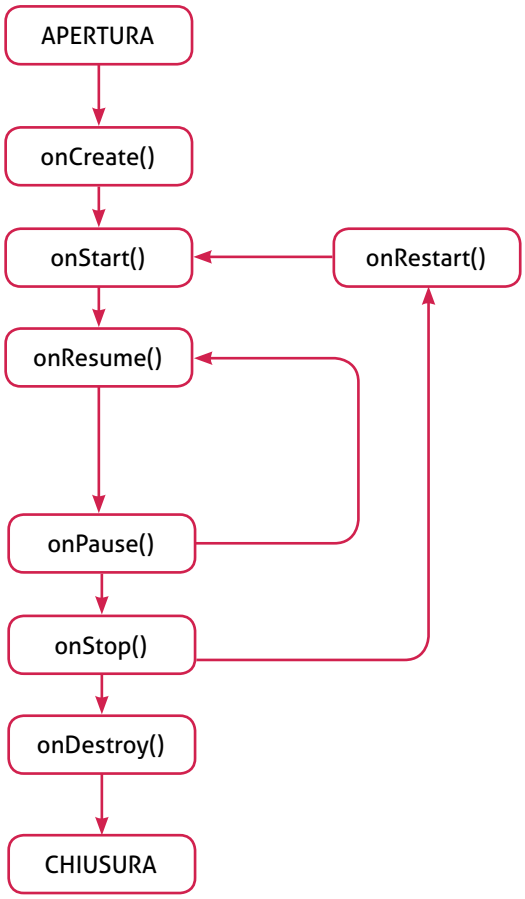
Il ciclo di vita delle *Activity*

L'*Activity* descrive la modalità con cui l'*app* interagisce con l'utente, in particolare gestisce la finestra su cui vengono posizionate tutte le componenti grafiche. Ogni *app* può essere composta da più videate e quindi da più *Activity*. La videata principale si chiama **Main Activity** e viene visualizzata all'avvio dell'applicazione. Le altre *Activity* che vengono aperte durante l'esecuzione dell'*app* si posizionano sopra la videata precedentemente visualizzata. Il pulsante *Back* chiude l'*Activity* corrente e visualizza l'*Activity* precedente.

Ogni *Activity*, da quando viene aperta a quando viene chiusa, segue un ciclo di vita che la fa passare tra diversi stati. Ad ogni passaggio di stato il sistema Android richiama l'esecuzione di un particolare metodo (chiamato **callback**).

Il diagramma a fianco mostra la sequenza con cui i metodi vengono eseguiti dall'apertura dell'*Activity* fino alla sua chiusura.



I metodi che descrivono il ciclo di vita dell'*Activity* sono elencati nella seguente tabella.

Metodo	Descrizione
onCreate()	Eseguito durante la creazione dell'*Activity*. Contiene solitamente le istruzioni per la creazione dell'interfaccia grafica.
onRestart()	Eseguito prima che l'*Activity* venga avviata dopo un precedente stop.
onStart()	Eseguito quando l'*Activity* diventa visibile all'utente.
onResume()	Eseguito quando l'utente può interagire con l'*Activity*.
onPause()	Eseguito quando l'utente non può più interagire con l'*Activity* perché Android ne sta aprendo un'altra.
onStop()	Eseguito quando l'*Activity* passa in secondo piano.
onDestroy()	Eseguito quando l'*Activity* viene chiusa.

Una *Activity* che implementa tutti i precedenti metodi può essere descritta con la seguente classe.

```
public class MainActivity extends Activity {

  @Override
  protected void onCreate (Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
  }

  @Override
  protected void onRestart() {
    super.onRestart();
  }

  @Override
  protected void onStart() {
    super.onStart();
  }

  @Override
  protected void onResume() {
    super.onResume();
  }

  @Override
  protected void onPause() {
    super.onPause();
  }

  @Override
  protected void onStop() {
    super.onStop();
  }

  @Override
  protected void onDestroy() {
    super.onDestroy();
  }
}
```

All'interno di ogni metodo, come prima operazione è opportuno richiamare il corrispondente metodo della sopraclasse, successivamente si possono aggiungere le istruzioni per gestire correttamente il passaggio di stato.