Esercitazione 5

Prima parte - ToggleButton ed ImageView

1) Creare un progetto Android di nome SwitchImage e dare nome alla activity di default "SwitchImageActivity"

2) Creare una cartella drawable ed inserire le seguenti due immagini:
    http://cdn1.iconfinder.com/data/icons/multimedia_iconset/128/button_play.png
    http://cdn1.iconfinder.com/data/icons/multimedia_iconset/128/button_pause.png

3) Modificare il Layout eliminando la textview aggiungendo un ToggleButton ed una ImageView impostando come src button_play

4) nel metodo onCreate
    i) prendere il riferimento alla ImageView e salvarlo in una variabile di istanza iv

    ii) Prendere il riferimento al ToggleButton e creare un OnCheckedChangeListener che imposti l'immagine in base allo stato:

```
if(isChecked){
    iv.setImageResource(R.drawable.button_pause);
}else{
    iv.setImageResource(R.drawable.button_play);

}
```

5) Eseguire e testare il programma

Seconda Parte - Drag di una immagine in una View Custom

1) Creare un progetto Android di nome DragImage e dare nome alla activity di default "DragImageActivity"

2) Creare una Classe "MyView" che estende la classe View e implementare il costruttore che ha come parametro il Context

3) Definire come variabili di istanza la x e la y dell'immagine da posizionare ed il riferimanto alla bitmap ed una variabile booleana per gestire lo stato di drag

```
private int x=100;
private int y=100;
private Bitmap img=null;
private boolean dragOn=false;
```

4) nel costruttore creato inizializzare la bitmap da una risorsa drawable

```
BitmapFactory.Options opts = new BitmapFactory.Options();
opts.inJustDecodeBounds = true;
img = BitmapFactory.decodeResource(context.getResources(),
R.drawable.ic_launcher);
```

5) Sovrascrivere il metodo onDraw della bitmap per disegnare l'immagine

```
@Override
protected void onDraw(Canvas canvas) {
        canvas.drawBitmap(img, x, y, null);
}
```

6) Impostare la nuova view come root element della Activity e eseguire il programma

    ...

```
7) Riscrivere il metodo onTouchEvent
     i) Prendendo il tipo di evento e la posizione
          int eventaction = event.getAction();
        int touchx = (int)event.getX();
        int touchy = (int)event.getY();

     ii) Impostando uno switch che gestisca gli eventi di ACTION_DOWN, ACTION_MOVE,
ACTION_UP
                  switch (eventaction ) {
                       case MotionEvent.ACTION_DOWN: // touch down so check if the finger
is on a ball
                                  ...
                            break;
                       case MotionEvent.ACTION_MOVE:   // touch drag with the ball
                             ...
                           break;

                       case MotionEvent.ACTION_UP:
                             ...
                            break;
                       }
     iii) in caso di ACTION_DOWN capire se è stata cliccata l'immagine

8) gestione dei tre eventi:
     i) evento MotionEvent.ACTION_DOWN - attivare il drag
               if (tx > x & tx > x + img.getWidth() & ty > y
                        & ty < y + img.getHeight()) {
                    dragOn = true;
               }
     ii) evento MotionEvent.ACTION_MOVE - spostare l'immagine e ridisegnare il quadro
               x = tx;
               y = ty;
               invalidate();
     iii) evento MotionEvent.ACTION_UP - finire il drag
               dragOn = false;

9) Eseguire il programma e testarlo

[10) L'offset del click rispetto all'angolo in alto a sx dell'immagine mediante due
variabili di istanza offsetx e offsety]
```

```
Terza Parte - Drag di una view

1) Creare un progetto Android di nome DragView e dare nome alla activity di default
"DragViewActivity"

2) Creare una cartella drawable ed inserire le seguenti due immagini:
     http://cdn1.iconfinder.com/data/icons/Car_Icon_Set_BevelAndEmboss-Net/53/lorry.png
     http://cdn1.iconfinder.com/data/icons/Car_Icon_Set_BevelAndEmboss-Net/53/van.png

3) Modificare il main.xml eliminando la textview ed il linearlayout ed aggiungendo un
RelativeLayout ed una ImageView.

4) Impostare come src della ImageView l'immagine "lorry" ed eliminare gli eventuali
attributi di posizionamento del RelativeLayout
```

5) Creare tre variabili private di istanza per memorizzare la view quando selezionata e l'offset del click rispetto all'angolo in alto a sinistra della view (ImageView) selzionata

```
    private View selected_item = null;
    private int offset_x = 0;
    private int offset_y = 0;
```

4) Nel metodo onCreate

i) Prendere il riferimento alla ImageView ed impostare un listener per l'evento di touch che imposti le tre varibili di istanza all'inizio del touch (MotionEvent.ACTION_DOWN) e ritorni false per propagare l'evento

```
        ImageView iv = (ImageView) findViewById(R.id.imageView1);
        iv.setOnTouchListener(new View.OnTouchListener() {

            @Override
            public boolean onTouch(View v, MotionEvent event) {
                if(event.getActionMasked()==MotionEvent.ACTION_DOWN){
                    offset_x = (int) event.getX();
                    offset_y = (int) event.getY();
                    selected_item = v;
                }
                return false;
            }
        });
```

ii) Prendere il riferimento al relative layout ed impostare un listener per l'evento di onTouch

```
            RelativeLayout rl = (RelativeLayout) findViewById(R.id.relativeLayout1);
            rl.setOnTouchListener(new View.OnTouchListener() {

                @Override
                public boolean onTouch(View v, MotionEvent event) {

                    return true; // Evento consumato
                }
            });
```

5) nel metodo onTouch

i) contollare se esiste una view selezionata; altrimenti ritornare

```
        if(selected_item==null) return false;
```

ii) Controllare se è un evento di movimento (del tocco o mouse) - MotionEvent.ACTION_MOVE,

```
        if(event.equals(MotionEvent.ACTION_MOVE)){
```

iii) calcolare la x e la y per la view

```
                    int x = (int) event.getX() - offset_x;
                    int y = (int) event.getY() - offset_y;
```

iv) usare le coordinate come margini usando un RelativeLayout.LayoutParams per impostarle nella view

```
                    ViewGroup.LayoutParams lp = new ViewGroup.LayoutParams(
                              new ViewGroup.MarginLayoutParams(
                                        RelativeLayout.LayoutParams.WRAP_CONTENT,
                                        RelativeLayout.LayoutParams.WRAP_CONTENT));
                    lp.setMargins(x, y, 0, 0);
                    selected_item.setLayoutParams(lp);
```

v) Controllare se è un evento di rilascio del tocco - MotionEvent.ACTION_UP e deselezionare la view

```
                if(event.equals(MotionEvent.ACTION_UP)){
                    selected_item=null;
                }
```

6) Eseguire e testare il codice

7) Modificare il metodo onTouch per limitare il movimento dell'immagine prima di impostare i margini (determinare il valore opportuno)

```
int w = getWindowManager().getDefaultDisplay().getWidth() -128;
int h = getWindowManager().getDefaultDisplay().getHeight()-128;
if (x > w)
     x = w;
if (y > h)
     y = h;
```

8) Eseguire e testare il codice

[9) Aggiungere la seconda immagine in una imageview ed attivare il drag anche su quella]