Department of Computer Technology and Information Systems

CTIS 487 - Mobile Application Development

SPRING 2020 - 2021

Lab Guide Extra Splash Screen

OBJECTIVES : Splash Screen + Recycler View
Instructor : Neşe ŞAHİN ÖZÇELİK **Assistant** : Ruşen ASAN

Create the following design SplashActivty which will be launcher activity duration for 5 seconds.


```
public class SplashActivity extends AppCompatActivity {
    Intent intent;
    // Splash screen timer
    private static int SPLASH_TIME_OUT = 5000;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Hiding title bar using code
        getSupportActionBar().hide();

        setContentView(R.layout.activity_splash);

        // Hiding the status bar
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);

        // Locking the orientation to Portrait
        setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);

        /*
         * postDelayed() method causes the Runnable object to be added to the message queue to
         * be run after the specified amount of time elapses. The runnable will be run on the
         * thread to which this handler is attached.
         */
        new Handler().postDelayed(new Runnable() {

            /*
             * Showing splash screen with a timer. This will be useful when you
             * want to show your application logo or company logo
             */
            @Override
            public void run() {
                // This method will be executed once the timer is over
                // Start your application's main activity which is a ListView
                intent = new Intent(SplashActivity.this, MainActivity.class);

                // Close this activity
                finish();

                startActivity(intent);
            }
        }, SPLASH_TIME_OUT);
    }
}
```

Step1: Create a House class

House
-name: String -words: String -logo: int
+House(String,String,int) +getName():String +setName(String) +getWords():String +setWords(String) +getLogo():String +setLogo(String) +toString():String

Step2: Create empty MainActivity and its layout

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerGot"
    android:layout_width="0dp"
    android:layout_height="286dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="112dp"
    android:layout_marginEnd="16dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

Step3: Create recycler_layout

```
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/constLayout"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">

       <androidx.cardview.widget.CardView
        android:layout_width="337dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="149dp">

            <ImageView
                android:id="@+id/rec_img"
                android:layout_width="70dp"
                android:layout_height="60dp"
                android:layout_marginStart="16dp"
                android:layout_marginTop="4dp"
                android:layout_marginBottom="8dp"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent"
                app:srcCompat="@mipmap/ic_launcher" />

            <TextView
                android:id="@+id/rec_tv"
                android:layout_width="125dp"
                android:layout_height="44dp"
                android:layout_marginStart="8dp"
                android:layout_marginTop="16dp"
                android:text="TextView"
                android:textStyle="bold|italic"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintHorizontal_bias="0.008"
                app:layout_constraintStart_toEndOf="@+id/rec_img"
                app:layout_constraintTop_toTopOf="parent" />
```

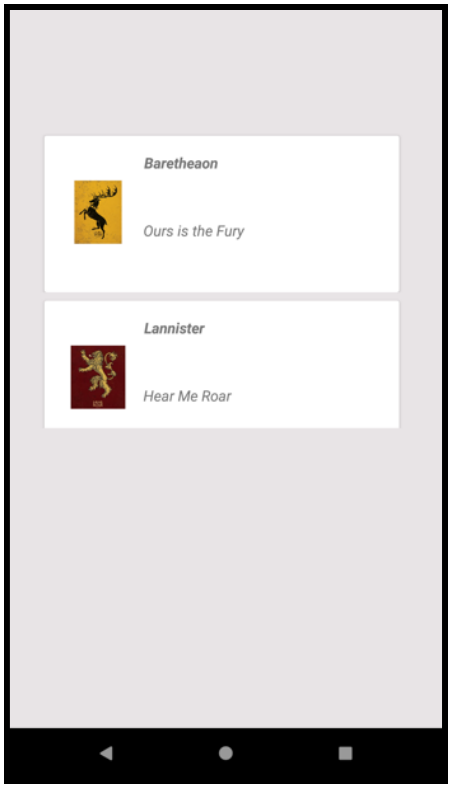
```
            <TextView
                android:id="@+id/rec_exp"
                android:layout_width="125dp"
                android:layout_height="44dp"
                android:layout_marginStart="8dp"
                android:layout_marginTop="20dp"
                android:layout_marginBottom="25dp"
                android:text="TextView"
                android:textStyle="italic"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintStart_toEndOf="@+id/rec_img"
                app:layout_constraintTop_toBottomOf="@+id/rec_tv"
                app:layout_constraintVertical_bias="0.0" />

        </androidx.constraintlayout.widget.ConstraintLayout>

    </androidx.cardview.widget.CardView>
</androidx.constraintlayout.widget.ConstraintLayout>
```

STEP 4: To create custom recycler view

- As customized recycler item house image and next to it house name and the house words will be displayed, get the recycler_layout.xml and import it to your project.



