PROGRAMMAZIONE AVANZATA

CAPITOLO 1

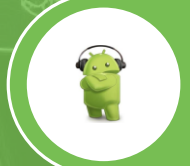
COSA CI PORTERÀ ALLA PRIMA APP?

Introduzione ad Android

Come é fatta una APP

Android Studio

Le Activity



Event e Event Listener

Cosa é Android?

Sistema operativo per dispositivi mobili

Basato sul Kernel Linux

Sviluppato da Google

Progetto Open Source (AOSP)!

Le caratteristiche di Android

Open Source (AOSP)

Virtual Machine (Dalvik/ART)

UI basata su Direct Manipulation

SQLite

Disponibile per diversi dispositivi

Grafica basata su SGL

Grafica 3D basata su OPENGL ES 2.0

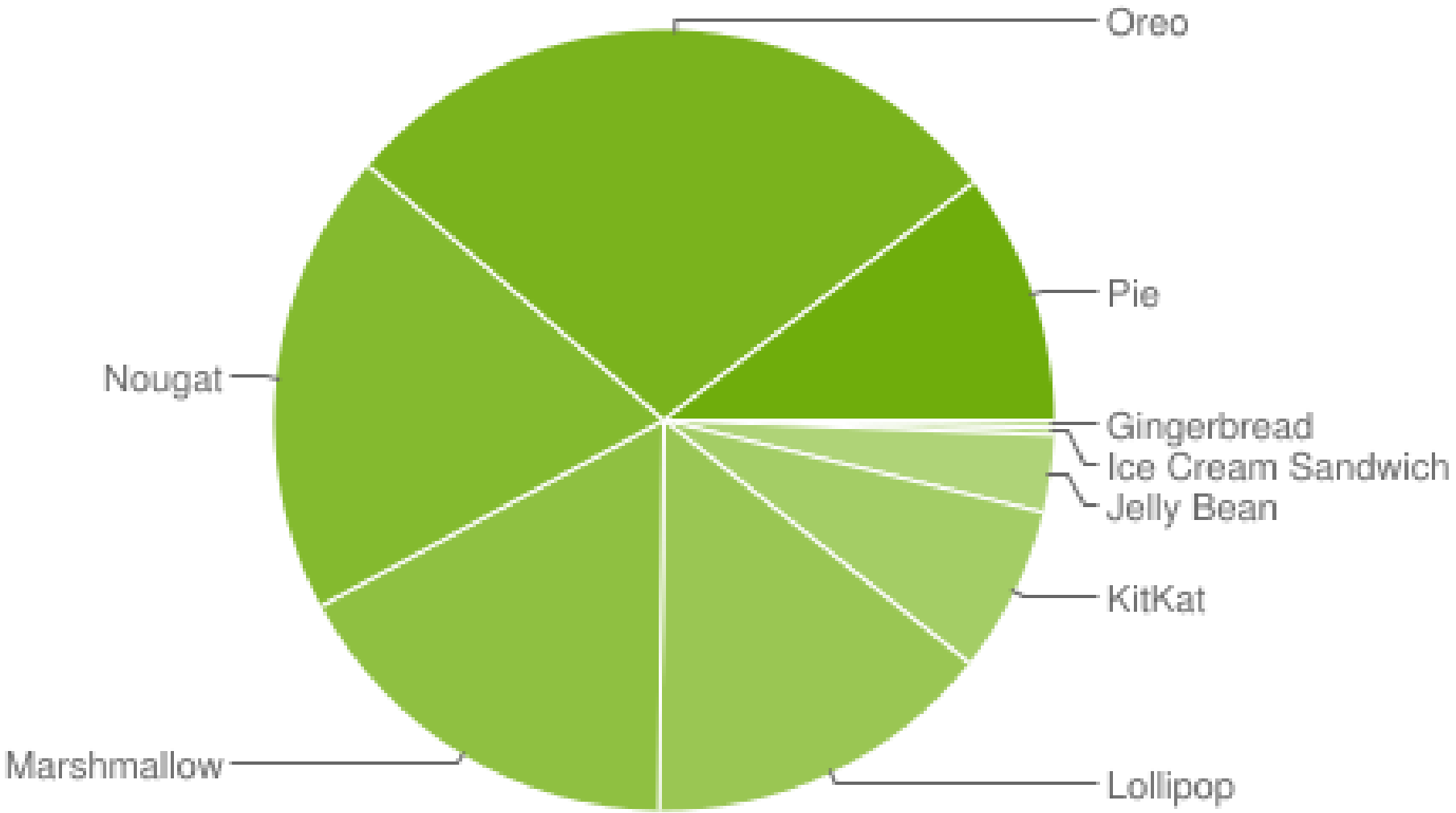
Le versioni di Android

Nomi basati su dolci ?

