Tutorial. Page view

Objectiu

Crear un slide d'imatges de diferents URL

Pas a pas. Llibreries

1. Afegim l'última versió de la llibreria Glide que ens permetrà mostrar imatges de URL

```
implementation 'com.github.bumptech.glide:glide:4.13.0'
annotationProcessor 'com.github.bumptech.glide:compiler:4.13.0'
```

2. Crearem una nova activitat en la que mostrarem el slider. Allà li afegirem el widget PageView

```
<androidx.viewpager.widget.ViewPager
    android:id="@+id/vpager"
    android:layout_width="409dp"
    android:layout_height="729dp"
    android:layout_marginStart="1dp"
    android:layout_marginTop="1dp"
    android:layout_marginEnd="1dp"
    android:layout_marginBottom="1dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

3. Page view funciona semblant al Recycler view, per tant hem de crear un item_image, que serà el que es replicarà per a cada imatge.
   Crea un xml que es digui item_image i afegeix-li una imatge, aquesta ha de tenir un ID.

4. Per poder mostrar les imatges amb la llibreria Glide necessitem crear abans una classe que es digui GlideModuleApp. La classe ha d'estar buida.

```
@GlideModule
public final class GlideModuleApp extends AppGlideModule {

}
```

   Documentació

5. Fes un rebuild del projecte. Build → Make project

6. Ara ens falta crear l'adaptador del Viewer

```
public class SlidingAdapter extends PagerAdapter {
```

```
    private ArrayList<String> urls;
    private Context context;

    public SlidingAdapter(Context context, ArrayList<String> urls) {
        this.context = context;
        this.urls = urls;
    }
}
```

Crearem una classe que extengui de PagerAdapter i li passarem l'ArrayList de URLs i el context.
Implementarem els @override dels diferents mètodes que volem sobreescriure

destroyItem ens servirà per eliminar l'item actual

```
@Override
public void destroyItem(ViewGroup container, int position, Object object) {
    container.removeView((ConstraintLayout) object);
}
```

isViewFromObject determina si una vista de pàgina està associada a un objecte clau específic tal com retorna instantiateItem(ViewGroup, int).

```
@Override
public boolean isViewFromObject(@NonNull View view, @NonNull Object object) {
    return view == object;
}
```

getCount determina la quantitat d'elements a mostrar, per tant ha de fer un return de la mida de l'ArrayList

```
@Override
public int getCount() {
}
```

instantiateItem crea la vista per la posició donada

```
@Override
public Object instantiateItem(ViewGroup view, int position) {
                                        View              imageLayout              =
LayoutInflater.from(view.getContext()).inflate(R.layout.item_image, view, false);

    final ImageView imageView = imageLayout.findViewById(R.id.imageView);

    GlideApp.with(context)
            .load(urls.get(position))
            .into(imageView);

    Objects.requireNonNull(view).addView(imageLayout);

    return imageLayout;
}
```

GlideApp ens carregarà la imatge de la URL

7. Des de l'activitat has de cridar a l'adaptador com ho fèiem al RecyclerView

```
ViewPager mPager = findViewById(R.id.vpager);
mPager.setAdapter(new SlidingAdapter(MainActivity.this, urls));
```