To create custom RecylerView

- In MainActivity class, create an ArrayList to hold House objects and add House objects by calling prepareData() method. Think about this cities can be taken from a database or json file through network connection.

```
private ArrayList<House> recyclervalues = new ArrayList<>();

@Override
protected void onCreate(Bundle savedInstanceState) {
      ...
      prepareData();
      ...
}

public void prepareData() {
    Collections.addAll(recyclervalues, new House(R.drawable.baratheon, "Baretheaon", "Ours is the
Fury"),
            new House(R.drawable.lannister, "Lannister", "Hear Me Roar"),
            new House(R.drawable.martell, "Martell", "Unbowed,Unbent,Unbroken"),
            new House(R.drawable.stark, "Stark", "Winter is Coming"),
            new House(R.drawable.targeryan, "Targeryan", "Fire and Blood"),
            new House(R.drawable.tyrell, "Tyrell", "Growing Strong"));
    /**********or ************/
    /*
    recyclervalues.add(new House(R.drawable.lannister, "Lannister", "Hear Me Roar"));
    ...
    //This data can be taken from file, database or network.*/
}
```

- Create recycler object in MainActivity

```
private RecyclerView recyclerHouses;
```

- Create MyRecyclerViewAdapter object and set the adapter to recycler.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    ...
    recyclerHouses = findViewById(R.id.recyclerGot);
    layoutManager = new LinearLayoutManager(this);

    //LayoutManager.setOrientation(LinearLayoutManager.HORIZONTAL);
    layoutManager.setOrientation(LinearLayoutManager.VERTICAL);
    recyclerHouses.setLayoutManager(layoutManager);

    recyclerHouses.hasFixedSize();
    adapter = new MyRecyclerViewAdapter(this, recyclervalues);
    recyclerHouses.setAdapter(adapter);
}
```

- Create MyRecyclerViewAdapter class
    - **MyRecyclerViewItemHolder:** That method needs to construct a RecyclerView.ViewHolder and set the view it uses to display its contents. The type of the ViewHolder must match the type declared in the Adapter class signature. Typically, it would set the view by inflating an XML layout file. Because the view holder is not yet assigned to any particular data, the method does not actually set the view's contents.
    - The layout manager then binds the view holder to its data. It does this by calling the adapter's **onBindViewHolder()** method, and passing the view holder's position in the RecyclerView. The onBindViewHolder() method needs to fetch the appropriate data, and use it to fill in the view holder's layout.

```
public class MyRecyclerViewAdapter extends
RecyclerView.Adapter<MyRecyclerViewAdapter.MyRecyclerViewItemHolder> {
    private Context context;
    private ArrayList<House> recyclerItemValues;

    //Custom Dialog
    int selected;
    Dialog customDialog;
    Button btnDialogDone;
    TextView nameOfHouse;
    TextView houseWords;
    ImageView houseLogo;
    House currentSelectedHouse;

    public MyRecyclerViewAdapter(Context context, ArrayList<House> values) {
        this.context = context;
        this.recyclerItemValues = values;
    }

    @NonNull
    @Override
    public MyRecyclerViewItemHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int i) {
        LayoutInflater inflator = LayoutInflater.from(viewGroup.getContext());

        View itemView = inflator.inflate(R.layout.recycler_layout, viewGroup, false);

        MyRecyclerViewItemHolder mViewHolder = new MyRecyclerViewItemHolder(itemView);
        return mViewHolder;
    }

    @Override
    public void onBindViewHolder(@NonNull MyRecyclerViewItemHolder myRecyclerViewItemHolder, int i) {

        final House sm = recyclerItemValues.get(i);

        myRecyclerViewItemHolder.name.setText(sm.getName());
        myRecyclerViewItemHolder.words.setText(sm.getWords());
        myRecyclerViewItemHolder.img.setImageResource(sm.getLogo());


        myRecyclerViewItemHolder.parentLayout.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Toast.makeText(context, "Clicked on item " + sm.getName(), Toast.LENGTH_LONG).show();
                makeAndShowDialogBox(sm.getName(),sm.getLogo(),sm.getWords());
            }
        });

        myRecyclerViewItemHolder.img.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
    Toast.makeText(context, "Clicked on " + sm.getName() + " Image of item", Toast.LENGTH_LONG).show();
                makeAndShowDialogBox(sm.getName(),sm.getLogo(),sm.getWords());
            }
        });

        myRecyclerViewItemHolder.name.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
 Toast.makeText(context, "Clicked on " + sm.getName() + " TextView of item", Toast.LENGTH_LONG).show();
                makeAndShowDialogBox(sm.getName(),sm.getLogo(),sm.getWords());
            }
        });
    }

    @Override
    public int getItemCount() {
        return recyclerItemValues.size();
    }

    class MyRecyclerViewItemHolder extends RecyclerView.ViewHolder {
        TextView name, words;
        ImageView img;
        ConstraintLayout parentLayout;

        public MyRecyclerViewItemHolder(@NonNull View itemView) {
            super(itemView);
            name = itemView.findViewById(R.id.rec_tv);
            words = itemView.findViewById(R.id.rec_exp);
            img = itemView.findViewById(R.id.rec_img);
            parentLayout = itemView.findViewById(R.id.constLayout);
        }
    }
```

```
    private void makeAndShowDialogBox(String hname, int logo, String hWords) {
        AlertDialog.Builder mDialogBox = new AlertDialog.Builder(context);
        // set message, title, and icon
        mDialogBox.setTitle(hname);
        mDialogBox.setMessage(hWords);
        mDialogBox.setIcon(logo);

        // Set three option buttons
        mDialogBox.setPositiveButton("Ok",
                new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int whichButton) {
                        // whatever should be done when answering "YES" goes
                        // here
                    }
                });
        mDialogBox.create();
        mDialogBox.show();
    }

}
```

- Run the application, observe it. Opening the list scroll down-up.
- When click to the members of the Recycler List, there will be a toast message as shown.