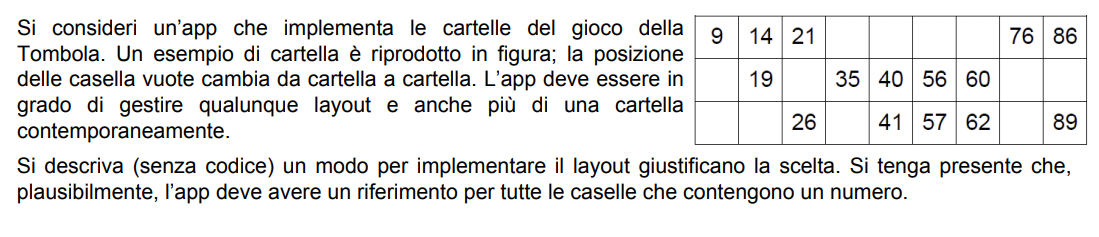
**Esame 2017-01-24**

**Domanda 1**

****

**Domanda 2**

**Si spieghi l’interazione che avviene fra un ListView e l’ArrayAdapter ad esso associato, possibilmente arricchendo la spiegazione con frammenti di codice.**

Gli elementi di una lista vengono memorizzati in un array, l’adapter fornisce gli elementi da visualizzare in base allo scorrimento effettuato dall’utente.

L’arrayAdapter puo essere inizzializzato con:

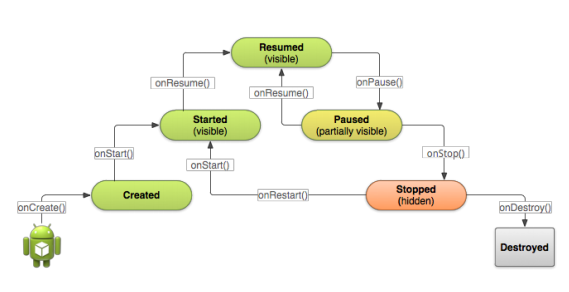
ArrayAdapter<Tipo di dato> arrayAdapter=new ArrayAdapter<Tipo di dato>(context,R.layout.list\_element,R.id.textViewList,array)

Una volta inizzializzato l’arrayAdapter deve essere associato al ListView:

listView.setAdapter(arrayAdapter);

**Domanda 3**

**Si descriva il ciclo di vita delle Activity. L’intero ciclo di vita va dall’esecuzione del metodo onCreate() all’esecuzione del metodo onDestroy(). Da quale a quale metodo vanno invece il periodo di “visibilità” ed il periodo di “foreground” dell’activity?**

****

Un app e nel periodo di foreground nello stato Started, Resumed e Paused

**Domanda 4**

**Il metodo startActivityForResult(Intent, int) permette di lanciare una nuova activity ed alla sua terminazione di recuperare un valore di ritorno ed ulteriori dati inseriti in un intent attraverso il metodo onActivityResult(int, int, Intent). Suppone che l’intent non sia sufficiente né per passare i dati all’activity chiamata né per per memorizzare i dati da restituire all’activity chiamante; per sfruttando i suddetti metodi, come risolveresti il problema?**

Per ovviare a questo problema possiamo usare il data storage, ovvero usando Shared Preferences oppure File, possiamo salvare i dati in modo privato per l’app.

Un esempio puo essere quello di usare gli shared preferences, ovverro salviamo i dati con gli shared preferences all’interno di un file xml. Tale fail sara accessibile a tutte le activity dell’app.

In questo modo possiamo lanciare l’altra activity con startActivityForResult e ne metodo onActivityResult andiamo a leggere i dati salvati con lo shared preferences.

**Domanda 5**

**Si faccia un esempio di risoluzione implicita di un intent spiegando cosa succede per arrivare all’esecuzione della relativa activity.**

Per lanciare una nuova activity con l’utilizzo di un intent implicito non sara indicata la classe dell’activity che si vuole lanciare ma android ne scegliera una adatta in base all’:

-Action

-Type

-Uri

-Category

Esempio:

Intent geoIntent = new Intent(android.content.Intent.ACTION\_VIEW, Uri.parse("geo:0,0?q=" + address));

startActivity(geoIntent);

In questo caso verra avviato l’activity che potra lavorare sul parametro addres (Esempio: Maps)

**Domanda 6**

**Che cosa è e a cosa serve l’oggetto Canvas? Si illustri, con sufficienti dettagli, un esempio in cui è necessario utilizzare un oggetto Canvas**

Un oggetto canvas e una grafica complessa con aggiornamenti frequenti, per disegnare il canvas si utilizza il metodo onDraw(Canvas canvas).

Un esempio in cui e necessario utilizzare un oggetto Canvas e un app in cui ce uno spartito musicale dove dobbiamo inserire delle note. Utilizziamo i canvas per disegnare e inserire note sul pentagramma.

