I LAYOUT

CAPITOLO 2

INIZIAMO A CAPIRCI QUALCOSA IN PIÚ

I Layout

Come realizzare una lista

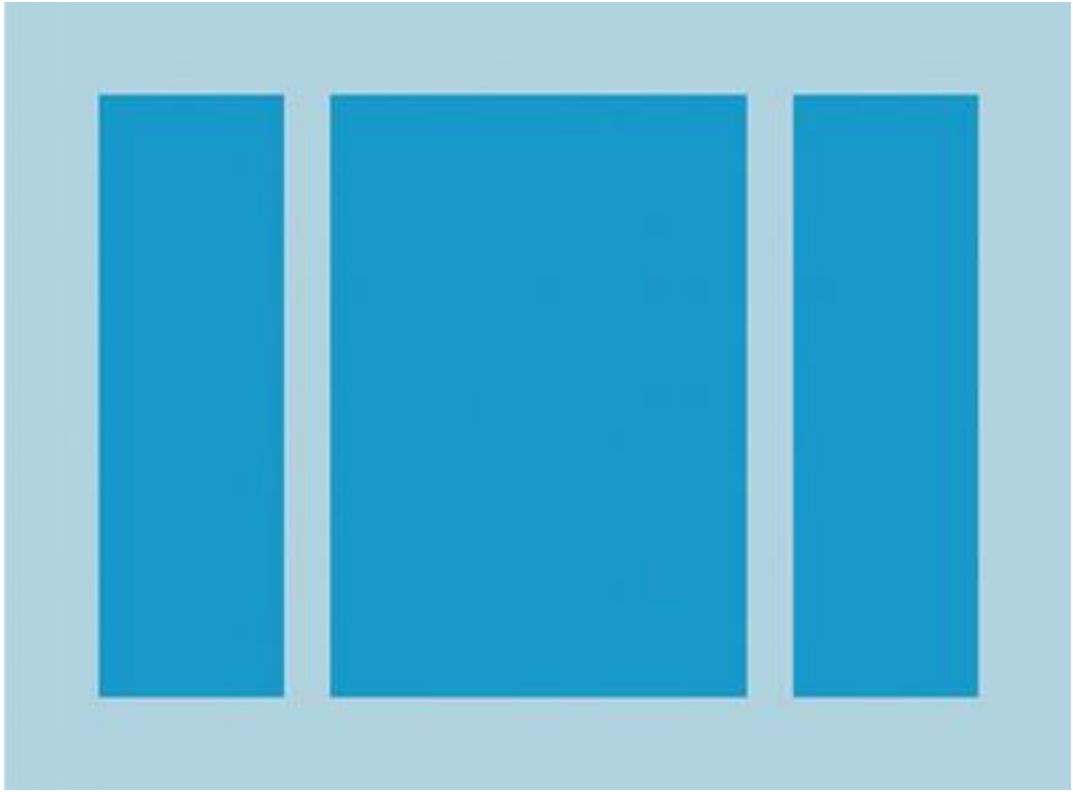
Gli intent in dettaglio

I Layout

- Definiscono come gli elementi grafici verranno posizionati all'interno della UI
- Ogni Layout ha le sue peculiarità e permette di definire alcuni attributi per sfruttarne le caratteristiche
- Esistono Layout per gli utilizzi più comuni, ma è anche possibile crearne di nuovi

LinearLayout

- Tutte le view sono organizzate in maniera lineare, una accanto all’altra
- Può avere orientamento orizzontale o verticale

