Naturalmente per interfacciarsi con il database messo a disposizione dal sito della Marvel occorre una continua disponibilità della connessione ad Internet. Questo infatti, insieme alla possibilità di controllare lo stato della connessione, sono infatti stati gli unici permessi che si sono resi necessari nello sviluppare la nostra app.

Essendo entrambi permessi normali, in quanto il rischio per la privacy dell’utente o per il funzionamento delle altre app è minimo, essi vengono automaticamente concessi dal sistema operativo e l’utente non può revocarli. Perciò non è necessario né chiedere all’utente l’autorizzazione esplicita e né tantomeno inserire nel codice controlli sulla loro effettiva autorizzazione da parte dell’utente.

Al fine di verificare tale connessione abbiamo aggiunto un controllo prima di effettuare ogni query che appunto invocasse il sistema operativo per ottenere una risposta a riguardo. Nel caso la connessione non risultasse disponibile abbiamo scelto di avvertire l’utente facendo comparire un toast sullo schermo che lo invitasse a verificare la sua connessione. La scelta del toast è stata fatta in quanto era la più semplice e meno invasiva per l’utente poiché l’errore, seppure di notevole rilevanza, può essere semplicemente risolto dall’utente stesso abilitando la connessione dati o il wifi, senza richiedere un riavvio dell’applicazione o altro.

Un altro controllo, anche questo gestito tramite un toast da mostrare all’utente, si è reso necessario in cui le 3000 query giornaliere a disposizione verso il server della Marvel dovessero esaurirsi.

Dato il frequente ricorso ai toast abbiamo deciso di creare una classe apposta che si occupasse di mostrare questi messaggi sullo schermo, avendo quindi attenzione di passargli di volta in volta il messaggio corretto e il context appropriato riferito all’activity in esecuzione in quel momento.

Data la frequente attesa di dati, dovendo quest’ultimi essere reperiti real time da un database esterno, abbiamo deciso di implementare una animazione che si occupasse di rendere noto all’utente il loading dei dati. L’animazione di per sé è molto semplice, richiedendo semplicemente che ogni schermata in cui risultasse necessaria, abbia una view chiamata ProgressBar. Abbiamo deciso quindi che all’apertura di una nuova activity e quindi il caricamento di un nuovo layout, questo presentasse solo la ProgressBar come view visibile. A seguito che i dati vengano recuperati e le varie recyclerView venissero riempite correttamente, si incrementa un contatore. Quando questo contatore raggiunge un certo valore, che dipende dal numero di recyclerView per quella schermata, si attiva un metodo che imposta la visibilità della ProgressBar a GONE e quella del resto della schermata in VISIBLE.