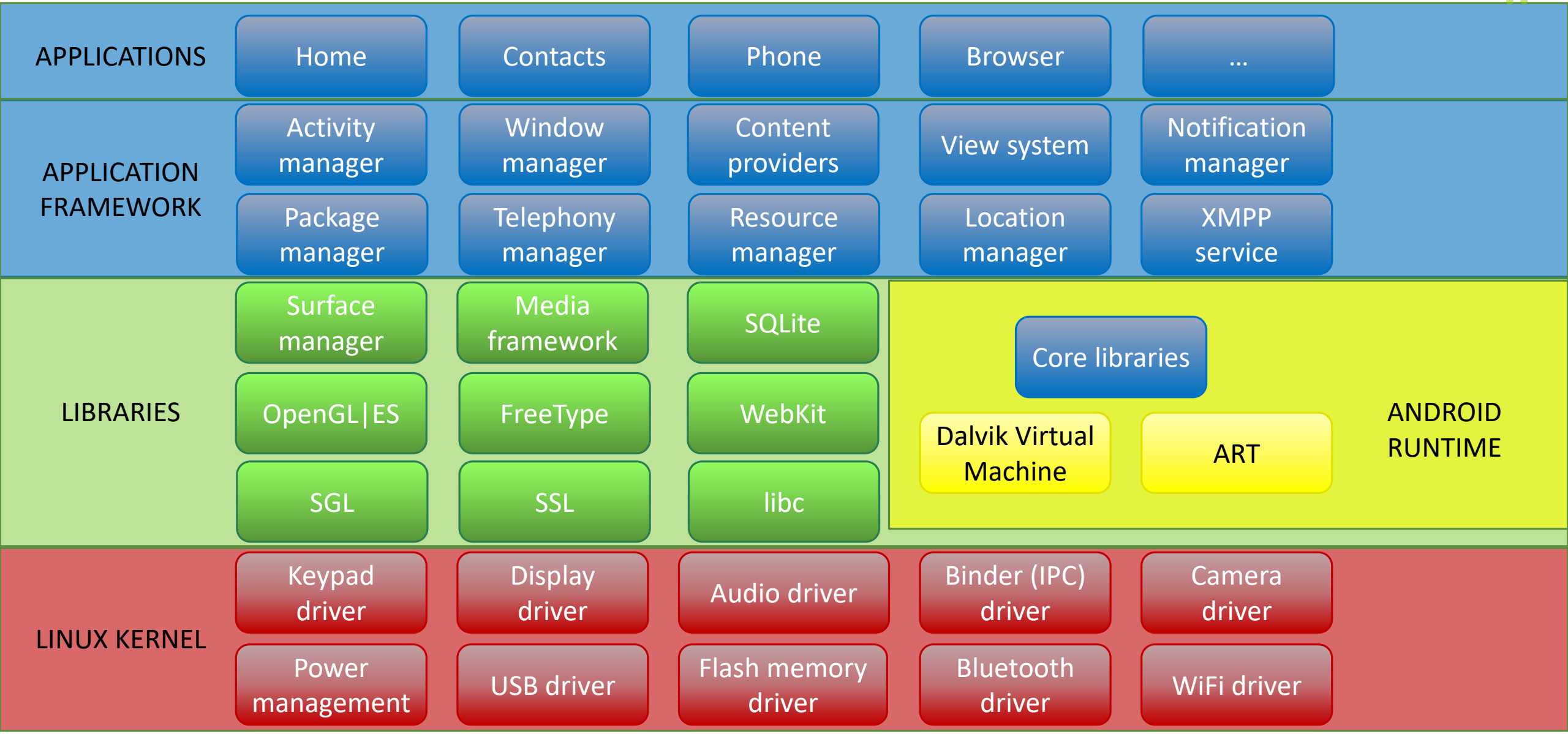
Ogni versione ha il suo numero

Ogni versione ha la sua API

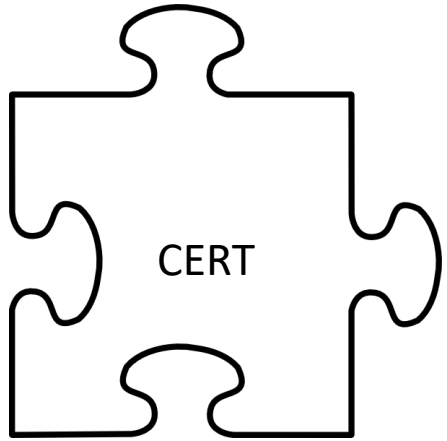
Le versioni di Android

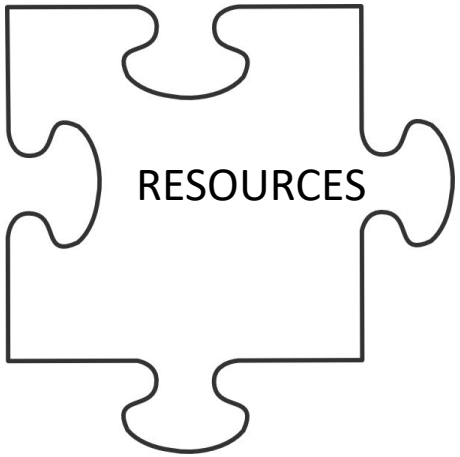


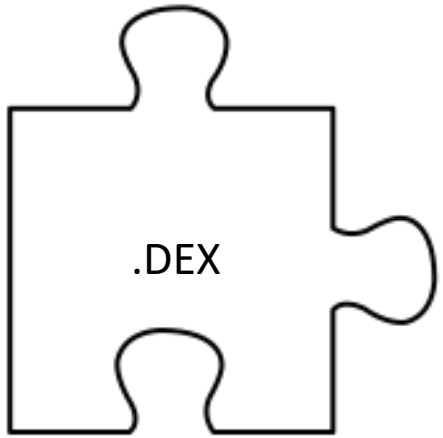
L'architettura di Android

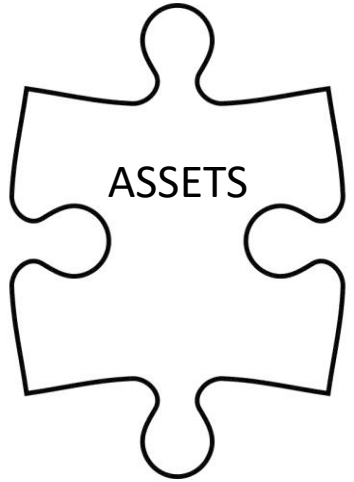


Come é fatta una APP?









Il file AndroidManifest.xml

- Fornisce informazioni sull'APP
- Indica la versione minima di Android su cui può girare l'APP
- Contiene l'elenco dei permessi richiesti
- Specifica il nome dell'APP
- Contiene l'elenco delle activity
- Permette di indicare da quale activity deve partire l'APP

Il file classes.dex

- Contiene la versione compilata di tutto il codice sorgente Java
- Interpretabile da Dalvik
- ART può interpretarlo per effettuare la compilazione

Le risorse

- Immagini
- Layout (xml)
- Componenti grafiche (xml)
- Stringhe (xml)
- Colori (xml)
- Dimensioni (xml)

Gli asset

- Database SQLite
- File audio MP3
- File video

Il certificato

- L'APP contiene il certificato dell'autore
- Meccanismo standard JAVA per la firma tramite META-INF
- Digest SHA-1 per ogni file

Cosa ci serve?