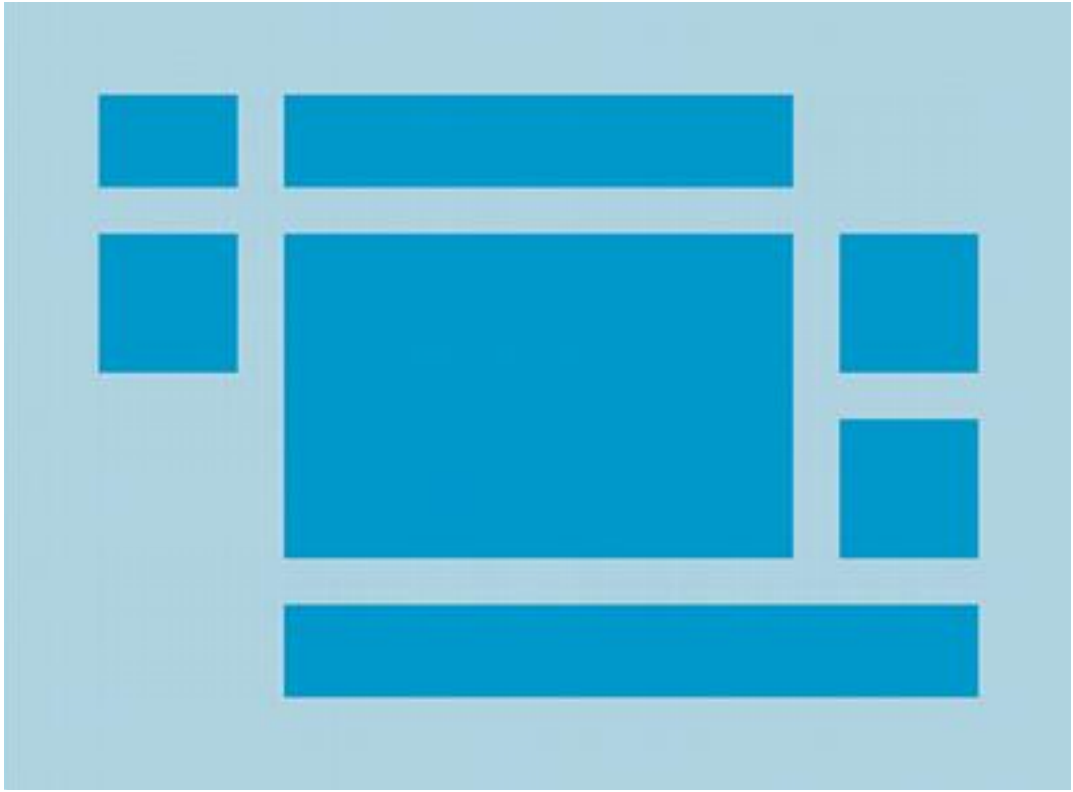


RelativeLayout

- La posizione di una view può essere specificata in relazione alla posizione delle altre
- Si può specificare la posizione anche in relazione all'intero Layout

