Android Development

Esercitazione Android 1
MyPoiAdvisor

Funzionalità Applicazione

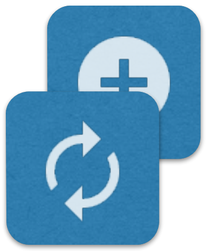
- L'applicazione consentirà all'utente di:
  - Visualizzare una lista di Punti di Interesse (Point of Interest - POI) sotto forma di Lista e Mappa
  - Aggiungere un nuovo punto di interesse specificando le seguenti informazioni:
    - Nome
    - Indirizzo
    - Latitudine
    - Longitudine
    - Type

Material design

Specifiche

- Material Design Theme
- Target SDK: Google API 21
- Compile With SDK: Google API 21
- Minimum Required SDK: Google API 8
- Usare il seguente set di icone (icons.zip):


Icona App

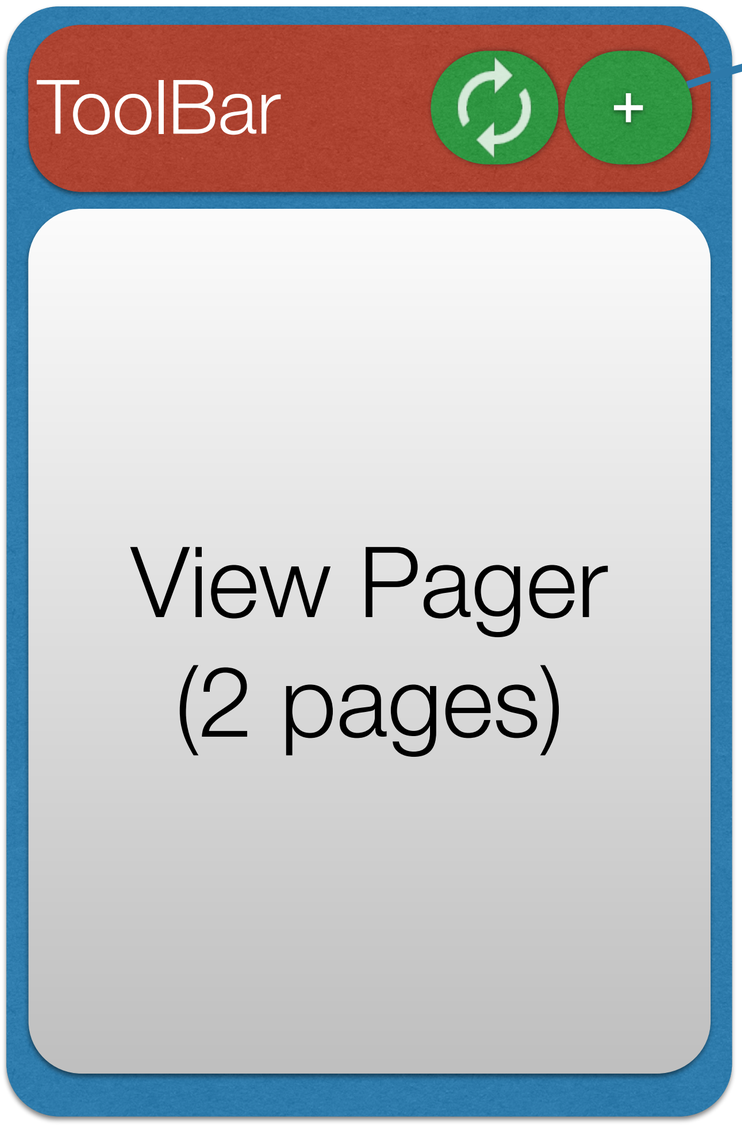

ActionBar
Actions


Icone
Elementi
Lista

http://romannurik.github.io/AndroidAssetStudio/index.html

Struttura Applicazione

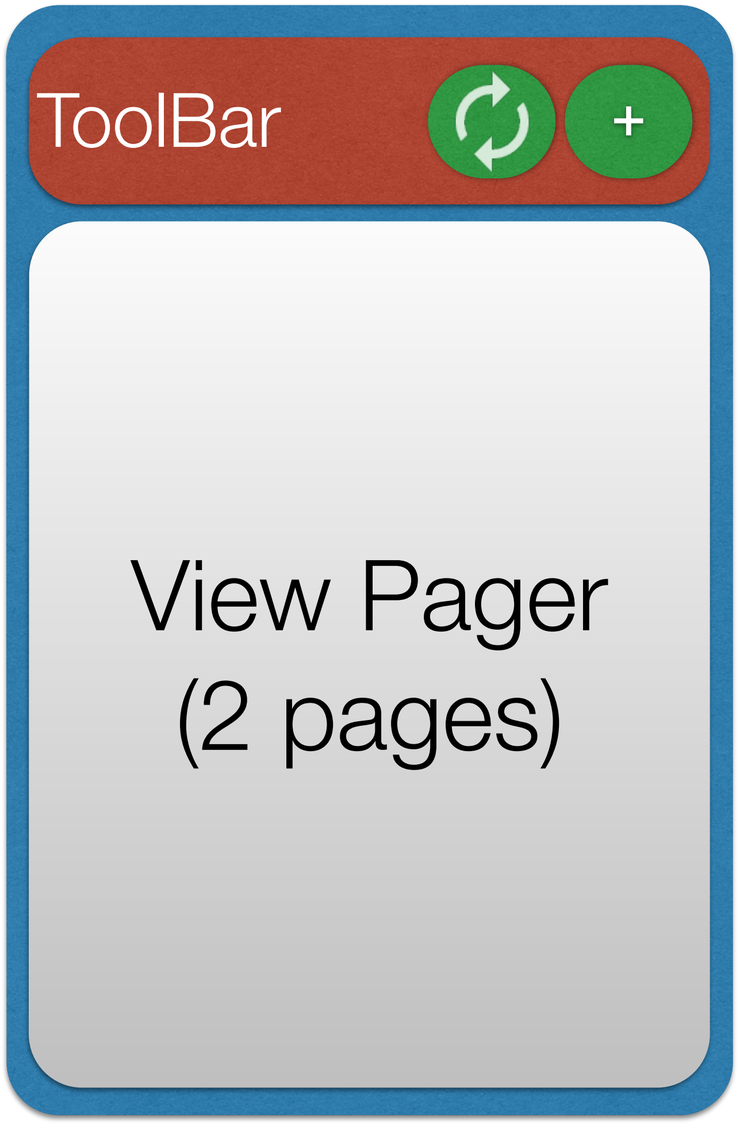
MainActivity



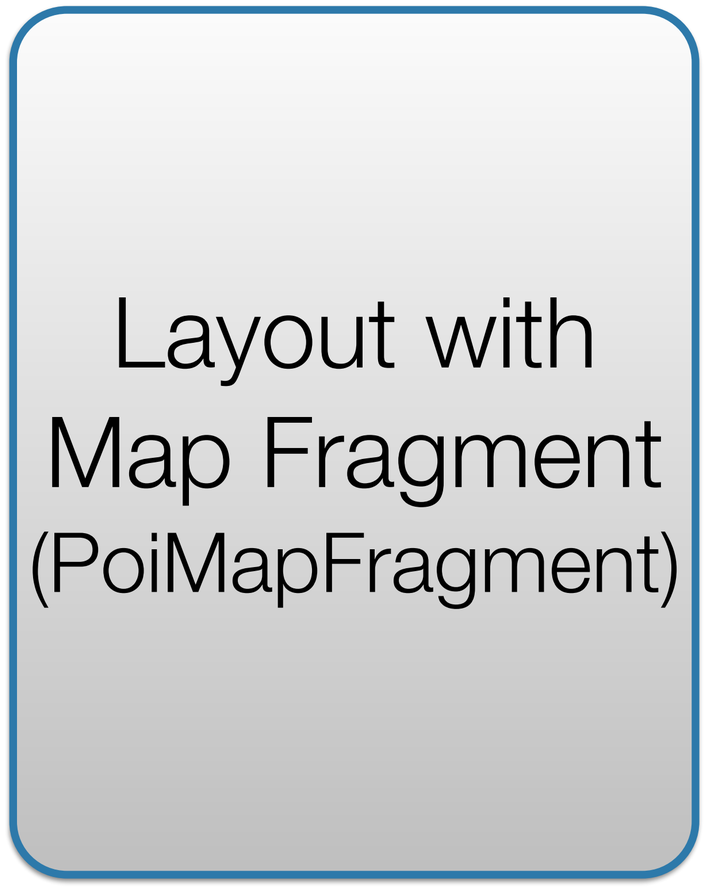
AddPoiActivity

Struttura Applicazione

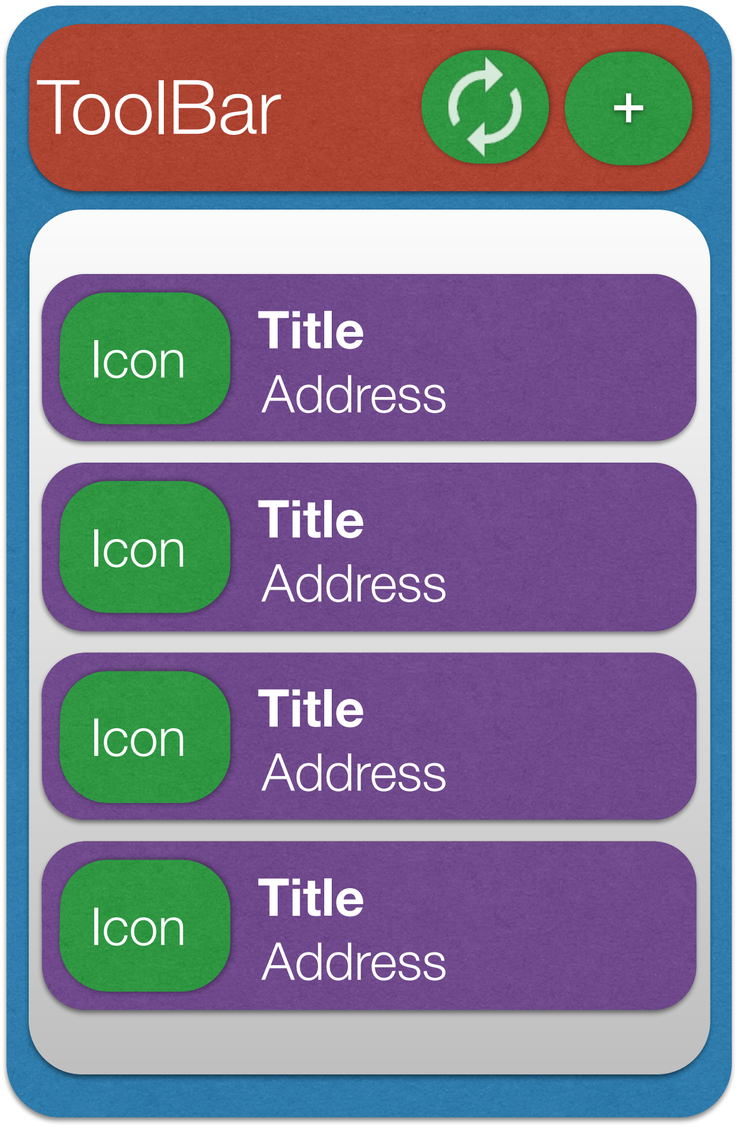
MainActivity



ViewPager



PoiDataManager



Icona personalizzata
in base alla tipologia del PoI