Android Development Kit

- Pacchetto di strumenti
- Compilazione
- Librerie
- Debug
- Versioni per ogni API level

Cosa ci serve?

Android Studio

- IDE basato su Intellij Idea
- Multiplatform
- Include ADK
- Ottimizzato per lo sviluppo su Android
- IDE ufficiale sponsorizzato da Google

https://developer.android.com/studio/index.html

Android Studio

Proviamolo nella pratica!!!

Android Studio

E creiamo la nostra prima APP!!!

Cosa è Gradle?

- Sistema per l'automazione dello sviluppo
- Alternativo a Apache Ant – Apache Maven
- DSL basato su Groovy
- Gestione delle dipendenze tra pacchetti

?????

Android Studio

Pronti?
Facciamo partire la nostra prima APP!

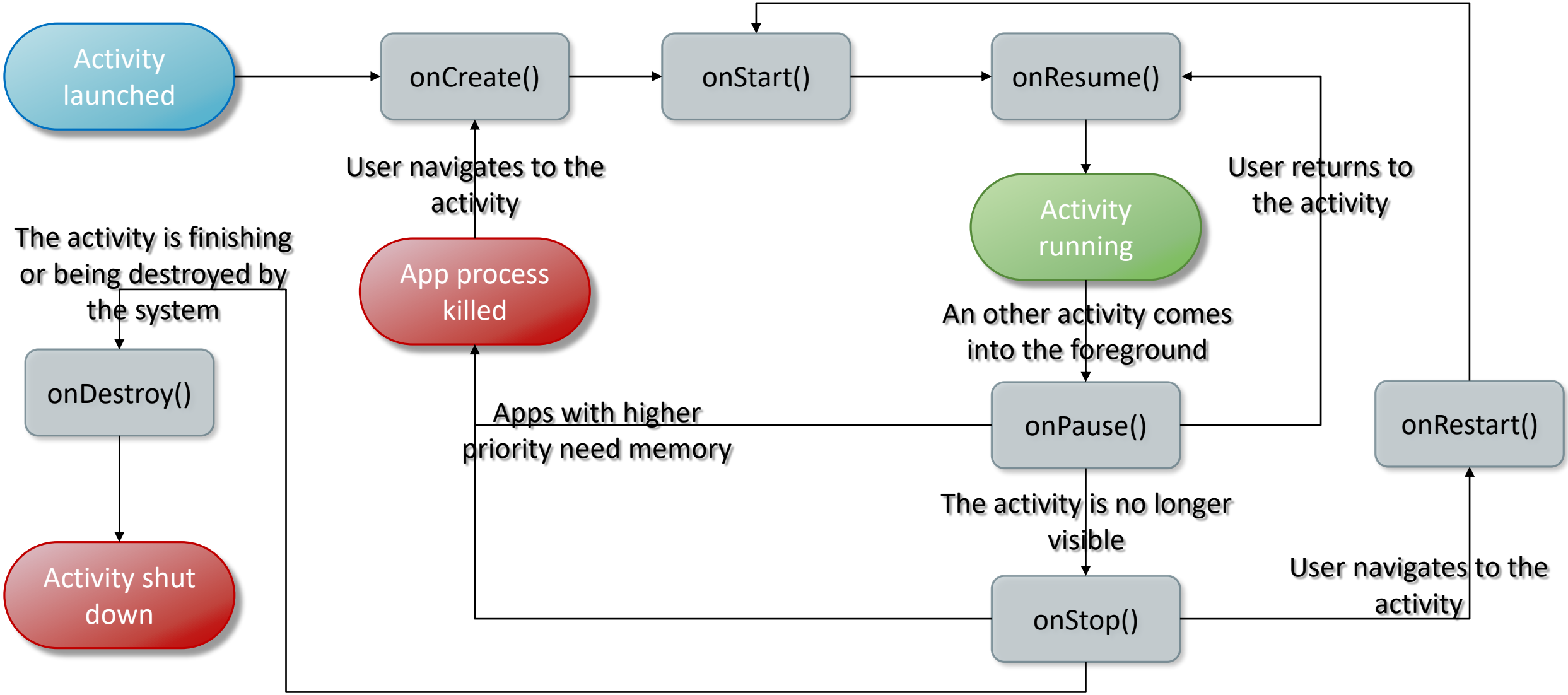
Le Activity

- Sono i mattoni principali delle APP
- Rappresentano il punto di inizio di una APP
- Una Activity fornisce una finestra in cui disegnare tutti gli elementi grafici
- Una Activity tiene traccia del contesto e dei dati
- Permettono la transizione dei dati tra le diverse schermate di una APP

Le Activity

- Devono essere dichiarate nel Manifest
- Su una delle Activity nel Manifest è necessario specificare un filtro per identificarla come l'Activity di default
- Devono estendere la classe AppCompatActivity

Il ciclo di vita delle Activity



onCreate()

- É obbligatorio implementarlo
- Deve richiamare il metodo del padre
- Si usa per implementare la logica di avvio della activity
- Data bind, thread in background, inizializzazione variabili, configurazione UI