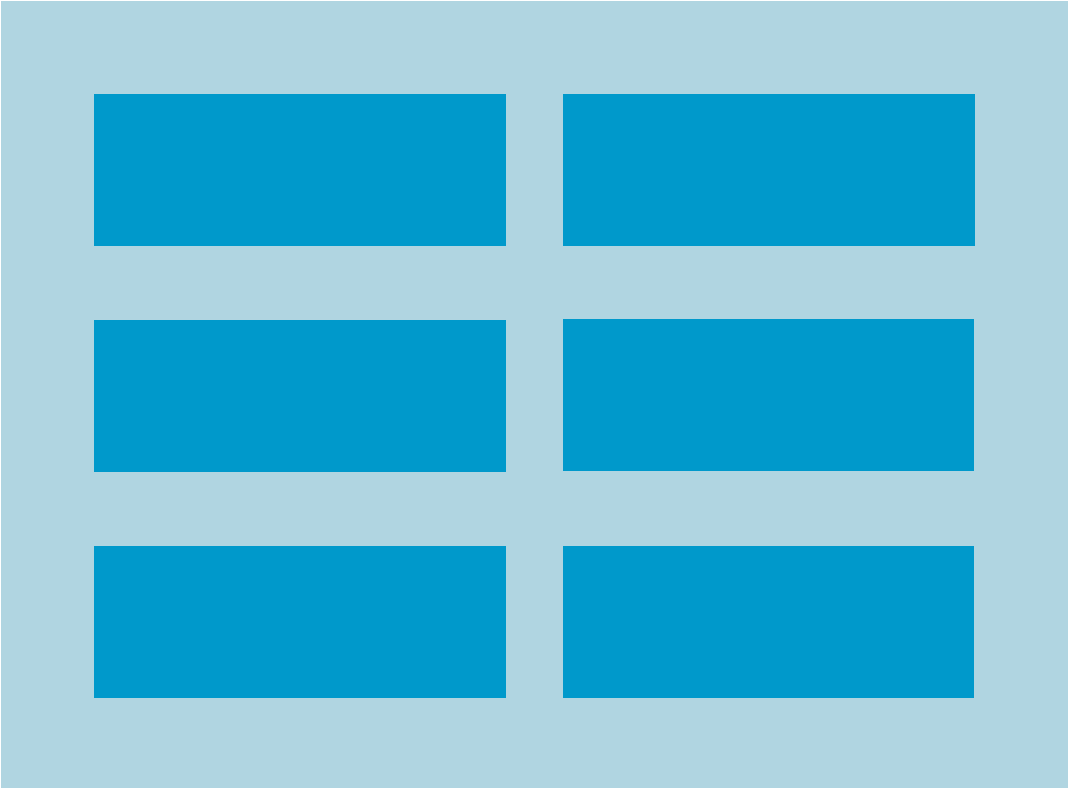
TableLayout

- Le view sono organizzate in maniera tabellare
- Tutte le view vengono distribuite in righe e colonne
- Si può specificare che una view deve occupare più colonne



GridLayout

- Le view sono organizzate in griglia
- Simile al TableLayout
- Si può specificare che un elemento occupi più righe o più colonne

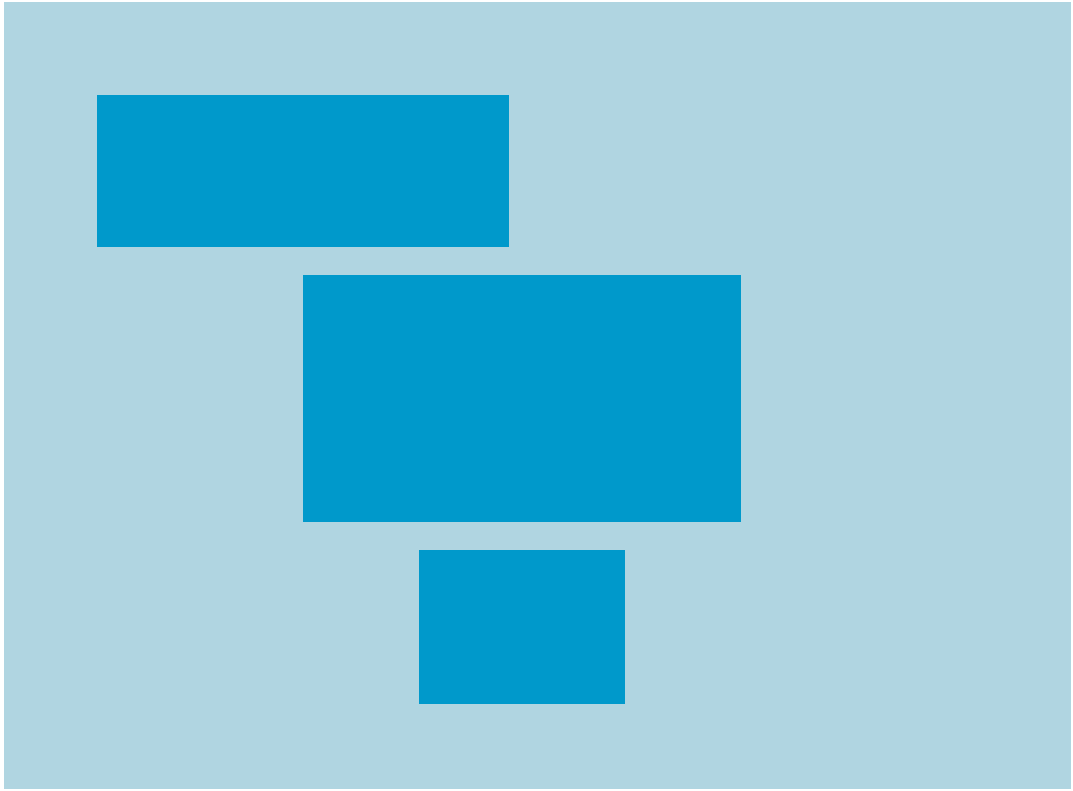


FrameLayout

- Si utilizza per mettere un placeholder
- Al suo interno, tipicamente verrà inserito un altro layout
- L'ordine delle View al suo interno definisce anche l'ordine di visualizzazione

AbsoluteLayout

- La posizione delle view viene specificata in maniera assoluta (x,y)
- Non è molto flessibile