PoiDescriptor

```
public class PoiDescriptor {

    public static String POI_TYPE_GENERIC = "generic";
    public static String POI_TYPE_SCHOOL = "school";
    public static String POI_TYPE_GAS_STATION = "gas_station";
    public static String POI_TYPE_BUS_STOP = "bus_stop";
    public static String POI_TYPE_CINEMA = "cinema";
    public static String POI_TYPE_STORE = "store";

    private String type = null;

    private String name = null;

    private String Address = null;

    private double lat = 0.0;

    private double lng = 0.0;

    public PoiDescriptor(String type, String name, String address, double lat,
            double lng) {
        super();
        this.type = type;
        this.name = name;
        Address = address;
        this.lat = lat;
        this.lng = lng;
    }

[…]

}
```

Variabili statiche e pubbliche per definire il tipo dei PoI.
Da usare per fare i confronti quando servono.

Variabili della classe che definiscono le caratteristiche di un PoI

Costruttore principale della classe con cui creare un nuovo PoiDescriptor con tutti i parametri necessari

PoiDataManager

```
public class PoiDataManager {

    private static PoiDataManager instance = null;
    private ArrayList<PoiDescriptor> poiList = null;

    private PoiDataManager(){
        this.poiList = new ArrayList<PoiDescriptor>();
    }

    public static PoiDataManager getInstance(){
        if(instance == null)
            instance = new PoiDataManager();
        return instance;
    }

    public void addNewPoi(PoiDescriptor poiDescr){
        this.poiList.add(poiDescr);
    }

    public void removePoi(int position){
        this.poiList.remove(position);
    }

    public void removePoi(PoiDescriptor poiDescr){
        this.poiList.remove(poiDescr);
    }

    public ArrayList<PoiDescriptor> getPoiList(){
        return poiList;
    }
}
```

Definizione del Singleton tramite Costruttore privato e metodo getInstance();

Per avere accesso al riferimento dell'oggetto usare sempre:

PoiDataManager.getInstance().<metodo>();

esempio:
PoiDataManager.getInstance().getPoiList();

Metodi del singleton per la gestione della lista di PoI. Aggiunta, rimozione e recupero Lista

