RELAZIONE MOBILE PROGRAMMING

RecyclerView

Le RecyclerView sono uno strumento molto utile quando c’è bisogno di organizzare una lista di informazioni secondo uno schema preciso. Nell’applicazione, ogni elemento ha altre liste di elementi a lui collegate (es i fumetti, le serie e gli eventi collegati ad un eroe) che vanno quindi mostrati in maniera minimale all’utente.

L’utilizzo delle RecyclerView è stato essenziale, difatti in ogni activity di dettaglio sono state introdotte almeno 2 RecyclerView, per lo più utilizzate per contenere View gestite da un LinearLayoutManager con opzione di scorrimento orizzontale.

Inoltre, la RecyclerView è un oggetto che mantiene in una sua cache tutti gli elementi della lista che non possono essere mostrati all’utente (per motivi di spazio nel dispositivo) per poi andare a caricarli quando quest’ultimo effettua uno scroll il basso.

In questo modo è stato possibile gestire con successo le liste di elementi che venivano scaricate, che a volte potevano contenere anche più di 50 elementi.

Adapter

Ogni elemento della RecyclerView è rappresentato mediante un Layout indipendente, che deve essere gestito mediante appositi ViewHolder.

Ogni Holder è a sua volta gestito da un Adapter, che svolge la funzione di ponte fra la UI e i ViewHolder.

L’Adapter si occupa di collegare i dati agli Holder mediante il metodo onBindViewHolder(), andando quindi ad assegnare ogni Holder ad una specifica posizione nella lista. L’Adapter inoltre crea tutti i supporti necessari per la visualizzazione in base alle esigenze.

Nell’app, ogni View gestista dall’Holder è uno specifico elemento, che può essere un eroe, una serie, un fumetto o un evento, di cui sono mostrati un’immagine e il nome.

Ognuna delle View è inoltre clicabile, infatti l’Holder che la gestisce implementa il metodo di onClick(), andando a caricare la schermata di dettaglio per l’elemento selezionato dall’utente.

Menu contestuale (lo farà Adrian)

Per permettere all’utente di eliminare dalla lista dei preferiti un elemento, senza dover per forza aprire l’activity di dettaglio, è stato implementato un piccolo menù contestuale con la singola voce “elimina”, gestito dal FavouriteFragment.

Per far attivare il menu contestuale è stato usato il meccanismo dell’onLongClick(), che è stato implementato negli Holder gestiti dal FragmentAdapter. Ogni volta che viene effettuato un long press su uno dei preferiti, viene mostrato il menù contestuale e a questo punto l’utente può selezionare / deselezionare altri elementi con un semplice press ed eliminarli tramite la voce del menù, che si chiude subito dopo l’eliminazione.

Caching con Shared Preferences

Per poter mantenere sempre identici i suggeriti del giorno, è stato neccessario aggiungere un meccanismo di caching, per fare questo la scelta è ricaduta sulle Shared Preferences. Infatti, i dati da salvare consistevano in delle stringhe per ogni eroe, fumetto e serie e le SharedPreferences si prestano molto bene in quanto permettono il salvataggio basato su una coppia chiave-valore in un file.

Il metodo per ottenere il file su cui salvare i dati è getPreferences(), in quanto è necessario un solo file per contenere tutto, tale file viene resettato ogni giorno con i nuovi suggeriti del giorno