'app

Activity

Service

Content Provider

Broadcast Receiver

Intent

Widget

Notification

3.2.2 Service

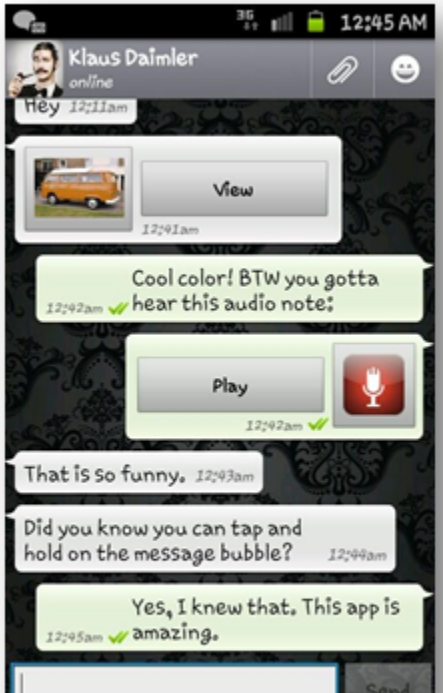
Un’attività dell'app o del sistema invisibile all'utente (gira in background).

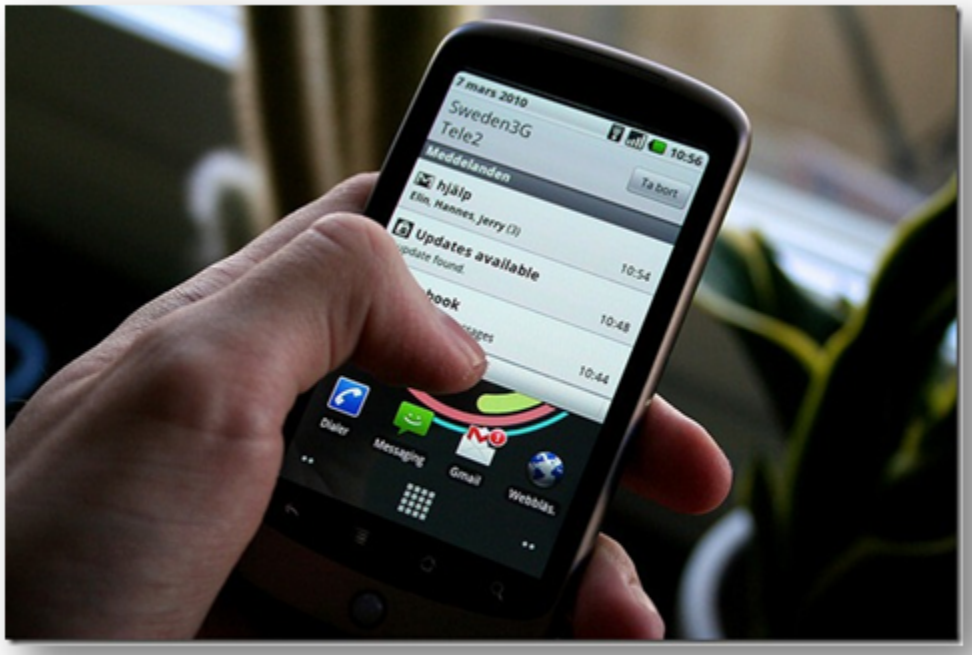
Alcuni esempi:

- Raccoglie dati del sistema.
- Aggiorna le informazioni di una particolare app.
- Scarica nuovi feed RSS da un server.

3.2.2 Service

Non **comunica direttamente con l'utente** ma può comunicare con le activity (anche di altre applicazioni) e con il sistema.





3.2.2 Service

Se fondamentali per un'applicazione posso essere istanziati come *foreground service*.

3.2.3 Elementi principali di un'app

Activity

Service

Content Provider

Broadcast Receiver

Intent

Widget

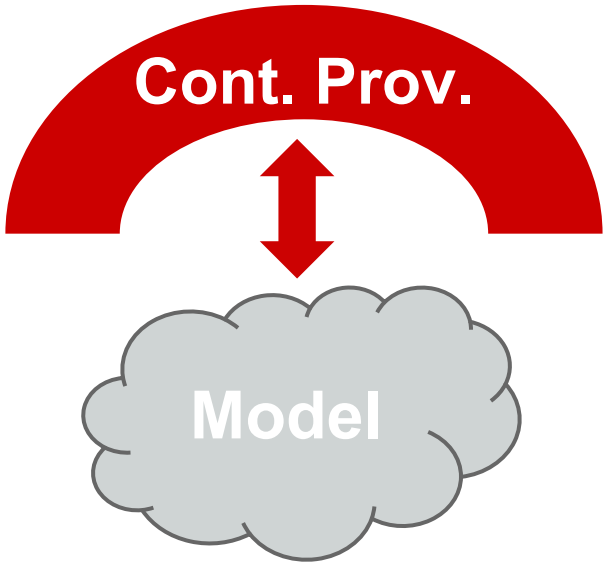
Notification

3.2.3 Content Provider

È un'astrazione di insiemi di **dati strutturati**.

Gestisce l'accesso ai dati.

Garantisce la sicurezza delle operazioni.

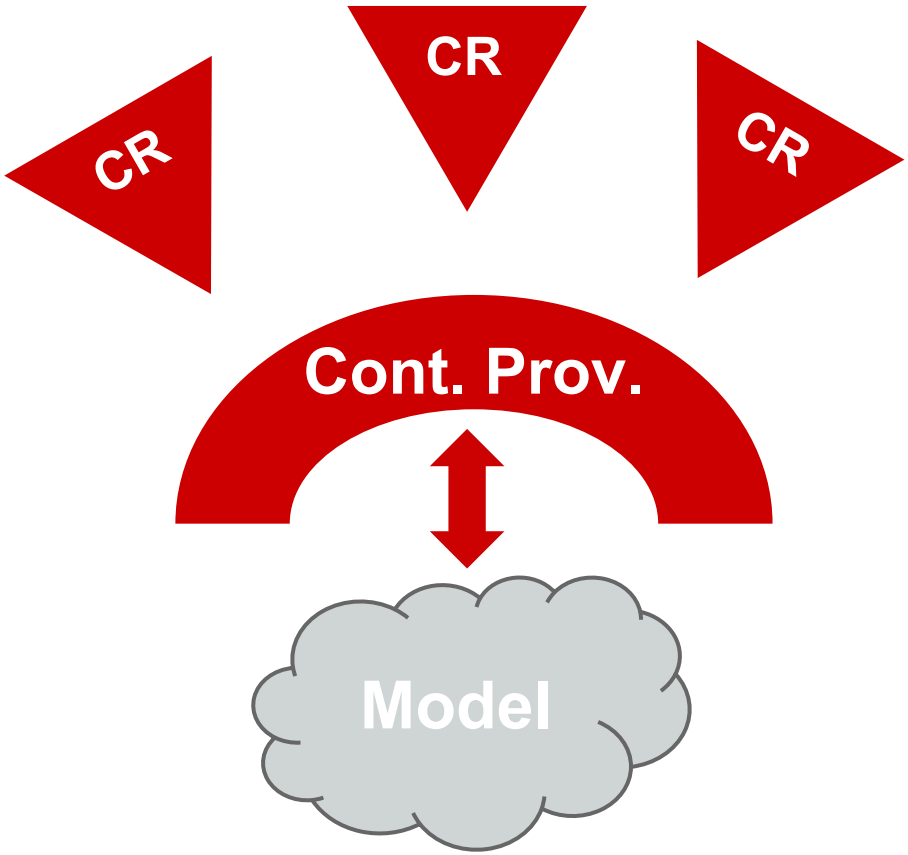


3.2.3 Content Resolver

Ogni applicazione accede ai CP tramite un Content Resolver.

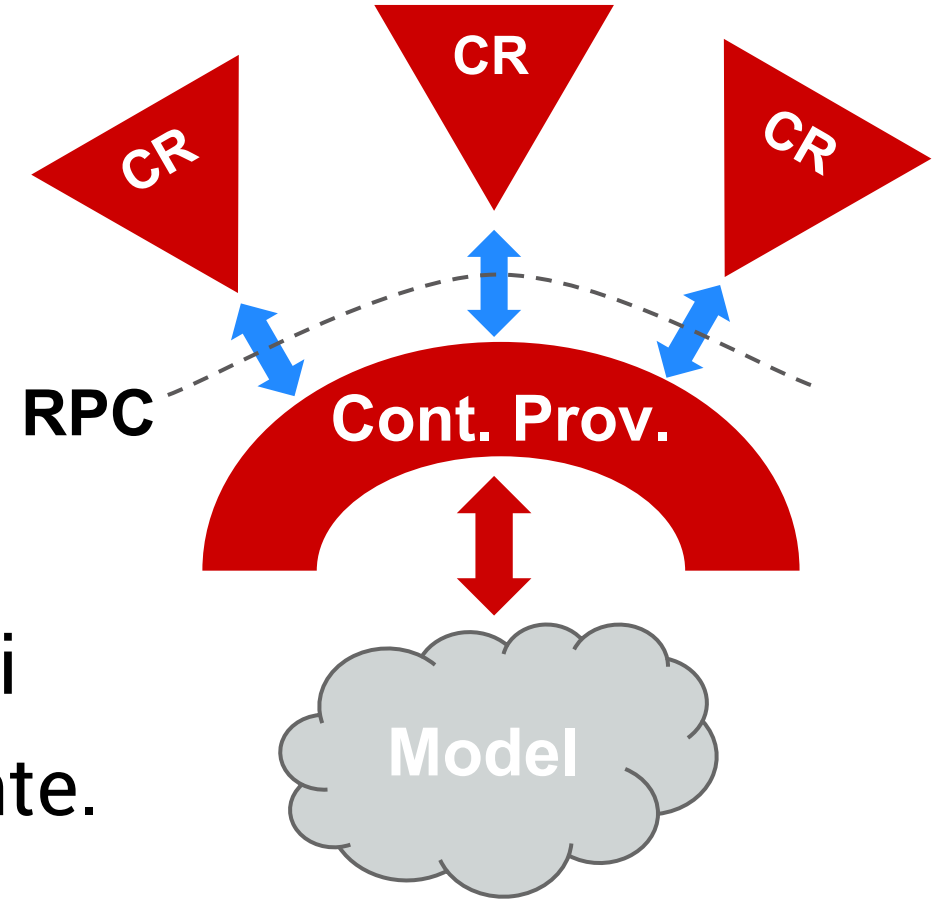
Individua il CP corretto.