PoiDataManager - Default DataSet

```
private PoiDataManager(){
        this.poiList = new ArrayList<PoiDescriptor>();

        //Add the default DataSet
        this.poiList.add(new PoiDescriptor(PoiDescriptor.POI_TYPE_CINEMA, "Cinecity", "Viale GP Usberti 7A", 44.766013, 10.321014));
        this.poiList.add(new PoiDescriptor(PoiDescriptor.POI_TYPE_SCHOOL, "Unipr - DII", "Parco Area delle Scienze, 181/a - 43124 Parma", 44.764883, 10.308673));
        this.poiList.add(new PoiDescriptor(PoiDescriptor.POI_TYPE_SCHOOL, "WASN Lab", "Parco Area delle Scienze, 181/a - 43124 Parma", 44.765075, 10.308053));
        this.poiList.add(new PoiDescriptor(PoiDescriptor.POI_TYPE_GAS_STATION, "Gas Station", "Tangenziale Sud", 44.769390, 10.319680));
        this.poiList.add(new PoiDescriptor(PoiDescriptor.POI_TYPE_BUS_STOP, "TEP - Università Farmacia", "Viale GP Usberti", 44.765555, 10.318559));
        this.poiList.add(new PoiDescriptor(PoiDescriptor.POI_TYPE_BUS_STOP, "TEP - Università Ingegneria", "Viale GP Usberti", 44.765326, 10.310470));
        this.poiList.add(new PoiDescriptor(PoiDescriptor.POI_TYPE_STORE, "Centro Commerciale Conad Campus", "Viale GP Usberti", 44.766488, 10.319568));
    }
```

Il costruttore privato del Singleton crea e aggiunge in automatico un gruppo di PoiDescriptor per avere a disposizione subito dei dati da visualizzare e con i quali lavorare all'interno della vostra App.

Passi dello sviluppo

- Creazione Nuova applicazione “MyPoiAdvisor”
- Impostare Tema Material [Esempio HelloMaterial]
- Impostare MainActivity (La classe e il layout verranno creati in automatico con il nuovo progetto)
  - Layout XML semplice con un Relative Layout e una TextView
  - Aggiungere la ToolBar ed impostarla come Action Bar [Esempio HelloToolBar]
- **Testare il funzionamento della MainActivity con ToolBar**
- Creare due Fragment Classe Java + Layout XML dedicati ed elementari (singolo RealtiveLayout + TextView) [Esempio HelloFragment]
  - PoiListFragment
  - PoiMapFragment
- Aggiungere un ViewPager alla MainActivity [Esempio HelloViewPager]
  - Aggiungere il tag XML del ViewPager nel layout della MainActivity
  - Definire la classe per il ViewPager Adapter
  - Impostare e configurare il ViewPager nella classe della MainActivity
- **Testare il funzionamento del ViewPager con i Fragment Elementari che sono stati creati**

Passi dello sviluppo

- Aggiungere una RecyclerView al Fragment PoiListFragment per la visualizzazione dei punti di interesse [Esempio HelloList]
  - Tag RecyclerView nel layout del Fragment
  - Definizione Layout dedicato per ogni elemento della lista (Immagine + Testo)
  - Definizione dell'Adapter e integrazione con il Singleton PoiDataManager per il recupero dei dati
  - Impostare l'adapter e gestire la RecyclerView nel codice del Fragment
  - **Testare il funzionamento !**
- Creare l'Activity "AddPoiActivity" (Classe + Layout) per l'aggiunta di un nuovo PoI (Dichiarare la nuova Activity nel Manifest !)
  - All'interno della nuova Activity ci dovrà essere un Button per Salvare il PoI tramite PoiDataManager.getInstance.addNewPoi(…)
  - Al click sul Button, e dopo il corretto salvataggio, si può fare la chiusura dell'Activity chiamando il metodo finish();
- Integrare una Action (file menu dedicato) nella toolbar della MainActivity per lanciare tramite Intent la nuova Activity creata AddPoiActivity
- Integrare una Action per il refresh della lista e la conseguente notifica all'adapter
- **Testare il funzionamento !**

Passi dello sviluppo

- Configurare opportunamente la vostra applicazione per il supporto delle mappe tramite il Manifest [Esempio HelloGoogleMaps]
- Aggiungere una MapFragment (SupportMapFragment) al Fragment PoiMapFragment per la visualizzazione dei punti di interesse su una mappa [Esempio HelloGoogleMaps]
  - Impostare il codice all'interno di PoiMapFragment per il recupero della Mappa
  - Aggiungere i Marker
  - Il tasto Refresh nell'ActionBar della MainActivity dovrà andare ad aggiornare il contenuto dei due Fragment [Esempio MultiFragments]
- **Testare il funzionamento !**

Suggerimento per il Refresh dei dati

```
@Override
public void setMenuVisibility(final boolean visible) {
  super.setMenuVisibility(visible);
  if (visible && !isFirstLoadCompleted) {
    //REFRESH YOUR CONTENT !
  }
}
```

All'interno di un Fragment ed in particolare di un Fragment collocato in un ViewPager si può utilizzare questo metodo per sapere quando un Fragment diventa visibile e se necessario fare un refresh dei contenuti in maniera automatica.

Android Development

Esercitazione Android 1
MyPoiAdvisor