Report Laboratory Work 1

Elena Morelli

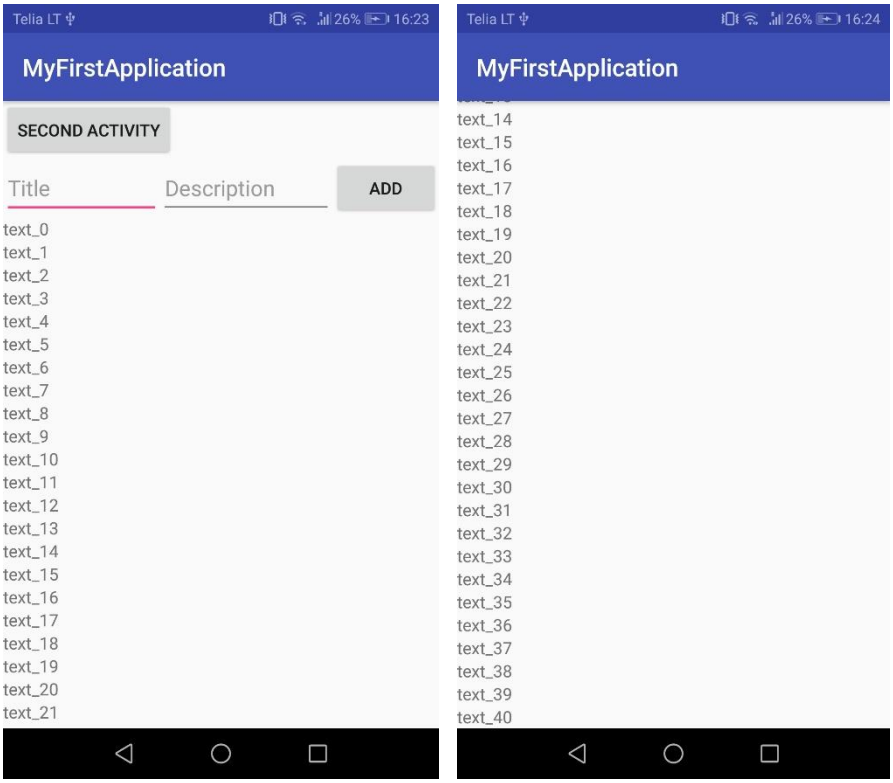
1. Remove a button from the first Activity

```xml
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
   <!-- <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button"
        android:text="@string/button"/> -->

    <Button
        android:id="@+id/secondActivityButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Second Activity" />
```

Deleted the button in the layout and deleted it in the activity class

```java
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.firstactivitydesign);
    title = (EditText) findViewById(R.id.title1);
    description = (EditText) findViewById(R.id.description);
    Button addButton = (Button) findViewById(R.id.addButton);
    //myButton = (Button) findViewById(R.id.button);
   // myTextField = (TextView) findViewById(R.id.textfield);
    Button secondActivityButton = (Button) findViewById(R.id.secondActivityButton);
   // myButton.setOnClickListener(myButtonClick);
    secondActivityButton.setOnClickListener(startSecondActivity);
    secondActivityButton.setOnLongClickListener(startSecondActivityLong);
    addButton.setOnClickListener(addButtonListener);

}

/* View.OnClickListener myButtonClick = new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        myTextField.setText(myTextField.getText()+"\n"+"Next line");
    }
};*/
```

2. Add the option to scroll the content when the size of the contents in the Activity exceed the size of the window (using the Scrollview component).




```
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <LinearLayout
        android:id="@+id/linear"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">
    <!-- <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button"
        android:text="@string/button"/> -->

    <Button
        android:id="@+id/secondActivityButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Second Activity" />

    <TableLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <TableRow
```

Added a ScrollView element in the layout and a linear layout within. The linear layout was added because the scroll view can just hold only one child.

In the phone picture there were added more strings to show that everything is working.

3. Make a list of items to be created and transferred from the first Activity to be displayed on the second

```
public void runSecondActivity(boolean b){
    Intent intent = new Intent(context,SecondActivity.class);
    intent.putExtra( name: "flag",b);
    createList();
    intent.putExtra( name: "arrayList",(Serializable)list);
    context.startActivity(intent);
}

View.OnClickListener startSecondActivity = (v) → { runSecondActivity( b: true); };

View.OnLongClickListener startSecondActivityLong = (v) → {
        runSecondActivity( b: false);
        return true;
};

public void createList(){
    list.add(new ListItem( title: "Adrien",R.drawable.ic_3d_rotation_black_48dp, description: "Mathematics, Chemistry"));
    list.add(new ListItem( title: "Harry",R.drawable.ic_announcement_black_48dp, description: "Physics, Informatics"));
    list.add(new ListItem( title: "Louis",R.drawable.ic_alarm_black_48dp, description: "Geography, Chemistry"));
    list.add(new ListItem( title: "Mark",R.drawable.ic_account_box_black_48dp, description: "Mathematics, Chemistry"));
    list.add(new ListItem( title: "Luke",R.drawable.ic_accessibility_black_48dp, description: "Mathematics, Physics"));
}
```