Espone metodi **CRUD** (create, retrieve, update, delete).



3.2.3 Content Provider

Un'applicazione non può accedere ai dati di un'altra.



Usato per **condividere dati** ad altre applicazioni o per gestirli privatamente.

3.2.3 Content Provider

Android ne mette a disposizione molti già pronti.

Se ne possono implementare di **personalizzati**.

3.2.3 Content Provider

Esempi:

- Lista dei contatti.
- Lista delle immagini sul dispositivo.
- Dati che risiedono in un DB SQLite.
- Dati che risiedono su un web service.
- Lista delle parole non standard usate.

3.2.4 Elementi principali di un'app

Activity

Service

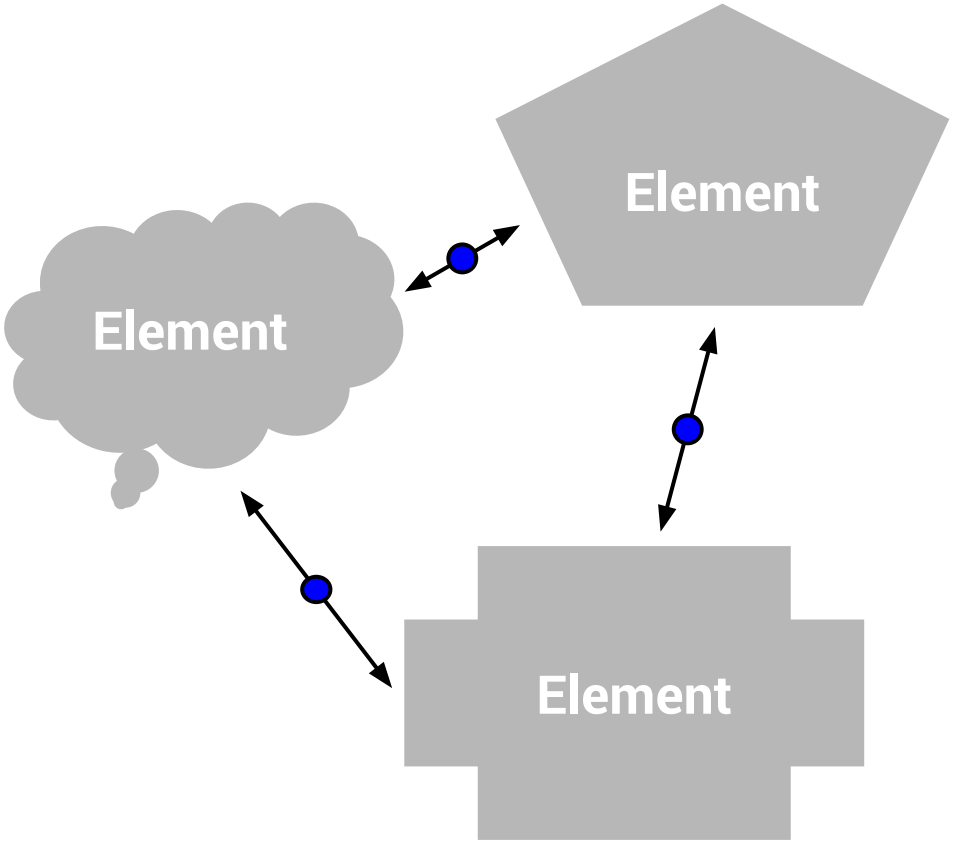
Content Provider

Broadcast Receiver

Intent

Widget

Notification



3.2.4 Intent

Costituiscono il **sistema di messaggistica** fra i vari componenti del sistema (RPC).

Ogni intent è una struttura che **incapsula dati**.

Questi dati costituiscono una astrazione di:

Un'*operazione* da eseguire.

Un *evento* che si è appena verificato.

3.2.4 Intent

Dati di esempio

Campo	Valore
Component name	`it.uniurb.codecamp.test`
Action	`ACTION_CALL`
Data	
URI	`tel:07224475`
Type	`integer`
Category	`CATEGORY_LAUNCHER`
Extras	`"tel_number":int`
Flags	`FLAG_ACTIVITY_SINGLE_TOP`

3.2.4 Intent

Inviare un'immagine:

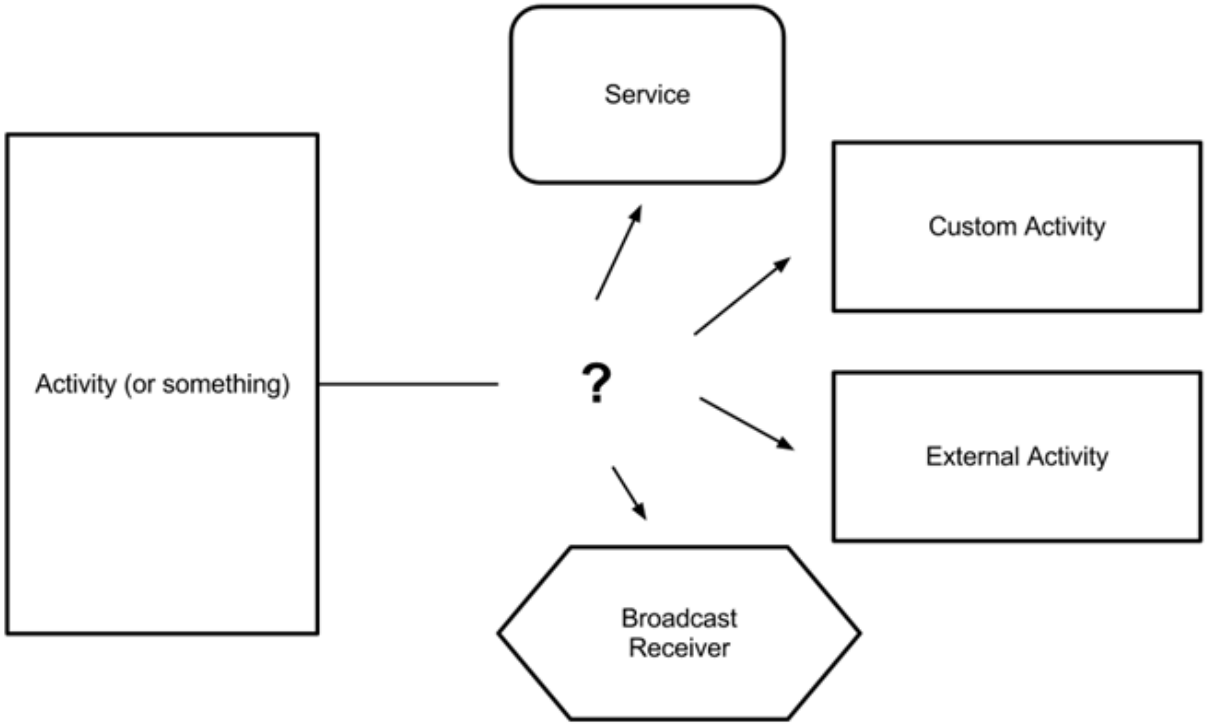
Campo	Valore
Action	`SEND`
Data	`content://media/external/pic16`

Visualizzare un punto in Google Maps

Campo	Valore
Action	`VIEW`
Data	`geo:latitude,longitude?z=zoom`

3.2.4.1 Intent Resolution

Il **mittente** è sempre noto mentre il **ricevente** può non esserlo a priori.



3.2.4.1 Intent Resolution

Intent **esplicito**

Il *component name* del componente destinatario è specificato (class name qualificato).

Intent **implicito**

Il component name non è specificato. Il sistema si occupa di trovare il componente giusto.

3.2.4.1 Explicit Intent

```
Intent intent = new Intent(this, MyActivity.class);
startActivity(intent);
```

Inviato a Activity o Service.

Solo ad Activity nel proprio contesto (class name sconosciuto).

3.2.4.1 Implicit Intent

```
Intent implicitIntent = new Intent(Intent.ACTION_VIEW,
                    Uri.parse("http://www.google.com"));

startActivity(implicitIntent);
```

Action

Data.URI

Android cerca il componente per noi:

Se ne trova **uno**, lo attiva.

Se ne trova **più di uno**,

viene chiesto all'utente quale scegliere.

Se non ne trova **nessuno**, l'avvio fallisce.

3.2.4.2 Intent Filter

Ogni componente dichiara quali Intent impliciti è disposto a ricevere tramite un **Intent Filter**.

```
<intent-filter>

    <action android:name="android.intent.action.VIEW" />
    <action android:name="android.intent.action.EDIT" />
    <action android:name="com.custom.notepad.action.EDIT_NOTE" />

    <category android:name="android.intent.category.DEFAULT" />

    <data android:mimeType="vnd.android.cursor.item/com.app.note" />

</intent-filter>
```

3.2.4.2 Implicit Intent Resolution

Android individua il subset di componenti adatti secondo un preciso ordine:

In base al **tipo del componente**.
(Activity, Service, Broadcast receiver).
In base al campo **`ACTION`**.
In base al campo **`CATEGORY`**.
In base al campo **`DATA (URI e Type)`**.