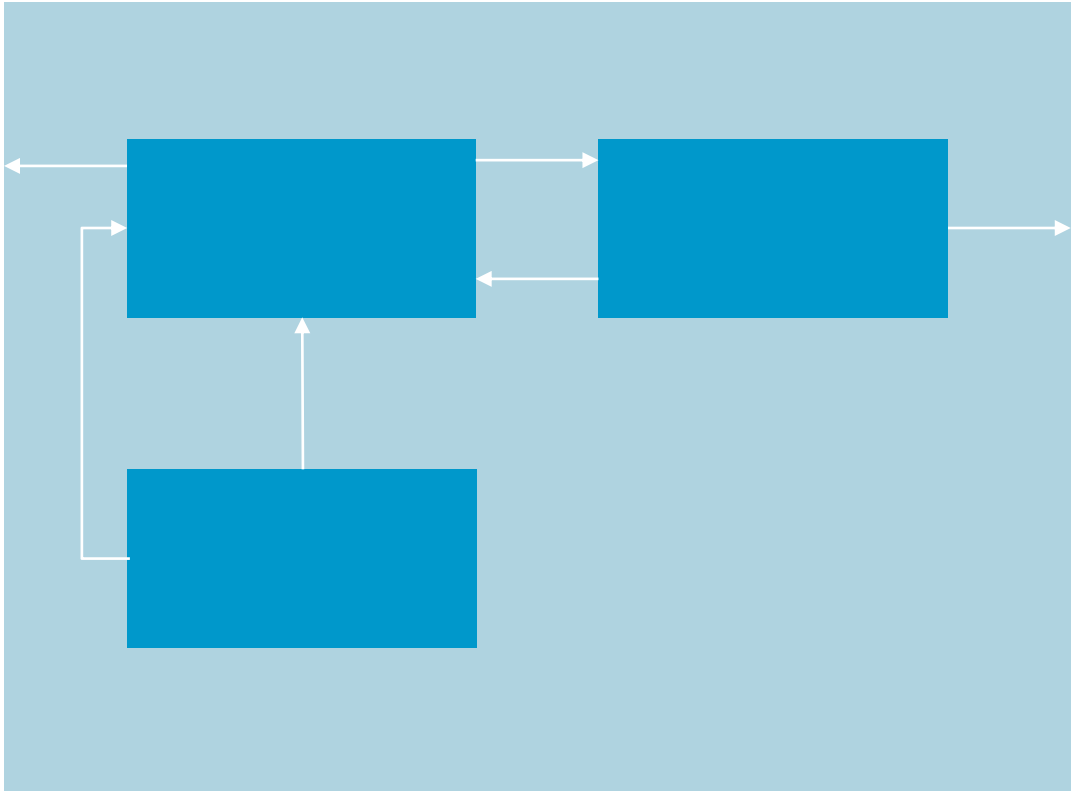


PercentRelativeLayout

- Estensione del RelativeLayout, ma possiamo utilizzare le percentuali per definire width, height e margin delle view
- Utilissimo per layout più complessi

ConstraintLayout

- Estensione del RelativeLayout, ma ogni view deve avere almeno due constraint, una per il posizionamento orizzontale e una per il verticale
- Estremamente versatile
- Estremamente performante
- Consigliato ufficialmente da Google



I Layout

- Possono essere annidati: immaginiamo un LinearLayout dentro un RelativeLayout
- Quando una Activity viene creata, è necessario specificare un Layout da utilizzare tramite il comando setContentView()
- Ogni Activity ha un FrameLayout generale, in cui viene inserito il Layout specificato tramite il comando setContentView()