FirstActivity.java

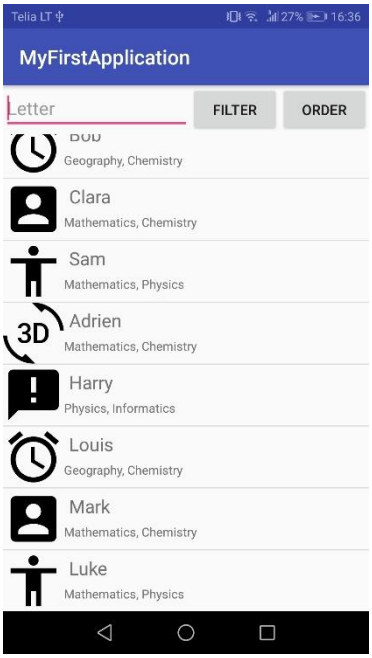
The createList method is used to fill the list that is about to be send to the second Activity.

To be able to send the list to the second Activity the ListItem class must implement serializable

```
ArrayList<ListItem> object = (ArrayList<ListItem>) getIntent().getSerializableExtra( name: "arrayList");

if(object != null){
    items.addAll(object);
}

if (getIntent().getBooleanExtra( name: "flag", defaultValue: true)){

    items.add(new ListItem( title: "Jack",R.drawable.ic_3d_rotation_black_48dp, description: "Mathematics, Chem
    items.add(new ListItem( title: "Jane",R.drawable.ic_announcement_black_48dp, description: "Physics, Informa
    items.add(new ListItem( title: "Bob",R.drawable.ic_alarm_black_48dp, description: "Geography, Chemistry"));
```

After getting the list from the intent, it's been added to the list that the adapter use.

SecondActivity.java



4. In the first Activity, place two EditText elements(title and description) in one row and place the “Add” button. The add button should create and object from titles and description (all its elements may be assigned the same Figure)

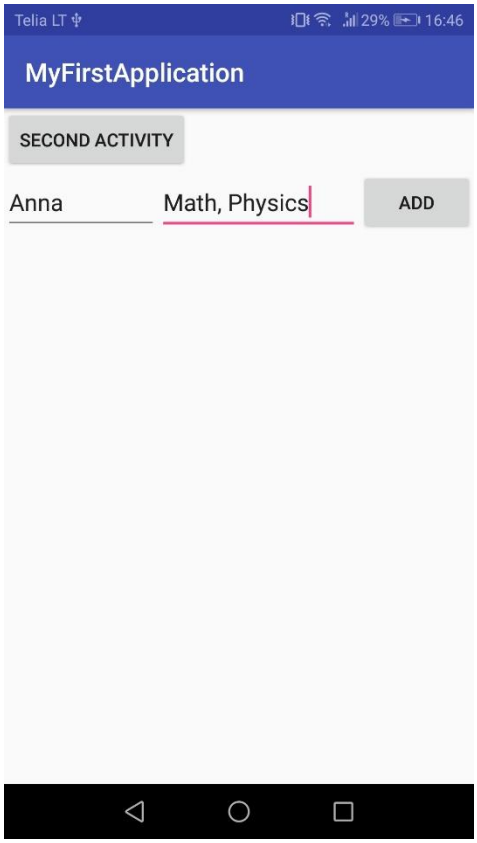
```
<TableLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TableRow
        android:layout_width="match_parent"
        android:id="@+id/row"
        android:weightSum="3">

        <EditText
            android:id="@+id/title1"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:hint="Title"
            android:inputType="text"
            android:layout_weight="2"/>

        <EditText
            android:id="@+id/description"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:hint="Description"
            android:inputType="text"
            android:layout_weight="1"/>

        <Button android:id="@+id/addButton"
            android:text="Add" />

    </TableRow>
</TableLayout>
```