3.2.5 Elementi principali di un'app

Activity

Service

Content Provider

Broadcast Receiver

Intent

Widget

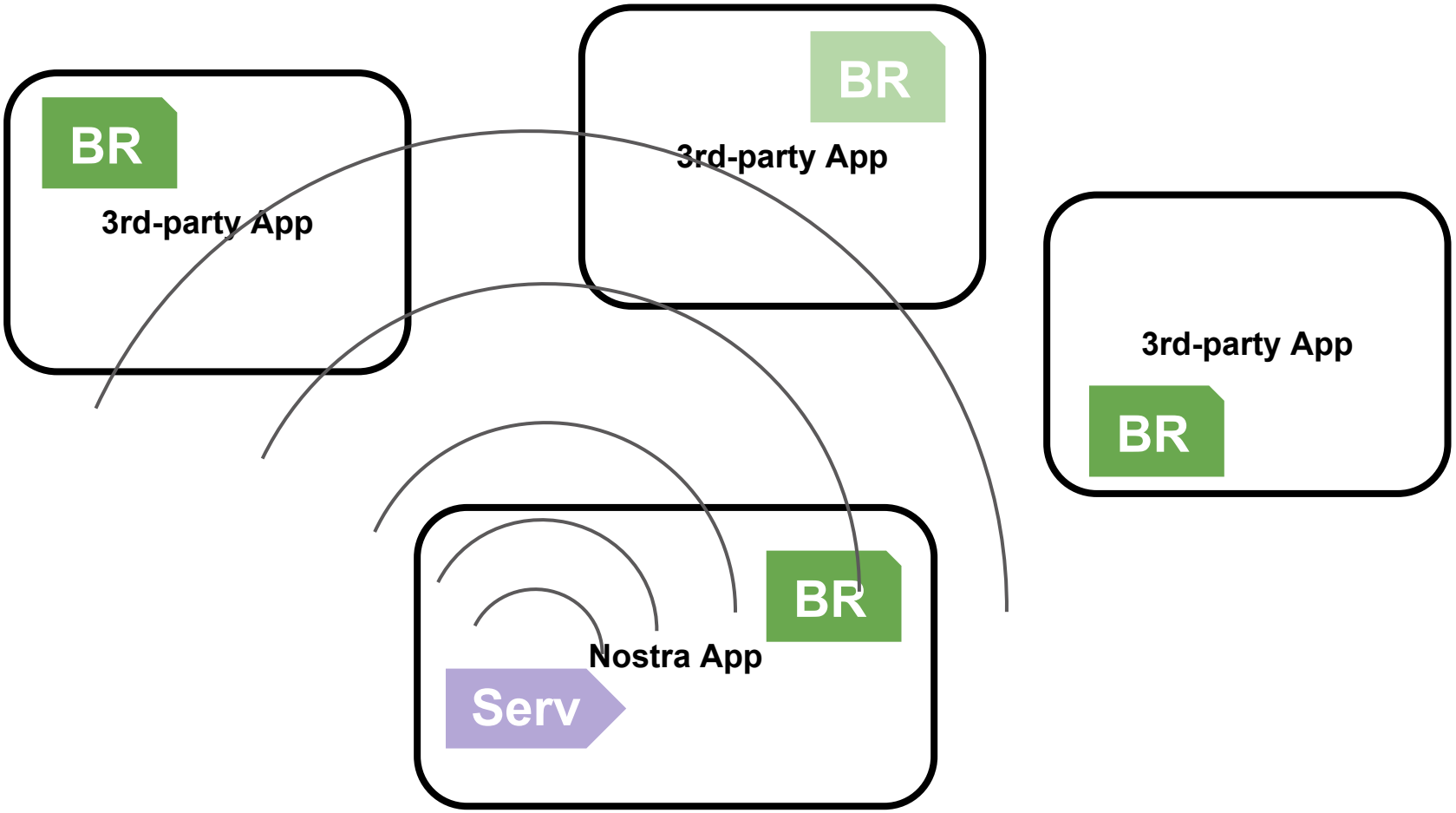
Notification

3.2.5 Broadcast Receiver

Un componente a cui è delegata la ricezione di **Intent in broadcast**.

Sistema per la comunicazione intra-applicazione/intra-processo.

3.2.5 Broadcast Receiver



3.2.5 Broadcast Receiver

Può essere/non essere associato ad un altro componente.

Non possiede una interfaccia grafica.

Può istanziare notifiche.

Può avviare un altro componente.

3.2.6 Elementi principali di un'app

Activity

Service

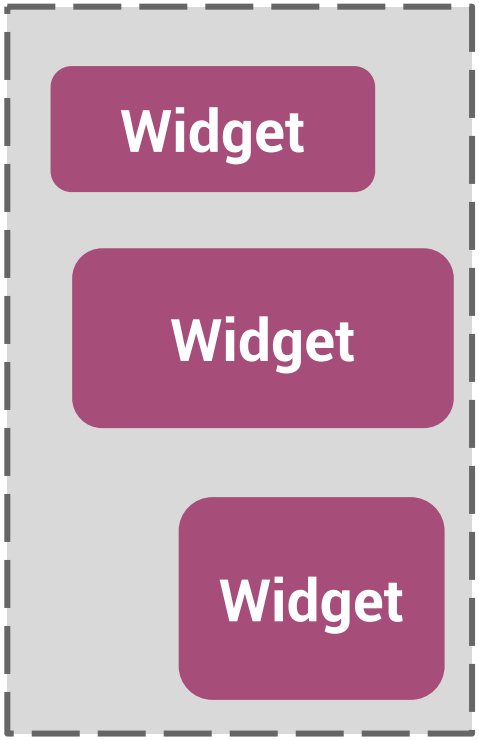
Content Provider

Broadcast Receiver

Intent

Widget

Notification



3.2.6 Widget

Inglobati in altre applicazioni: risiedono all' interno di "*App Widget Host*".

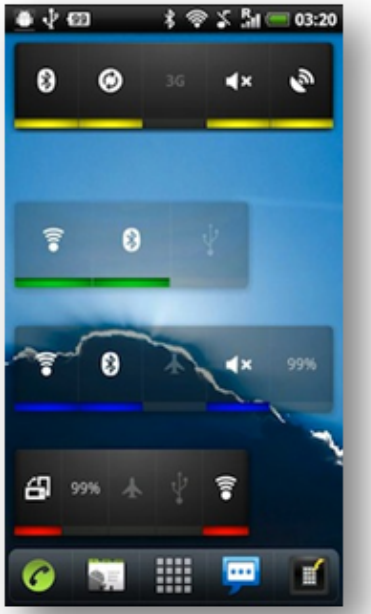
Ricevono **aggiornamenti periodici**.
Periodi di aggiornamento lunghi(min: 30').

Ricevono input dall'utente e aggiornamenti tramite un Broadcast Receiver (che devono implementare).

3.2.6 Widget


4:42 PM
16°
19° / 13°
Sydney
Showers
Wed
Thu
Fri
Sat
17°/13°
19°/8°
19°/11°
20°/7°
04:37 PM
Phone


13:47
13:47
13-Sep.-2009
München
München
Mitteleuropäische Zeit...
München
Telefon


03:20
3G
99%
99%


9:57 PM
What's happening?
Update
ackprash I hear dropbox opened up their android beta sign ups again, join first: http://db.tt/fBRR2P then sign up http://www.dropbox.com/androidbeta
about a minute ago via web
What's on your mind?
Share
Chris Mueller 6 months old tomorrow!!! Where did the time go!? But I do love this stage--sitting on her own, smiles, giggles. Oh motherhood can be so
about a minute ago

3.2.6 Widget


4:42 PM
16°
19° / 13°
Sydney
Showers
Wed
Thu
Fri
Sat
17°/13°
19°/8°
19°/11°
20°/7°
04:37 PM
Phone
13:47
13:47
13-Sep.-2009
München
München
Mitteleuropäische Zeit...
München
Telefon
03:20
3G
99%
99%
9:57 PM
What's happening?
Update
ackprash I hear dropbox opened up their android beta sign ups again, join first: http://db.tt/fBRR2P then sign up http://www.dropbox.com/androidbeta
about a minute ago via web
What's on your mind?
Share
Chris Mueller 6 months old tomorrow!!! Where did the time go!? But I do love this stage--sitting on her own, smiles, giggles. Oh motherhood can be so
about a minute ago

3.2.7 Elementi principali di un'app

Activity

Service

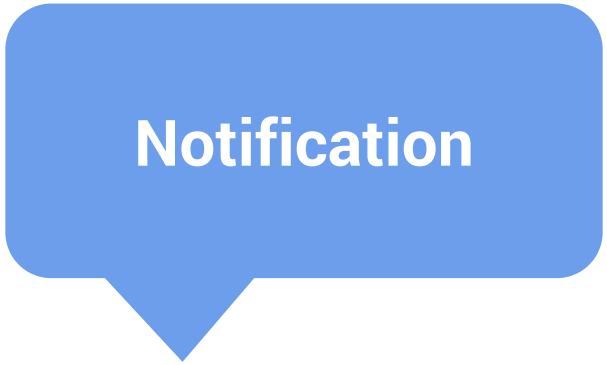
Content Provider

Broadcast Receiver

Intent

Widget

Notification



3.2.7 Notification

Servono per informare l'utente di un **evento scatenato** o **avvenuto**:

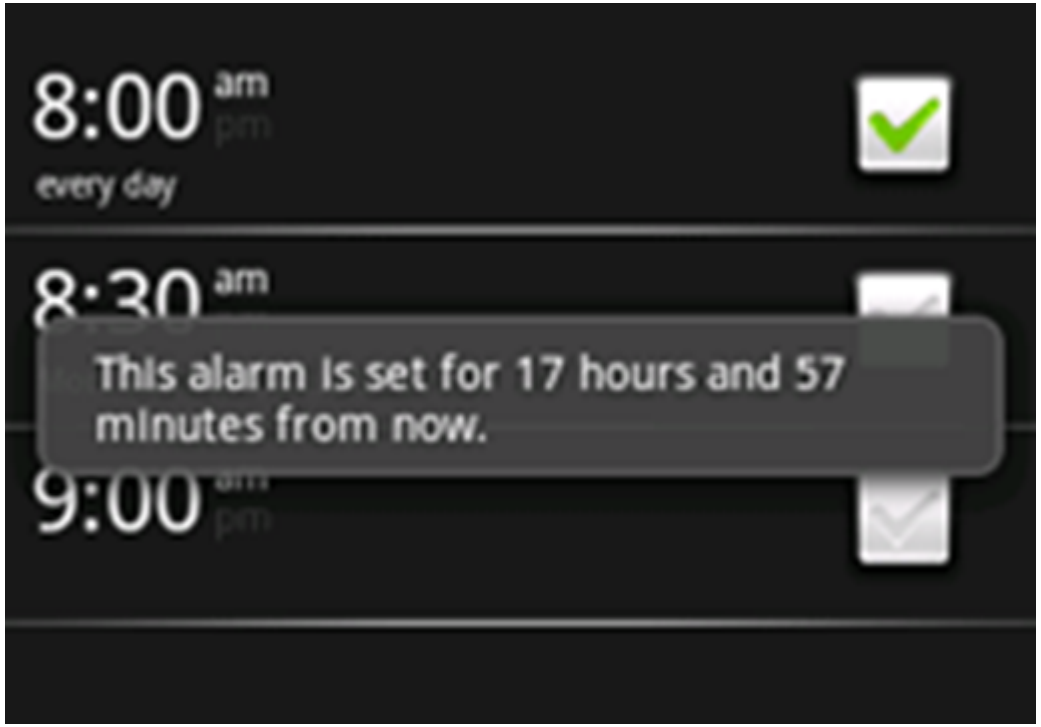
Nel sistema.