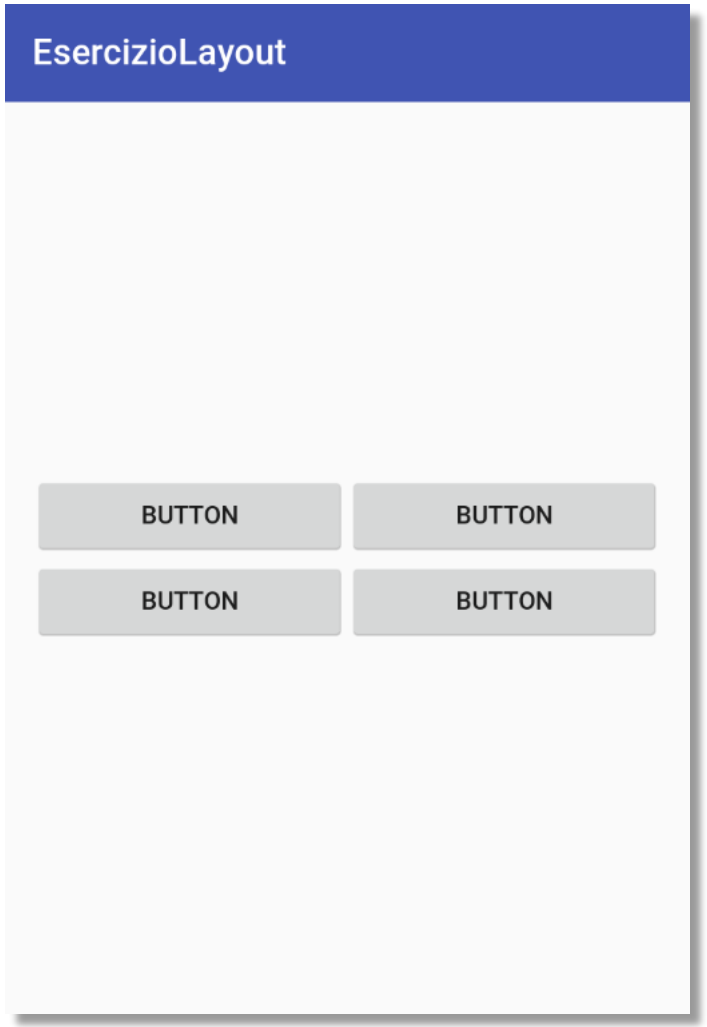
Alcuni attributi significativi

- orientation
- gravity e layout_gravity
- layout_weight e weightSum
- layout_align*
- layout_below e layout_above
- layout_center*
- layout_to*
- layout_column e layout_span
- rowCount e columnCount

FACCIAMO DEGLI ESEMPI PRATICI!

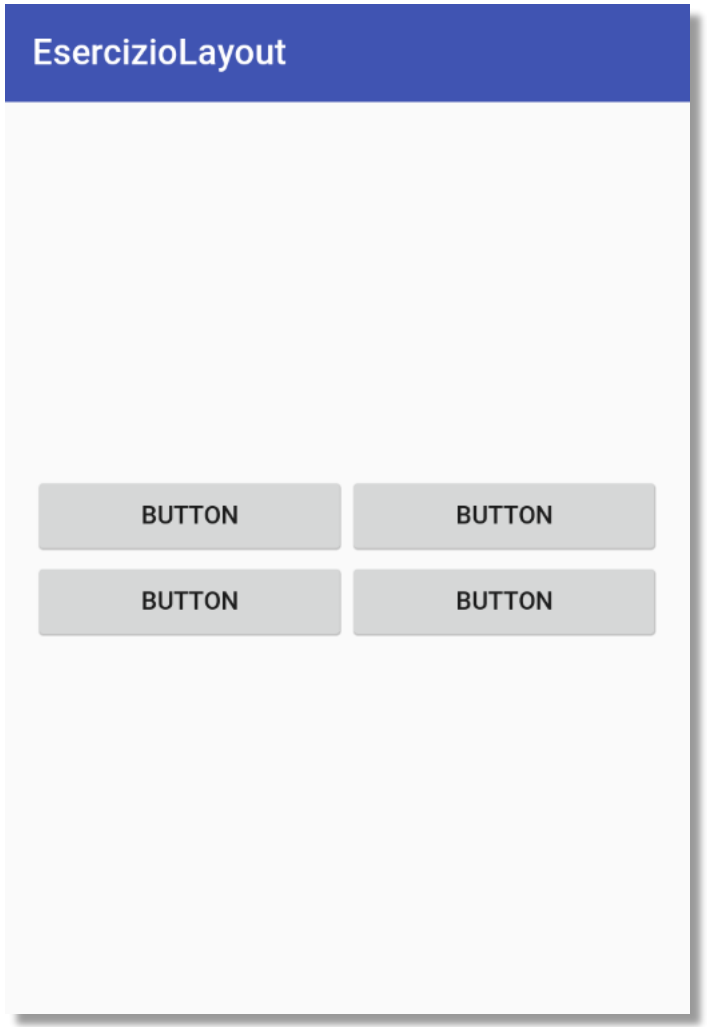
Esercizio

- Creare una APP con una sola activity, che contenga 4 button, disposti come in figura
- Utilizzare il relative layout per replicare la disposizione in figura