- savedInstanceState

onStart()

- Non è obbligatorio implementarlo
- Deve richiamare il metodo del padre
- Molto veloce
- Il sistema, in questo stato, crea effettivamente la UI
- Si può usare per registrare BroadcastReceiver

onResume()

- In questo stato, l'APP interagisce con l'utente
- Deve richiamare il metodo del padre
- L'Activity rimane in questo stato fino a quando non interviene un evento che fa cambiare lo stato
- Si può usare, ad esempio, per far partire delle animazioni

onPause()

- Quando entra in questo stato, l'Activity non viene distrutta, ma, in seguito ad un evento, ha perso il focus (multithread app o dialog o activity trasparente che copre)
- Deve richiamare il metodo del padre
- Normalmente, questo stato dura molto poco
- É sconsigliato effettuare operazioni lunghe o delicate in questo stato
- Si potrebbe utilizzare per fermare le animazioni o la musica o un video
- L'Activity riprende effettivamente l'esecuzione solo passando per onResume()

onStop()

- Quando entra in questo stato, l'Activity non viene distrutta
- Quando una Activity entra in questo stato, una nuova Activity è stata lanciata e copre l'intero schermo
- Deve richiamare il metodo del padre
- In questa fase è importante rilasciare tutte le risorse
- In questo stato, il sistema mantiene le informazioni su tutte le View della Activity
- Si può usare per compiere operazioni lunghe o onerose (salvare in db)