Nell'Activity con cui l'utente sta interagendo.

Nei componenti senza interfaccia (come i Service).

3.2.7.1 Toast Notification

Per eventi avvenuti in background.



Nessun input dall’utente.

3.2.7.2 Status Notification

Per eventi relativi avvenuti in background.

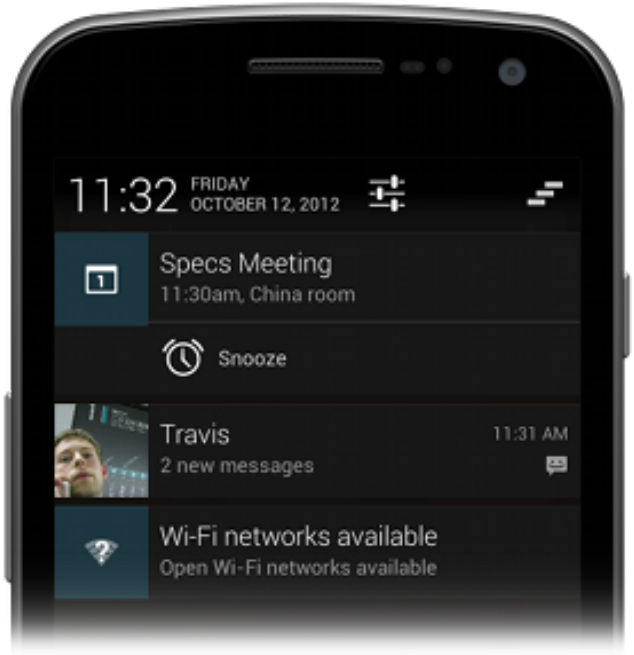
Richiedono l'input utente.

Sono persistenti.

Nella barra delle notifiche finché l'utente non interagisce cliccandoci o cancellandole.

Al click spesso viene lanciata un'activity.

3.2.7.2 Status Notification

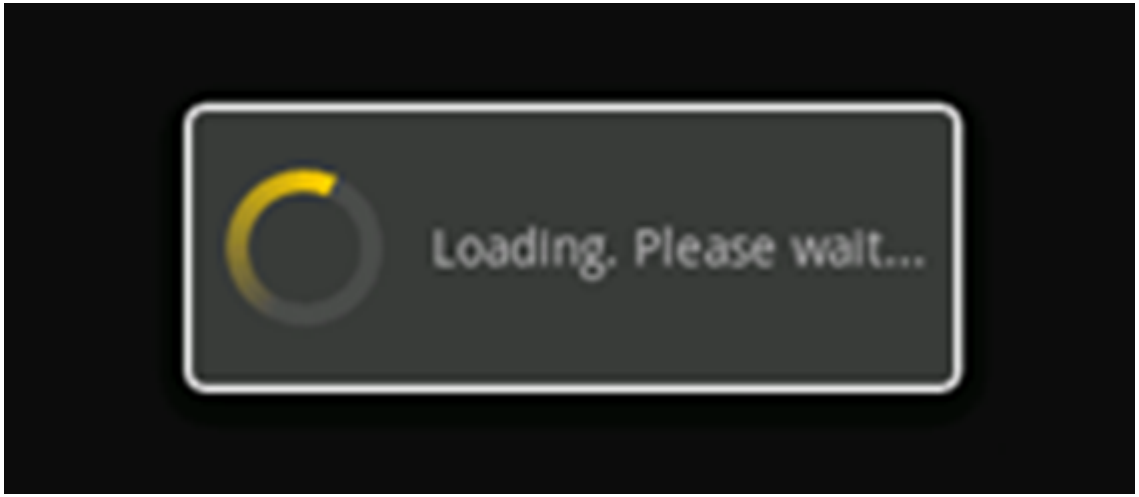


3.2.7.3 Dialog Notification

Per eventi relativi all'activity corrente.

(Caricamento, conferma, etc…)

Supportano l'input utente.



3.3

Approfondimenti sulle applicazioni Android

3.3 Elementi principali di un'app

Layout

Il *back stack* di navigazione.

Ciclo di vita delle Activity.

Ciclo di vita dei Service.

3.3.1 Elementi principali di un'app

Layout

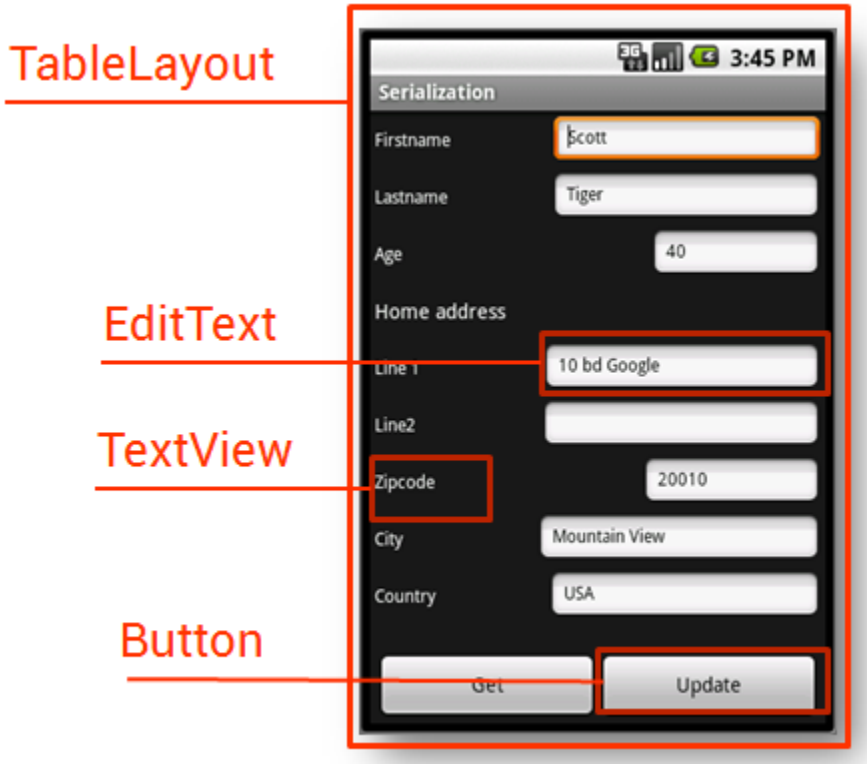
Il *back stack* di navigazione.

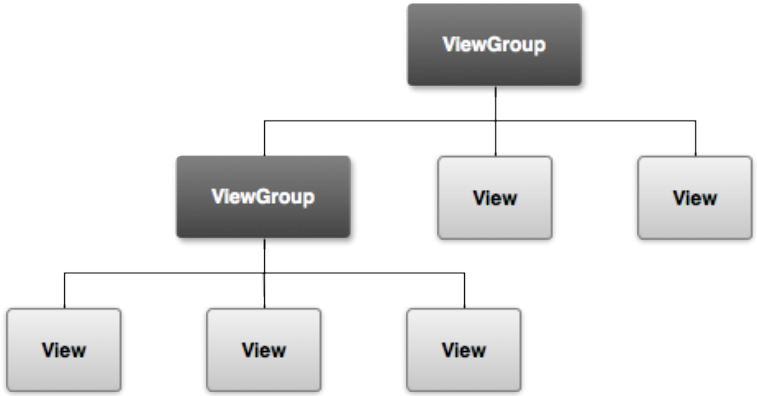
Ciclo di vita delle Activity.

Ciclo di vita dei Service.

3.3.1 Layout

Un activity è composta genericamente da *View* e *ViewGroup* (e *Fragment*).





3.3.1 Layout

Il layout è definito (normalmente) tramite un **file xml**.