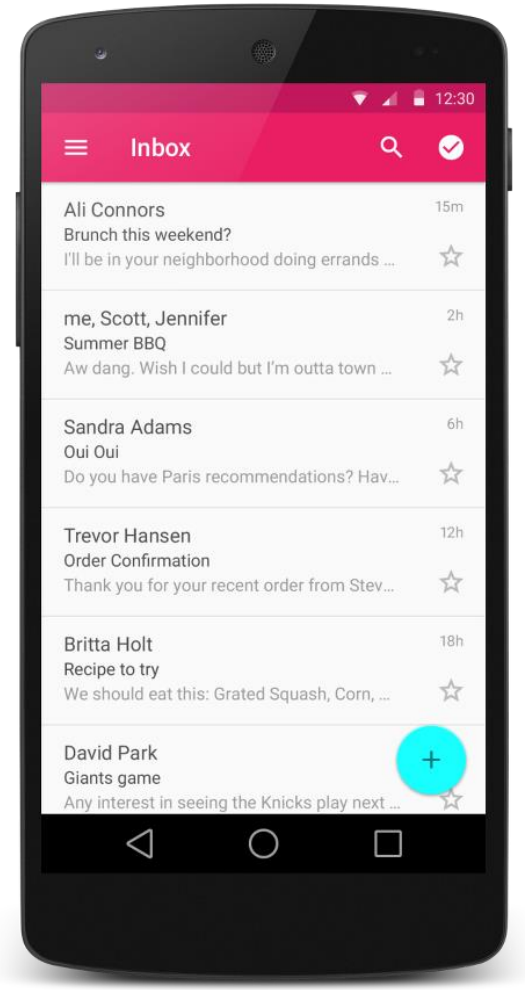
Esercizio

- Creare una APP con una sola activity, che contenga 4 button, disposti come in figura
- Utilizzare il table layout per replicare la disposizione in figura



E se volessimo una lista?

RecyclerView!



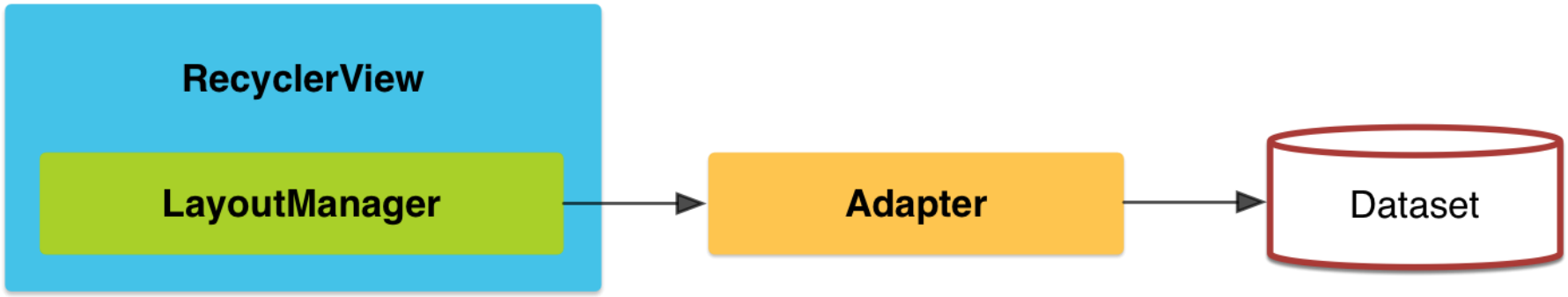
RecyclerView

- Sono una evoluzione delle ListView
- Offrono un insieme di funzionalità molto avanzate che possono essere poi personalizzate
- Animazioni
- Gestione della memoria ottimizzata
- Permettono di definire il layout per ogni elemento della lista
- Permettono di gestire molte interazioni con l'intera lista e con i singoli elementi della lista

RecyclerView

Hanno bisogno di:

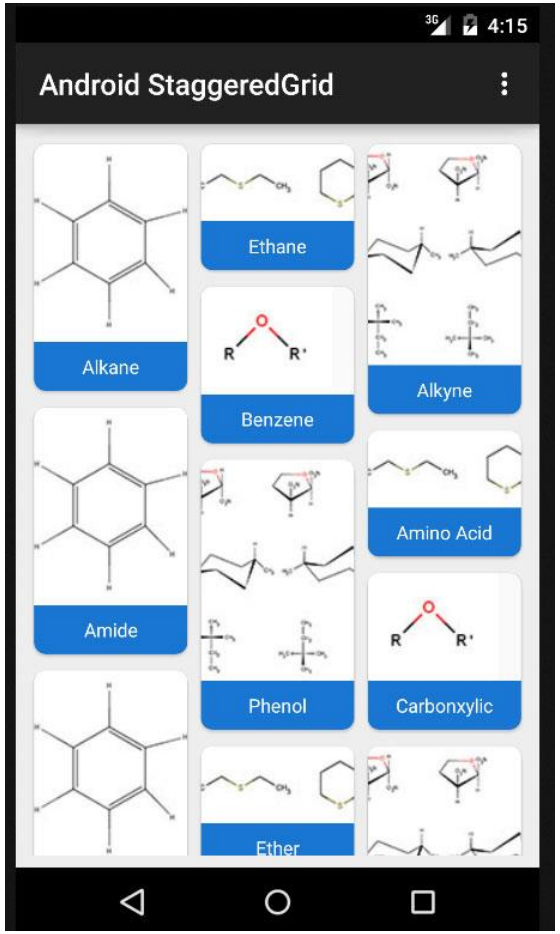
- un LayoutManager
- un Adapter
- dati



RecyclerView – LayoutManager

- Definisce come i singoli elementi della lista verranno disposti tra di loro
- LinearLayoutManager: gli elementi vengono disposti uno sotto l'altro (o accanto)
- GridLayoutManager: gli elementi vengono disposti in griglia
- StaggeredGridLayoutManager (??)

StaggeredGridLayoutManager



RecyclerView – Adapter

- Permette di fare la magia!
- Effettua il bind tra la UI e i dati
- Deve implementare l'interfaccia RecyclerView.Adapter<>
- Utilizza il pattern ViewHolder
- Imposta il layout per i singoli elementi della lista, e può impostare layout diversi da elemento a elemento

RecyclerView – Dataset

- Sono i dati che verranno visualizzati nella lista
- Possono essere statici o dinamici
- Possono essere un numero illimitato!!
- Potremmo caricarli dinamicamente da internet anche durante la consultazione della lista

TextView

- Widget standard
- Permette di visualizzare del testo statico

CREIAMO UNA LISTA!

Esercizio

- Creare un'APP: AppCapitoloDue
- L'APP deve contenere una Activity, al suo interno deve essere presente una lista di Insegnamenti. Per ogni insegnamento, la lista deve visualizzare il nome e il docente

- Suggerimenti:
  - Potete utilizzare gli array, gli arrayList, ecc
  - Dovete utilizzare una RecyclerView
  - Dovete creare un Adapter

Gli Intent

- Componenti che descrivono operazioni
- Permettono di dialogare con le Activity, con i Service e di inviare messaggi broadcast
- Diversi dagli eventi, perché questi ultimi sono legati ad una interazione fisica con la UI

Gli Intent

- Dato che tutte le applicazioni sono realizzate da Activity, possiamo richiamare un’altra applicazione tramite un Intent
- Gli Intent-Filter permettono di intercettare degli Intent
- Nel manifest, tramite un intent specifichiamo quale deve essere la Activity di partenza

Gli Intent

- Possono contenere informazioni aggiuntive, chiamate extra
- Gli extra vengono passati insieme all'Intent
- Chi riceve l'Intent, può accedere a tutti gli extra: possiamo far dialogare due Activity tramite gli extra!

Usiamo un Intent per avviare una seconda Activity!

Esercizio finale

- Creare un'APP: AppCapitoloDue
- L'APP deve contenere una prima Activity, al suo interno deve essere presente una lista di insegnamenti. Per ogni insegnamento, la lista deve visualizzare il nome e il docente.
- Il click su uno degli insegnamenti deve aprire una seconda Activity
- La seconda Activity deve contenere il nome, la descrizione, il docente, i CFU, l'AA, il semestre, il voto, ecc