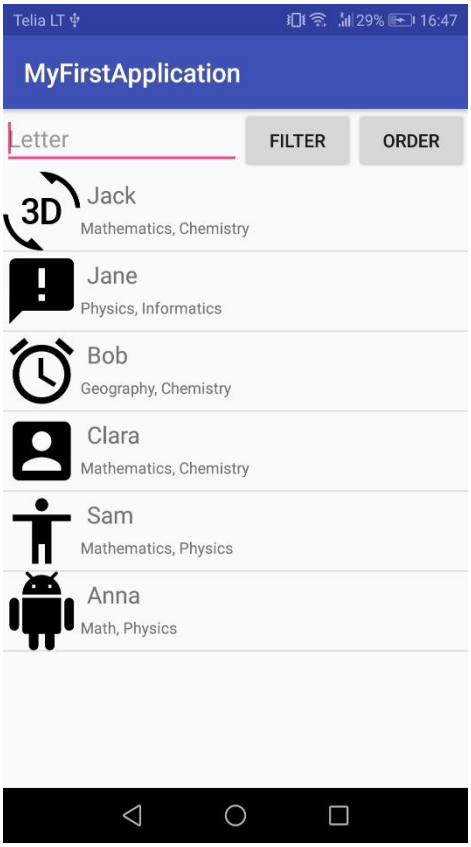
```
View.OnClickListener addButtonListener = (v) -> {
        if(!title.getText().toString().equals("") && !description.getText().toString().equals("")) {
            ListItem item = new ListItem(title.getText().toString(), description.getText().toString());
            list.add(item);
        }else{
            new AlertDialog.Builder( context: FirstActivity.this)
                    .setTitle("Info")
                    .setMessage("Fill both fields before advancing")
                    .setPositiveButton( text: "OK",  listener: null)
                    .show();
        }

};
```

FirstActivity.java

The item will be created only if both fields are not empty.

When the SecondActivity button will be pressed the item will be added and shown in the list in secondActivity



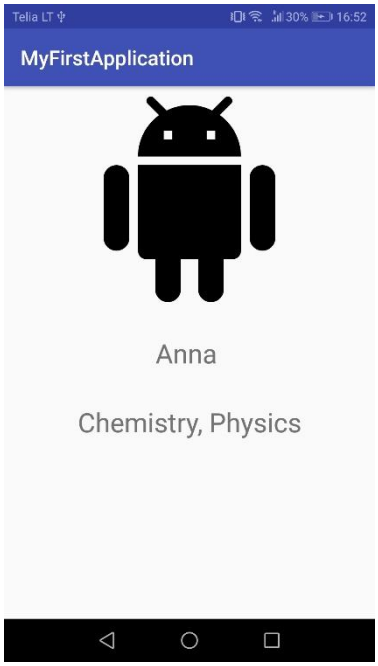


5. Create another Activity, which should contain:

- Image
- The name of the image
- Description

```
public class ThirdActivity extends AppCompatActivity{

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.thirdactivitydesign);
        ListItem item = (ListItem)getIntent().getSerializableExtra( name: "item");
        if(item != null) {
            TextView title = findViewById(R.id.titleView);
             title.setText(item.getTitle());
            TextView description = findViewById(R.id.descriptionView);
             description.setText(item.getDescription());
            ImageView image = findViewById(R.id.image);
             image.setImageResource(item.getImageId());
        }
    }
```

```
<ImageView
    android:id="@+id/image"
    android:layout_width="match_parent"
    android:layout_marginTop="10sp"
    android:layout_height="200dp"
    app:layout_constraintStart_toStartOf="parent" />

<TextView
    android:id="@+id/titleView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="32dp"
    android:text="@string/title"
    app:layout_constraintTop_toBottomOf="@+id/image"
    tools:layout_editor_absoluteX="0dp"
    android:textSize="26sp"
    android:gravity="center"/>

<TextView
    android:id="@+id/descriptionView"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginTop="32dp"
    android:text="@string/description"
    app:layout_constraintTop_toBottomOf="@+id/titleView"
    tools:layout_editor_absoluteX="0dp"
    android:textSize="26sp"
    android:gravity="center"/>
```




```
adapter = new ListAdapter( context: this,items);
myList.setAdapter(adapter);
myList.setOnItemClickListener((parent, view, position, id) → {
        Object object = myList.getItemAtPosition(position);
        ListItem item = (ListItem) object;
        intent.putExtra( name: "item",item);
        startActivity(intent);
});
```