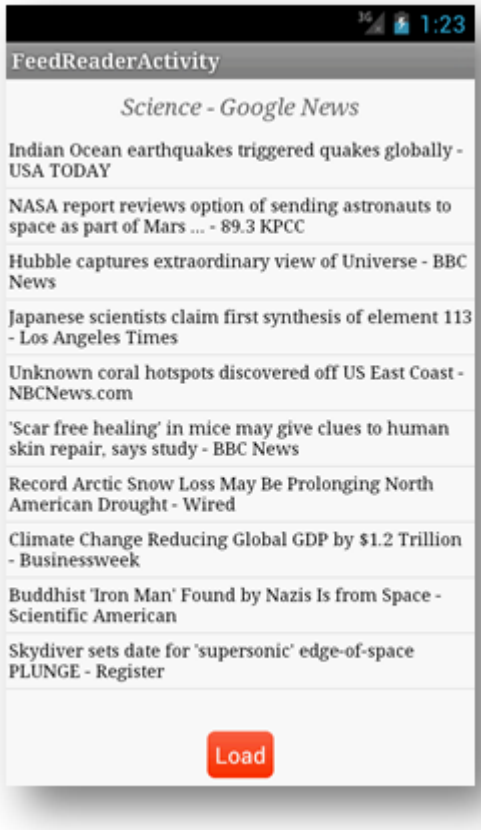
```
<RelativeLayout
   xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent" >

   <TextView
      android:id="@+id/rss_title"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      ... />

   <ListView
      android:id="@+id/rss_list"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      ... >
   </ListView>

   <Button
      android:id="@+id/load"
      android:layout_width="wrap_content"
      ... />

</RelativeLayout>
```



3.3.2 Elementi principali di un'app

Layout

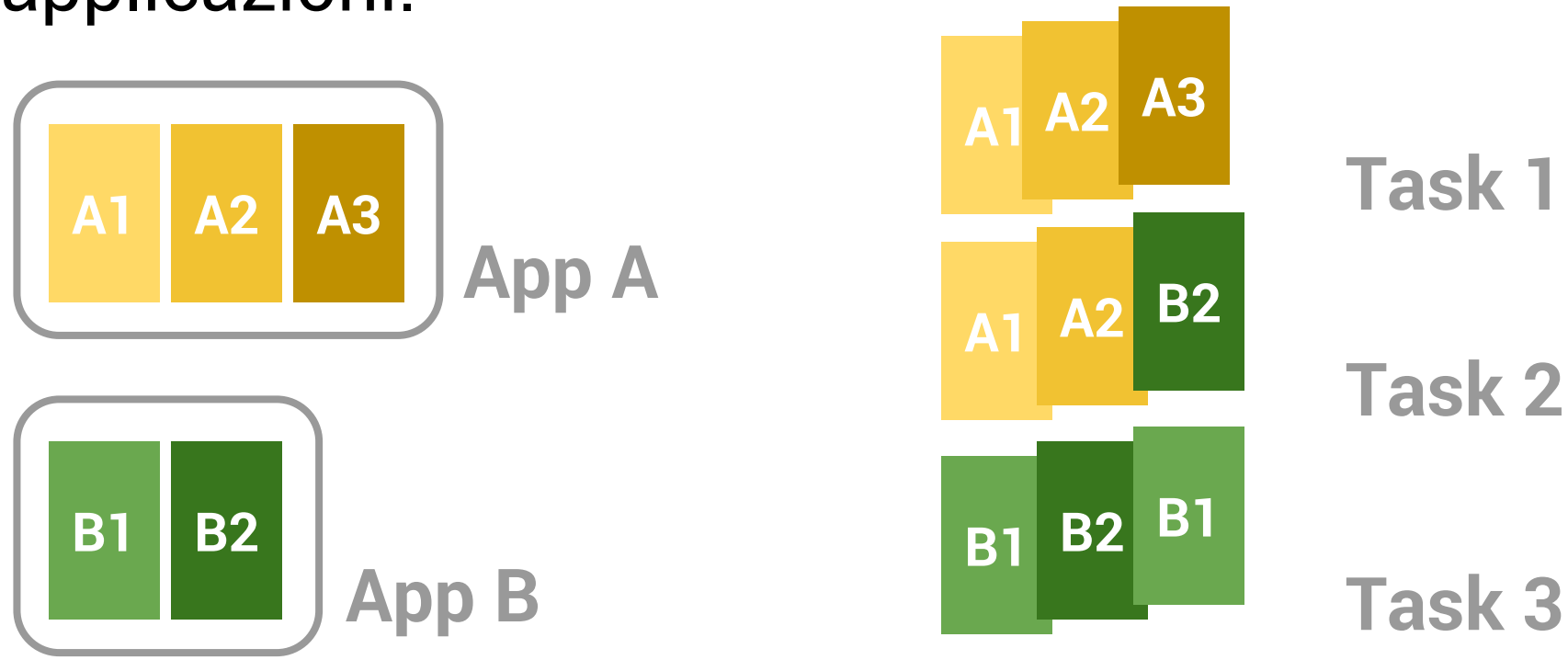
Il *back stack* di navigazione.

Ciclo di vita delle Activity.

Ciclo di vita dei Service.

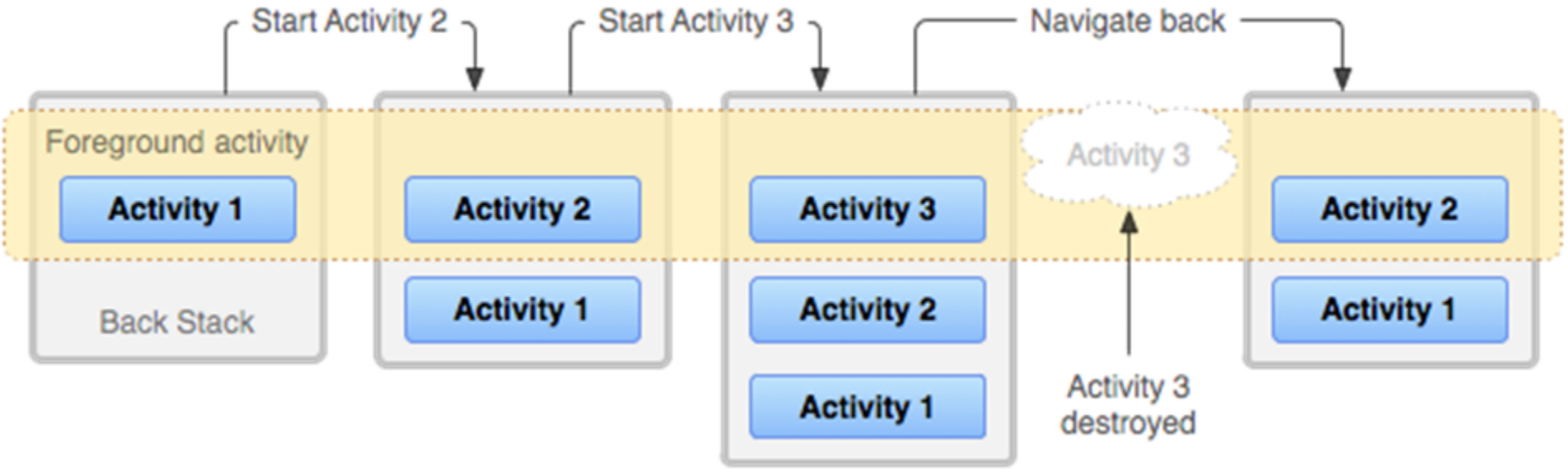
3.3.2 Back stack navigation

Ogni istanza di Activity appartiene ad un **task**. Un task può contenere Activity di diverse applicazioni.



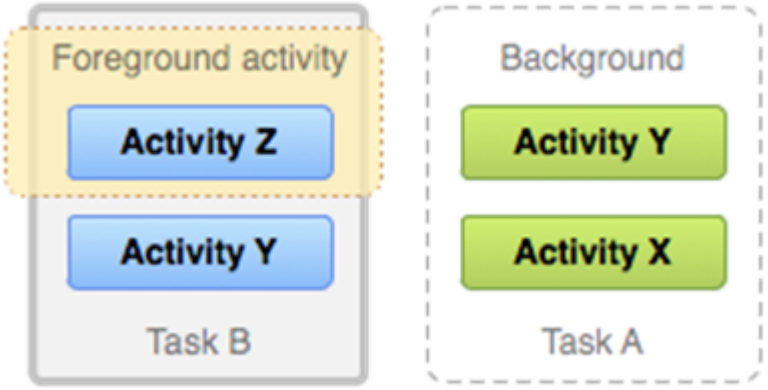
3.3.2 Back stack navigation

Ogni task contiene una collezione di Activity nell'ordine in cui sono state lanciate (**back stack**).



3.3.2 Back stack navigation

Il sistema gestisce contemporaneamente più task ma solo uno per volta è quello in **foreground**.



Un nuovo task inizia ogni volta che un' applicazione è avviata dal launcher:

```
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
```

oppure con le **modalità di lancio**.

3.3.2 Back stack navigation

La **struttura di un task** viene alterata:

- Lanciando una nuova activity (crescita).
- Tasto *Back* (decrescita).

Lo **stato di un task** (*back/fore*) cambia:

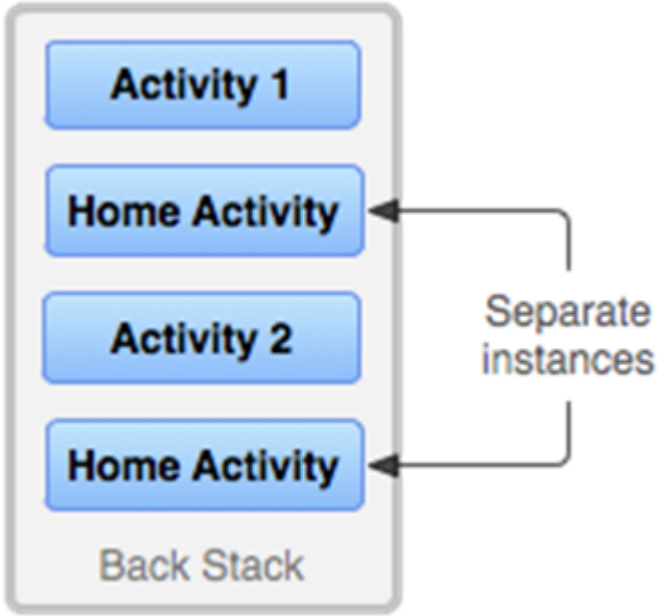
- C'è un'interruzione per eventi esterni.
- Tasto home.
- Avvio di un'activity dal menu delle activity recenti.

3.3.2.1 Modalità di lancio

Normalmente:

Quando viene avviata un'activity per la prima volta questa viene aggiunta al task attuale (oppure ad un task vuoto se MAIN/LAUNCHER).

Anche se la stessa activity era già stata lanciata ne viene creata una nuova istanza.



3.3.2.1 Modalità di lancio

Le Activity posso essere lanciate con modalità di lancio particolari.

Definite nel manifest o nell'intent di lancio.

