Esercizi Terza Parte

Gestione avanzata degli Eventi ed Intent Impliciti

Esercizio 1

Si crei una empty activity con le Views mostrate in figura:

L'EditText è in ascolto sulle modifiche del testo inserito dall'utente.

Quando si rileva che il testo inserito è un URL HTTP corretto (si supponga per semplicità che abbia l'unico vincolo di iniziare con "http"), l'attributo "text" di Button deve cambiare in "Apri Browser". Diversamente lo stesso attributo diventi "Scatta foto".

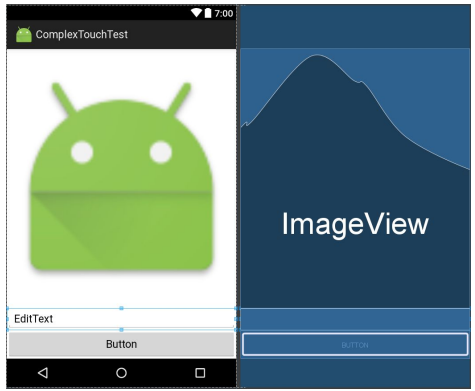
Al click del pulsante, si scatenino quindi i due eventi corrispondenti tramite intent impliciti (1, 2 a lato). Aprire una pagina web NON crea un bundle di risposta, mentre scattare una foto sì:

```
@Override
protected void onActivityResult(int requestCode, int resultCode,
Intent data) {

   if (requestCode == COD_REQ && resultCode == Activity.RESULT_OK) {
       Bundle extras = data.getExtras();
       Bitmap imageBitmap = (Bitmap) extras.get("data");
           imageView.setImageBitmap(imageBitmap);
   }
}
```

```
1) Uri uri =
Uri.parse("http://www.unimore.it");
Intent it = new Intent(Intent.ACTION_VIEW,uri);
startActivity(it);
```

```
2) Intent intent = new
Intent(MediaStore.ACTION_IMAGE_CAPTURE);
   activity.startActivityForResult(intent,
COD_REQ);
```



Esercizio 2

Si crei una empty activity che presenti una TextView al centro dello schermo.

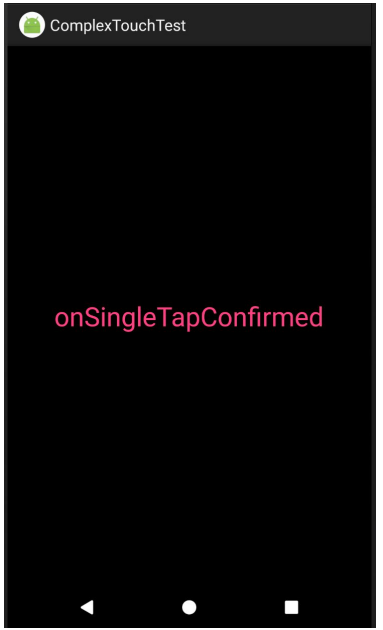
L’activity implementa le interfacce **GestureDetector**.**OnGestureListener** e **GestureDetector**.**OnDoubleTapListener**

Si leghi il double tap listener e il gesture detector all’intera activity.

L’utente provi le diverse “Gesture” sulla sua applicazione...

Il testo della TextView deve mostrare in quale callback ci troviamo tra tutte quelle implementate.

Perchè non sempre compaiono tutte?



Esercizio 3

Si parta da una Empty Activity.

Tale activity è composta da un LinearLayout verticale come sfondo a cui vengono aggiunti due Relative Layout di larghezza “match_parent” e di altezza pari alla metà dello schermo (interrogare il WindowManager).

A ciascun Relative Layout venga inizialmente assegnato un colore diverso. Abbiamo quindi un layout superiore ed uno inferiore: Vedere figura 1.

L’intera activity è in ascolto con le stesse interfacce dell’esercizio precedente, ma reagisca solo nei seguenti eventi:

Tap singolo: a caso, uno dei due relative layout cambia colore. Quest’ultimo colore deve essere casuale.

Tap doppio: come sopra, ma entrambi i layout cambiano colore.

Scroll : se lo scroll è dall’alto verso il basso, il layout superiore aumenta la sua altezza di 10 unità, mentre il layout inferiore decrementa di 10 unità. Viceversa se la direzione di scrolling è dal basso verso l’alto. (Fig: 2 e 3) [suggerimenti nella prossima slide]

Fig 1. Layout superiore (verde) ed inferiore (marrore)

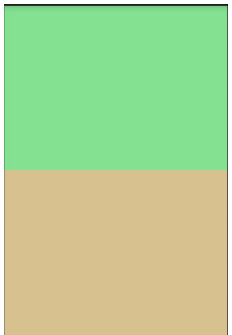


Fig 2. Scroll verso il basso

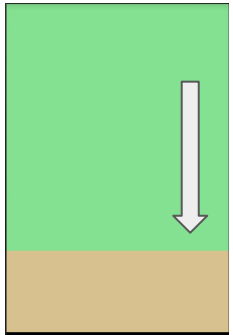
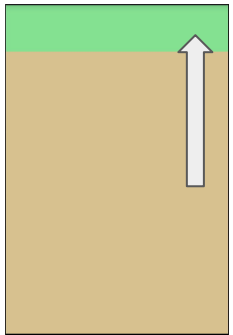


Fig 3. Scroll verso l’alto



Suggerimenti per esercizio precedente

Chiedere al WindowManager informazioni sulle dimensioni dello schermo:

```
Display display = getWindowManager().getDefaultDisplay();
Point size = new Point();
display.getSize(size); //in size abbiamo xMax ed yMax del display

//sono attributi pubblici della classe Point, chiamati “x” ed “y”
```

Get/Set altezza & larghezza di un layout:

```
ViewGroup.LayoutParams p = <refLayout>.getLayoutParams();
//in p.height e p.width troviamo le dimensioni correnti; possiamo cambiarle...
<refLayout>.setLayoutParams(p);
```