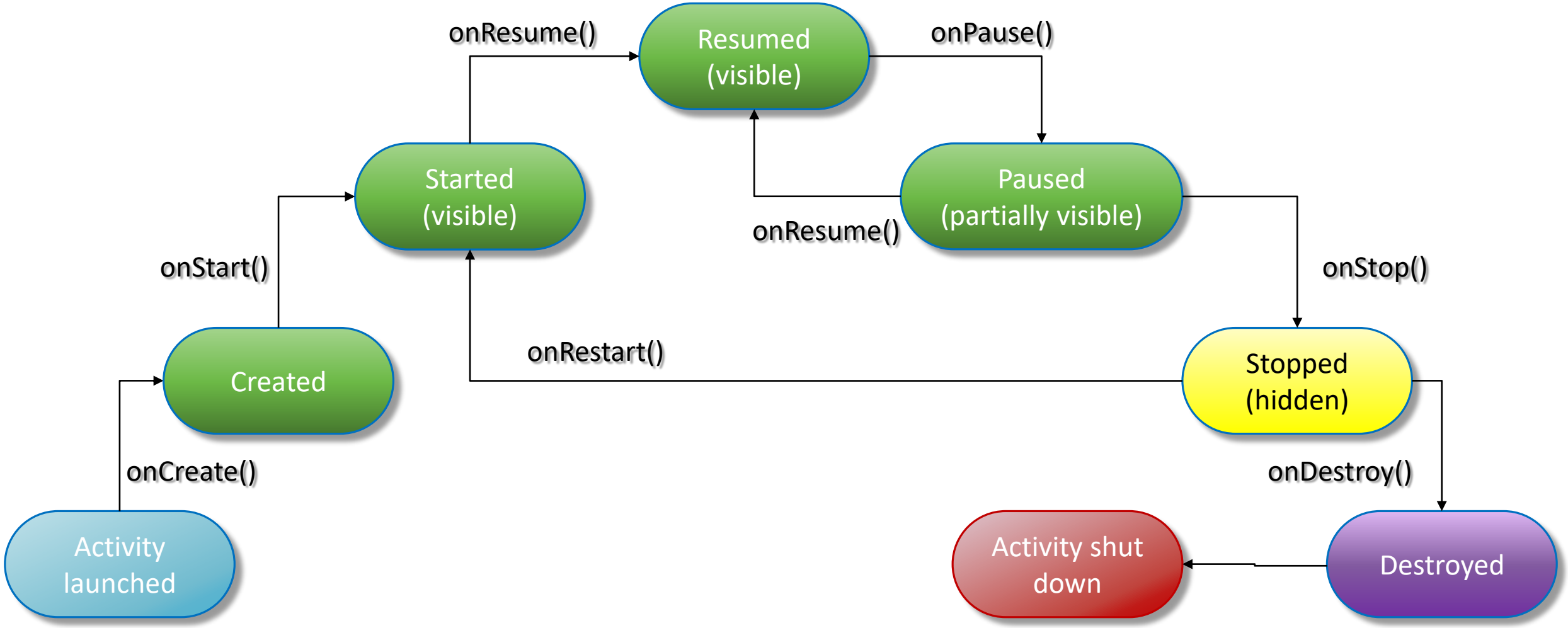
onDestroy()

- Chiamato prima che l'Activity venga distrutta
- Deve richiamare il metodo del padre
- Rilascia tutte le risorse che non sono state rilasciate in onStop()
- Viene richiamato quando si cambia orientamento, perché bisogna passare nuovamente da onCreate per ricreare la UI con la corretta interfaccia

Il ciclo di vita delle Activity

- onCreate, onStart, onResume, onPause, onStop e onDestroy **NON** sono degli stati, ma delle callback, cioè azioni che vengono richiamate al verificarsi di eventi durante il ciclo di vita delle activity
- Per poter raggiungere determinati stati, la activity deve richiamare queste callback

Il ciclo di vita delle Activity



Logcat

- Console di LOG visualizzabile tramite Android Studio
- È possibile scrivere nella console tramite la classe di comodo Log
- Log permette diversi livelli di log: verbose, debug, info, warn, error
- Una riga di log deve sempre avere un TAG, che permette di assegnare una sorta di etichetta alla riga, e un messaggio

Esercizio

- Creare un nuovo progetto di nome CicloDiVita
- Nell'Activity predefinita, ridefinire tutte le callback del ciclo di vita
- In ogni callback, inserire una riga di log per far stampare in Logcat la callback chiamata

Gli event

- Chiamiamo input event (evento di input) tutte le interazioni dell'utente con una View della Activity (o con l'Activity stessa)
  - Click
  - LongClick
  - Focus
  - Key
  - Touch
  - CreateContextMenu

https://developer.android.com/guide/topics/ui/ui-events

Gli event listener

- Permettono di intercettare un event e di effettuare delle operazioni al suo verificarsi
- Si registrano su un particolare event di una particolare View
- Devono essere creati implementando delle interfacce stabilite

Button

- Widget standard
- Un Button è un elemento contenente un testo o una immagine (o entrambi) che intercetta le azioni compiute su se stesso
- Ogni azione compiuta su un Button (event) può essere gestita tramite un opportuno EventListener

CREIAMO UN BUTTON!

Esercizio finale

- Modificare il progetto precedente inserendo un Button con la label “Click me!”
- La pressione del Button deve far comparire un messaggio di Log (debug) con la scritta “Hello world!”

Esercizio finale

- Modificare il progetto precedente inserendo una TextView
- La pressione del Button deve far comparire la scritta “Hello world!” all’interno della TextView