Modalità	Conseguenze
Default	Viene creata una nuova activity. Istanze diverse della stessa possono appartenere a diversi task. Ogni task può avere più istanze della stessa.
SingleTop	Viene creata una nuova istanza solo se l'activity non è al top dello stack (onNewIntent()). Il resto come Default.
SingleTask	Se c'è già un'istanza in un task viene riavviata quella altrimenti viene avviata una nuova activity in un nuovo task. Può esserci sempre solo un'istanza dell'activity nel sistema.
SingleInstance	Come sopra ma l'activity è sempre l'unica presente nel suo task.

3.3.3 Elementi principali di un'app

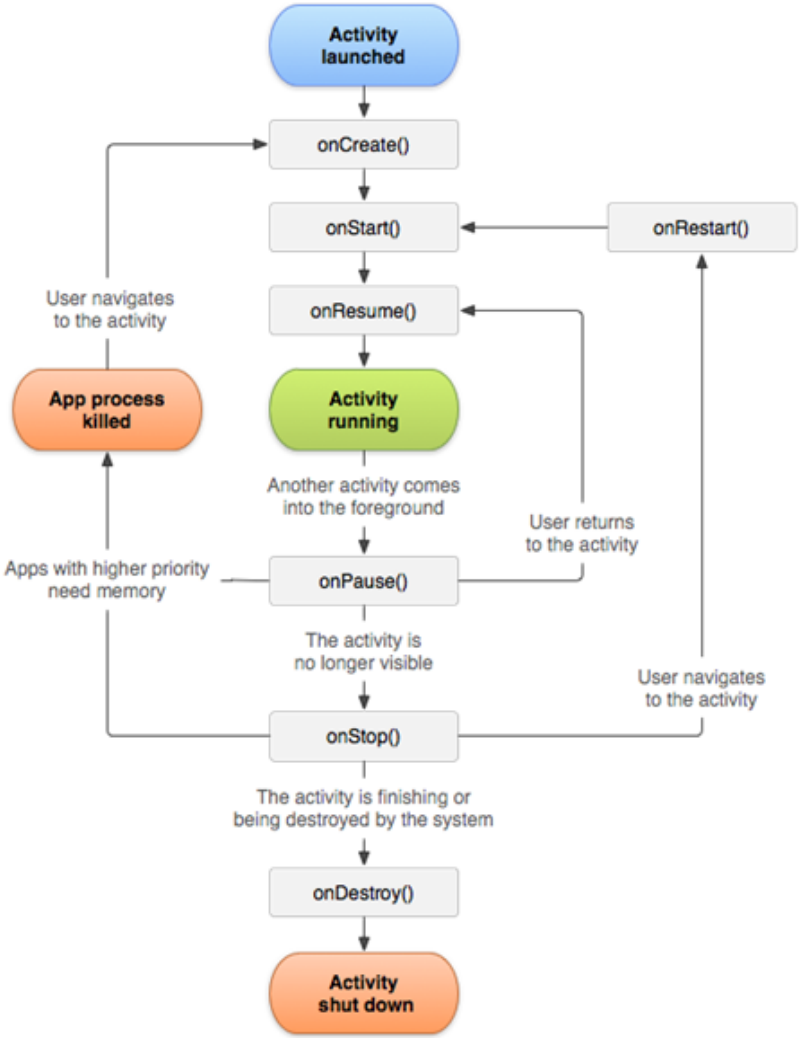
Layout

Il *back stack* di navigazione.

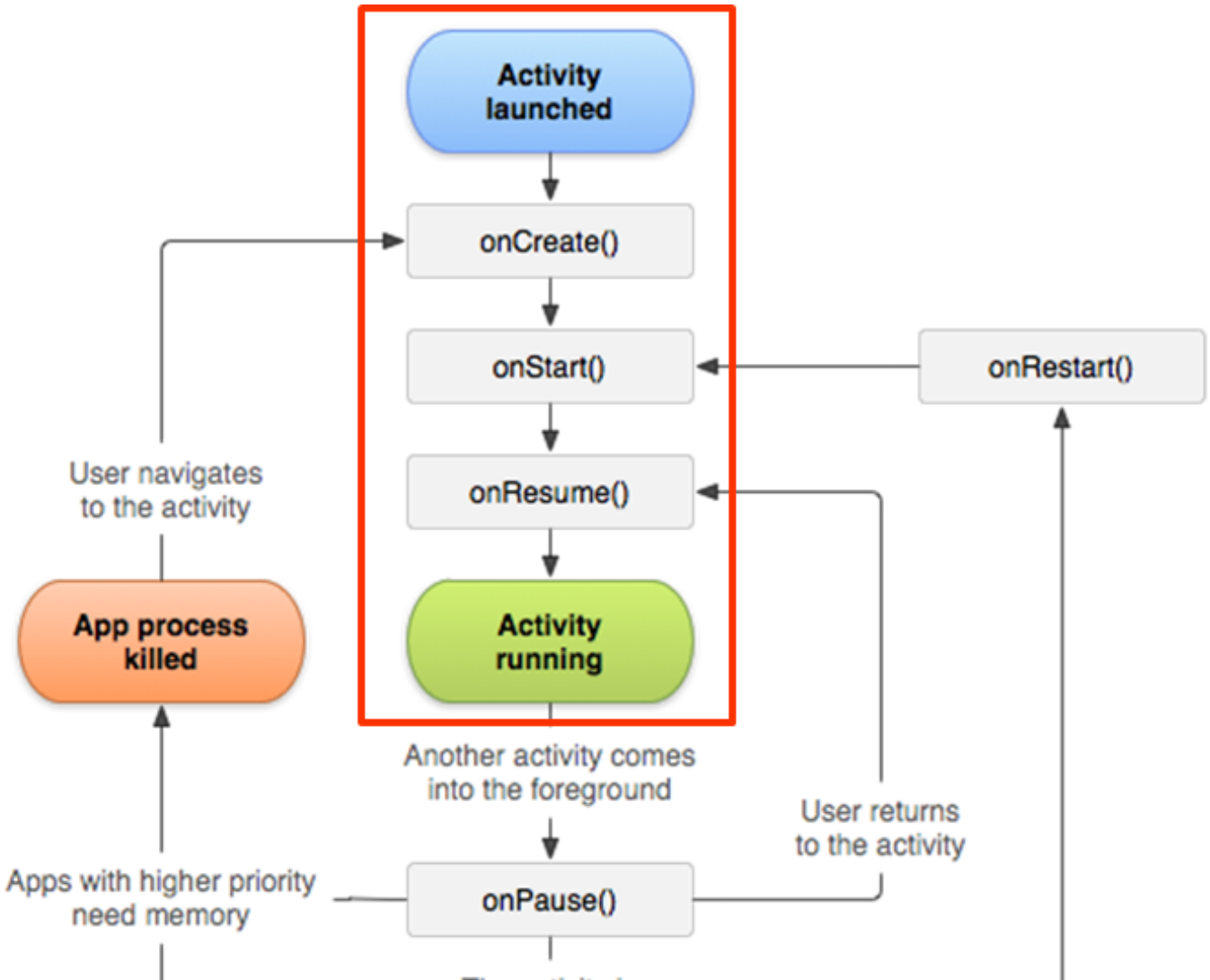
Ciclo di vita delle Activity.

Ciclo di vita dei Service.