**Domanda 7**

**Si scriva un file xml per la seguente animazione di un oggetto drawable:**

**1. rotazione di 2 giri completi a destra, dal tempo 0 al tempo 2 sec**

**2. traslazione di 300px a destra, dal tempo 3 sec al tempo 4 sec,**

**3. rotazione di 2 giri completi a sinistra, dal tempo 3 sec al tempo 5 sec.**

<**rotate  
 android:startOffset="0"  
 android:duration="2000"  
 android:fromDegrees="0"  
 android:toDegrees="720"  
 android:pivotX="80%"  
 android:pivotY="80%"** />

<**translate  
 android:startOffset="3000"  
 android:duration="1000"  
 android:fromXDelta="0"  
 android:fromYDelta="0"  
 android:toXDelta="300"  
 android:toYDelta="0"  
 android:fillAfter="true"** />

<**rotate  
 android:startOffset="3000"  
 android:duration="2000"  
 android:fromDegrees="0"  
 android:toDegrees="-720"  
 android:pivotX="80%"  
 android:pivotY="80%"** />

**Domanda 8**

**Che cosa è un oggetto MotionEvent? Si forniscano dettagli sul suo utilizzo.**

Il MotionEven rappresenta un movimento registrato da una periferica penna,trackball,mouse e dita sul display. Il movimento è rappresentato con ACTION\_CODE(cambiamento avvenuto) e ACTION\_VALUES(posizione e proprietàa del movimento)

Gli ACTION\_CODE possono essere:

– ACTION\_DOWN

un dito tocca lo schermo ed è il primo

– ACTION\_POINTER\_DOWN

un dito tocca lo schermo ma non è il primo

– ACTION\_MOVE

un dito che è sullo schermo si muove

– ACTION\_POINTER\_UP

un dito che è sullo schermo non lo tocca più

– ACTION\_UP

l’ultimo dito sullo schermo viene alzato

**Domanda 9**

**Cosa deve fare un app per utilizzare un sensore? Si spieghi possibilmente utilizzando opportuno snippet di codice.**

Molti smartphone, tablet hanno sensori di movimento, di ambiente, di posizione. I sensori forniscono dati “grezzi” e l’accuratezza dei dati dipende dalla qualita.

Un SensorManager ci dice i sensori disponibil e le caratteristiche del singolo sensore, inoltre ci permette di leggere i dati del sensore e usare Listners sui cambiamenti dei dati.

**public class** SensorActivity **extends** Activity **implements** SensorEventListener {

**private** SensorManager **sensorManager**;  
**private** Sensor **accelerometer**;

**protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);

**sensorManager** = (SensorManager) getSystemService(***SENSOR\_SERVICE***);  
 **accelerometer** = **sensorManager**.getDefaultSensor(Sensor.***TYPE\_ACCELEROMETER***);

}

**protected void** onResume() {  
 **super**.onResume();  
 **sensorManager**.registerListener(**this**, **accelerometer**,SensorManager.***SENSOR\_DELAY\_UI***);  
 **lastUpdate** = 0;  
}

**protected void** onPause() {  
 **sensorManager**.unregisterListener(**this**);  
 **super**.onPause();  
}

**public void** onSensorChanged(SensorEvent event) {

**if** (event.**sensor**.getType() == Sensor.***TYPE\_ACCELEROMETER***) {

**if** (now - **lastUpdate** > ***INTERVALLO***) {  
 **float** x,y,z;  
 x = event.**values**[0];  
 y = event.**values**[1];  
 z = event.**values**[2];  
  
 aggiorna\_media(x,y,z);  
  
 **textViewX**.setText(String.*format*(**"%7.4f"**, x));  
 **textViewY**.setText(String.*format*(**"%7.4f"**, y));  
 **textViewZ**.setText(String.*format*(**"%7.4f"**, z));

}

}

**public void** onAccuracyChanged(Sensor sensor, **int** accuracy) {  
 *// non ci serve fare niente, dobbiamo scriverlo perchè questo metodo  
 // è richiesto dall'implementazione di SensorEventListener*}

}

**Domanda 10**

**Quali sono i principali modi per fornire notifiche all’utente? Si fornisca una breve descrizione.**

I principali modi per fornire delle notifiche all’utente sono:

* Toast: breve messaggio temporaneo
* Dialog: ovvero una notifica che permette l’iterazione con l’utente
* Notifiche: status bar e cassetto delle notifiche
  + Messaggi: che spuntano sulla barra di stato come icone e una volta aperto il cassetto delle notifiche sara visibile un testo aggiuntivo. Saranno visibili fino a quando l’utente non li cancellera o revocati dall’app