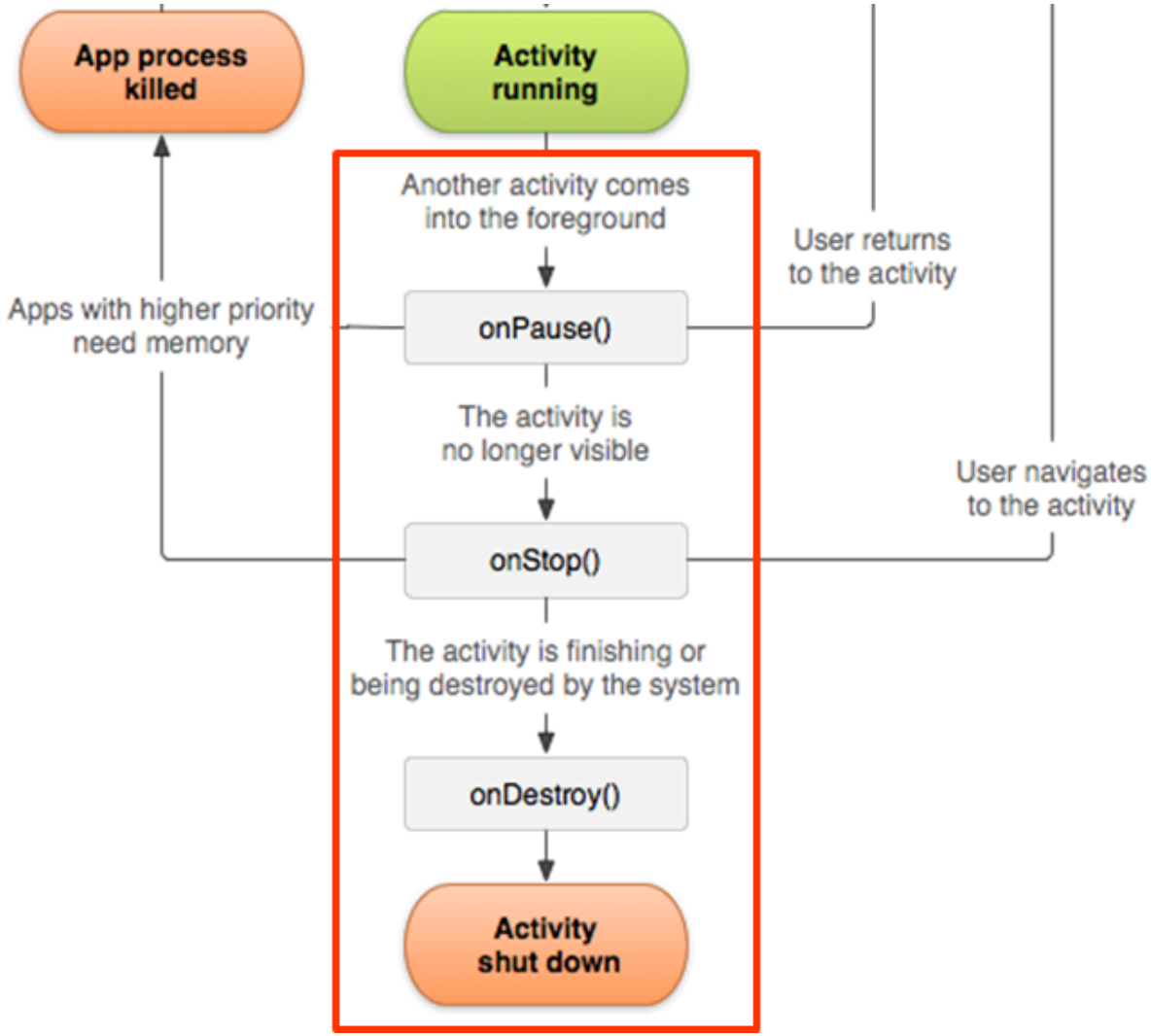
3.3.3 Activity's life cycle



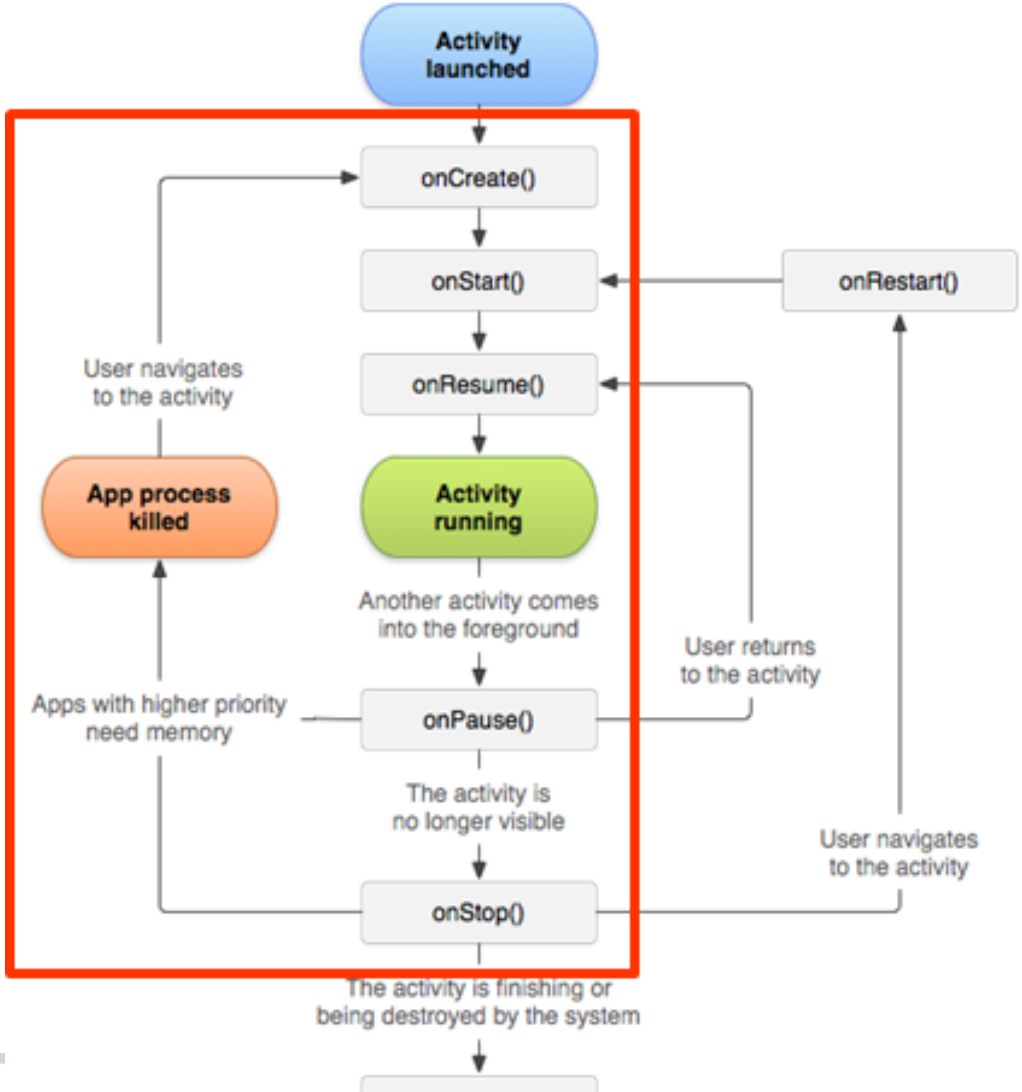
3.3.3 Activity's life cycle



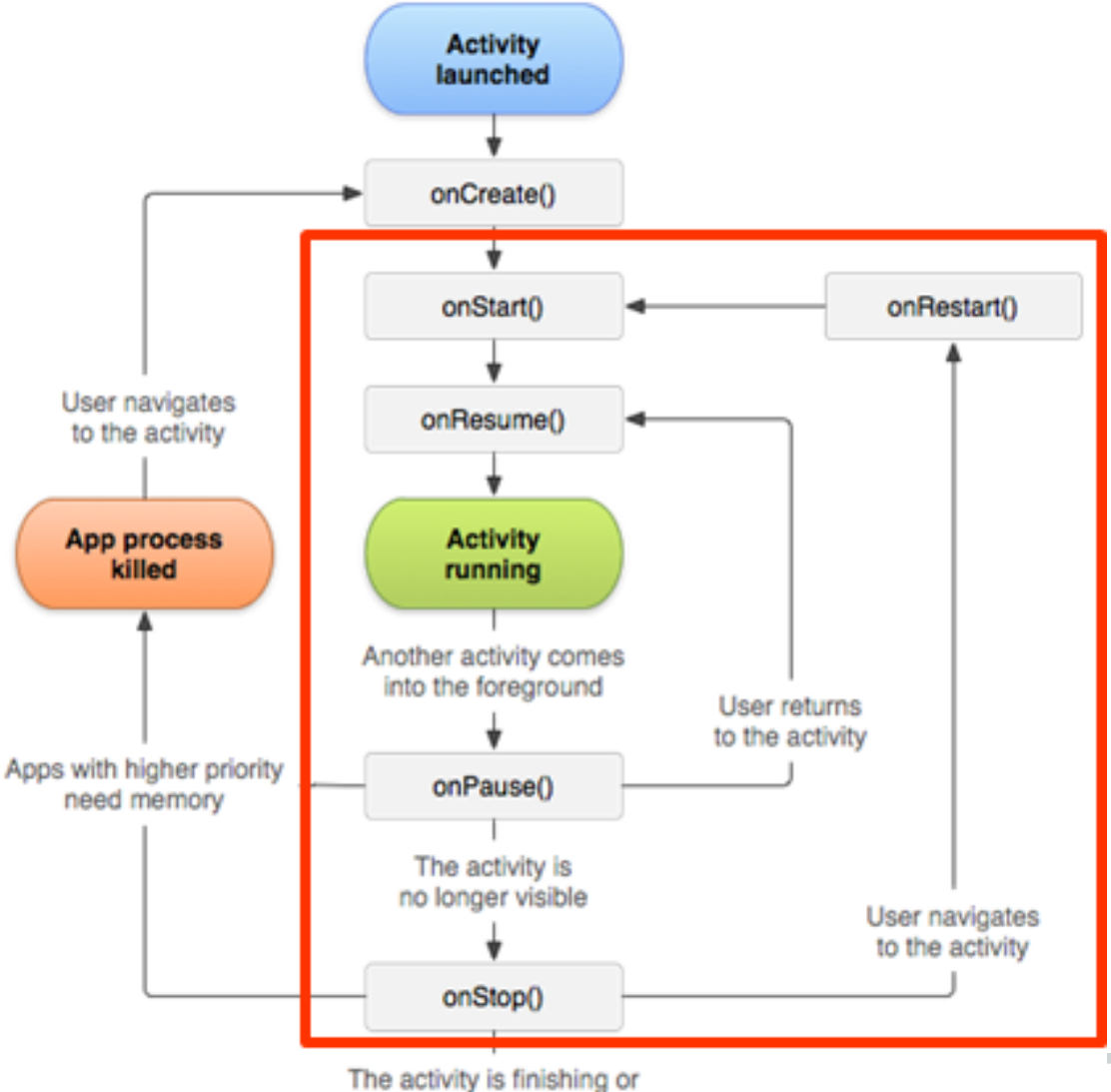
3.3.3 Activity's life cycle



3.3.3 Activity's life cycle

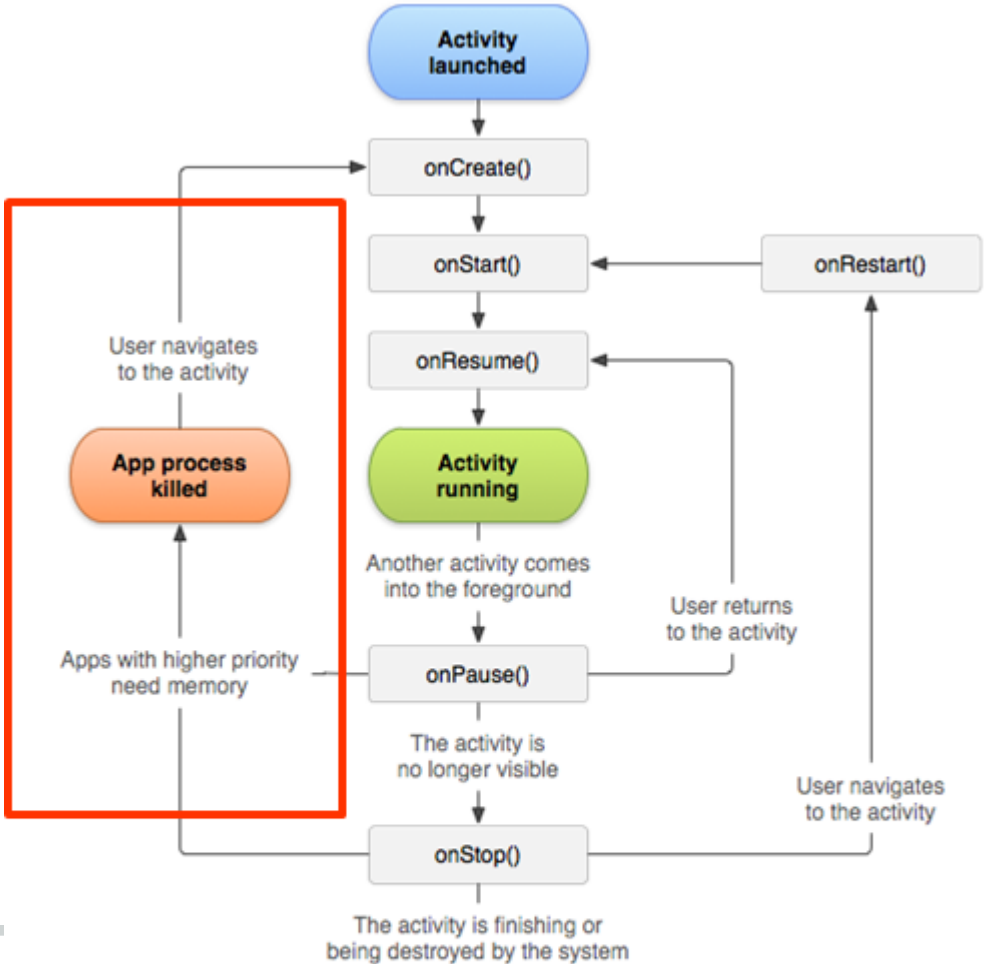


3.3.3 Activity's life cycle



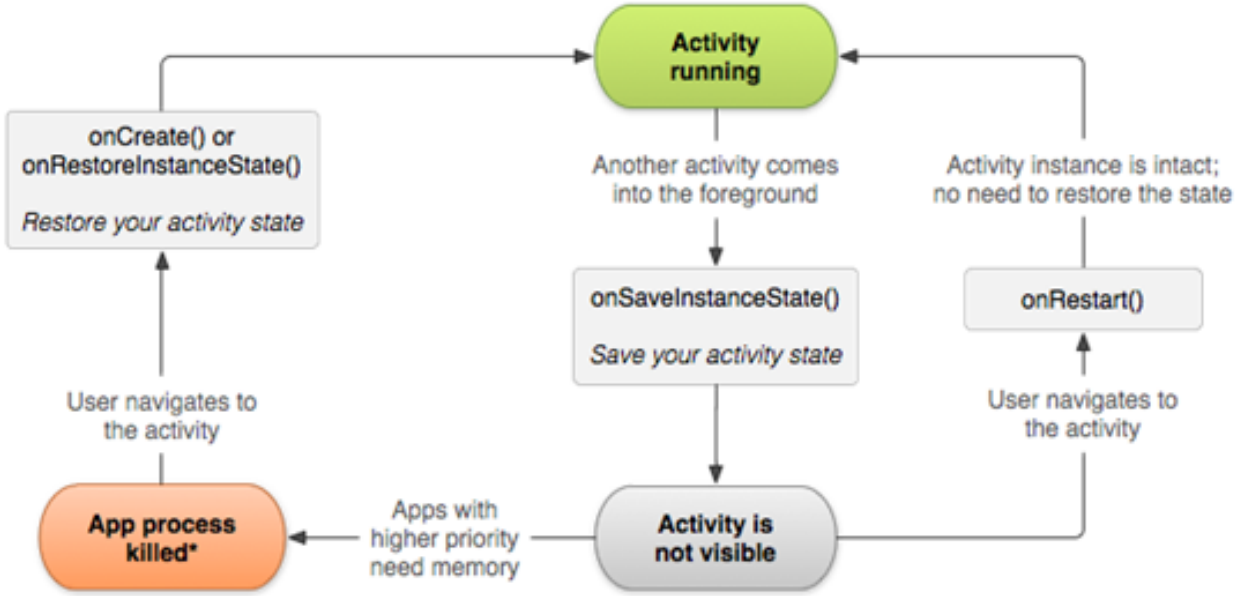
3.3.3.1 Activity's life cycle - state

Salvataggio dello stato dell'activity.

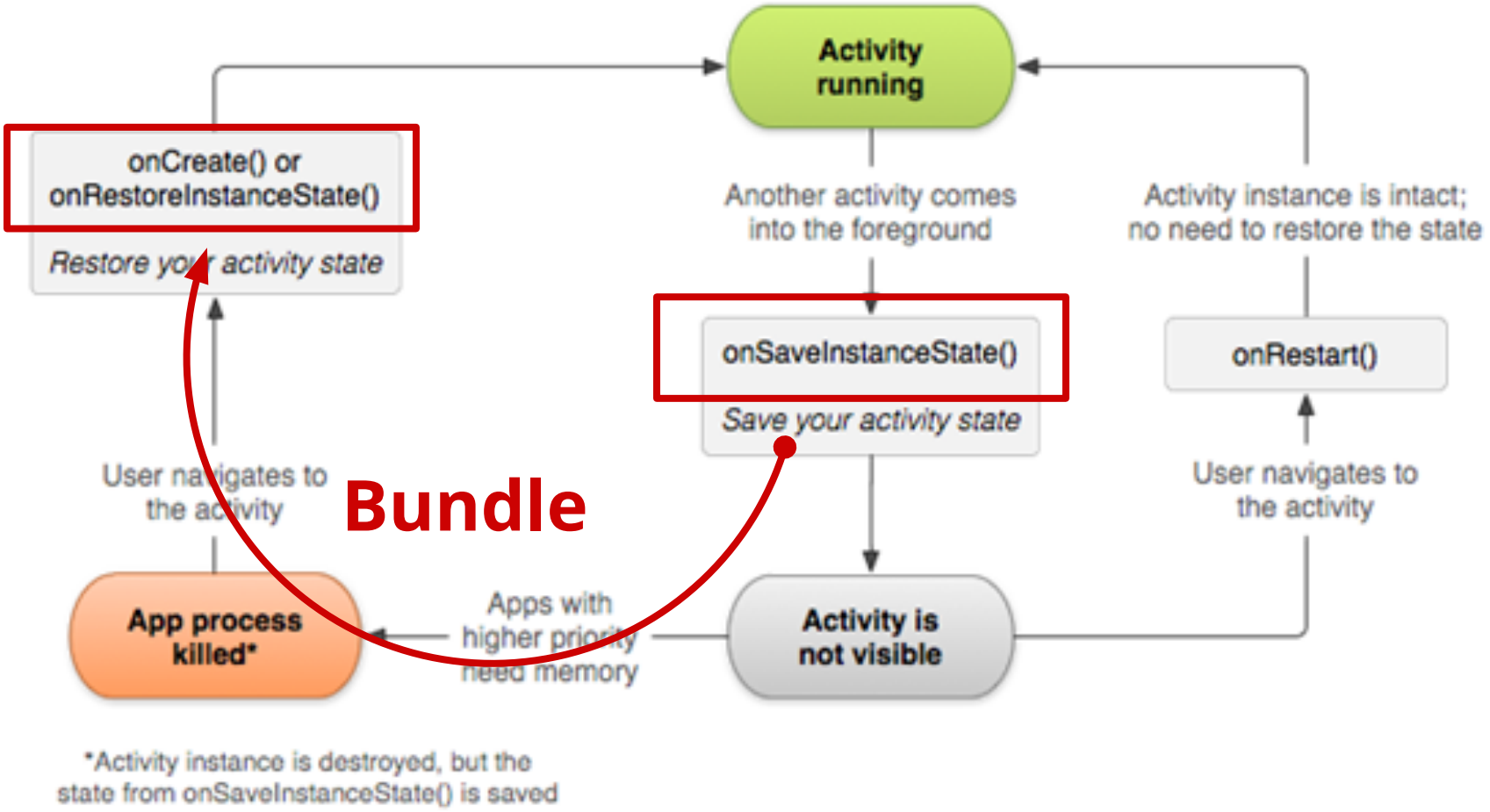


3.3.3.1 Activity's life cycle - state

Un'Activity può essere distrutta dal sistema per recuperare memoria, **ma** l'utente si aspetta di ritrovarla nello **stesso stato**.



3.3.3.1 Activity's life cycle - state



3.3.3.1 Activity's life cycle - state

Esempio: l'ActivityA che visualizza una lista di dati remoti (feed RSS).

1. L'ActivityA viene coperta da un'altra ActivityB.
2. Il sistema distrugge l'ActivityA
3. L'utente preme il tasto *back* e l'ActivityB viene distrutta.
4. L'ActivityA viene ricreata a partire dal *Bundle.*

3.3.3.1 Activity's life cycle - state

Quando sono chiamati?

`onRestoreInstanceState(Bundle)` subito dopo `onStart()`,

`onSaveInstanceState()` prima di `onStop()` e forse prima di `onPause()`.

Non ci sono garanzie che venga effettivamente chiamato (es: *back button*)

3.3.3.1 Activity's life cycle - state

Cosa **salvare** nel `Bundle`?

Lo stato transitorio dell'Activity.

Lo stato dell'interfaccia (lo stato grafico di ogni View viene salvato in ogni caso).

Cosa ***non*** **salvare** nel `Bundle`?

Le informazioni non strettamente relative a quell'istanza dell'Activity.

Le info persistenti dovrebbero essere registrate nel metodo `onPause()`.

3.3.4 Elementi principali di un'app

Layout

Il *back stack* di navigazione.

Ciclo di vita delle Activity.

Ciclo di vita dei Service.

3.3.4 Service's life cycle - state

Started:

Avviato da un altro componente invocando `startService()`.

Viene terminato o termina da solo.

Bound:

Avviato da 1+ componenti invocando `bindService()`.

Sopravvive fino all'ultimo `unbindService()`

3.3.4 Service's life cycle - state