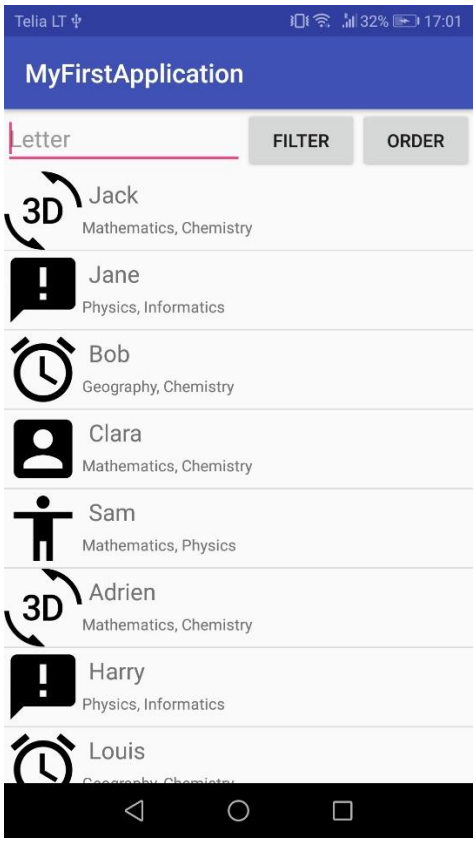
Created a simple activity that shows the Item details of the item selected in the listview.

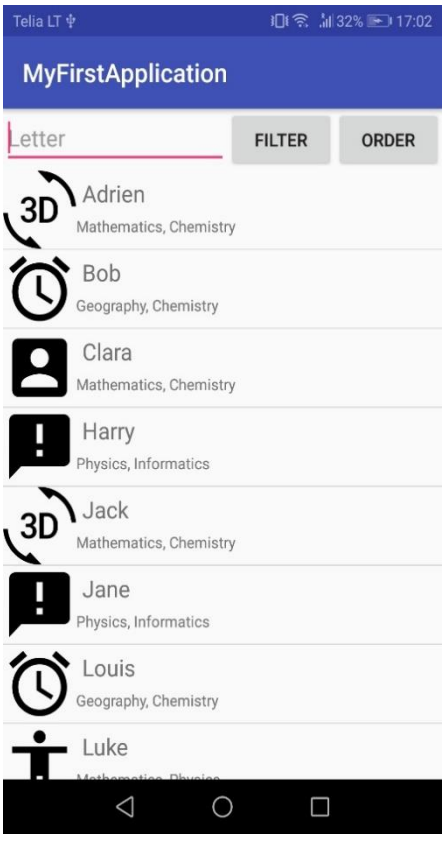
6. Add an option to sort the items in an alphabetical order

```
View.OnClickListener orderListener = new View.OnClickListener(){
    @Override
    public void onClick(View v) {
        orderingMethod();
       adapter.notifyDataSetChanged();
    }
};
```

```
private void orderingMethod(){
    adapter.sort( new Comparator<ListItem>() {
        @Override
        public int compare(ListItem o1, ListItem o2) {
            String s1 = o1.getTitle();
            String s2 = o2.getTitle();
            return s1.compareToIgnoreCase(s2);
        }
    });
}
```

Used adaper.sort method to order the list.





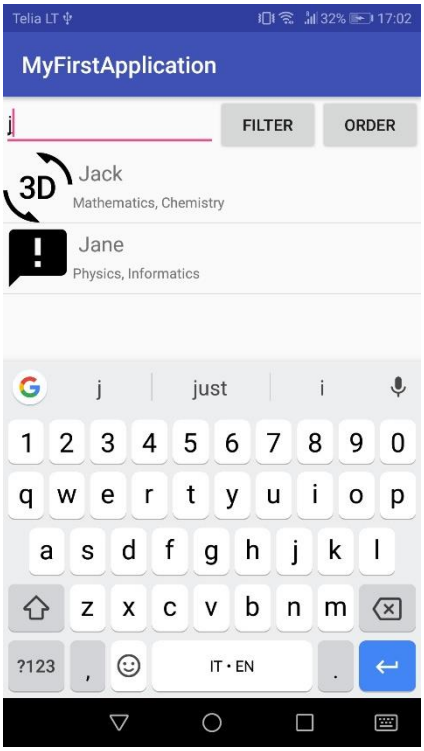
7. Make the filter by letter. In the text box, whrìen you enter a letter and it must filter list of the elements to display only those that begins with that letter

```
View.OnClickListener filterListener = new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String letter = filter.getText().toString().toUpperCase();
        if (!letter.equals("") && letter.length()== 1 ) {
            List<ListItem> list = new ArrayList<>();
            for (ListItem item : items) {
                if (item.getTitle().startsWith(letter)) {
                    list.add(item);
                }
            }
            if(list.isEmpty()){
                createDialog("No match found");
            } else {
                setNewAdapter(list);
            }
        } else if (letter.equals("")) {
            setNewAdapter(items);
        }else{
            createDialog("Insert just one letter !");
        }
```

Added an EditText and a button to the second Activity. When the add button will be pressed the insert letter will be compared with every item's title first letter, if they will match, the item will be added to a new list.

```
private void setNewAdapter(List<ListItem> list){
    adapter = new ListAdapter(context, list);
    myList.setAdapter(adapter);
    adapter.notifyDataSetChanged();
```

The list that contains the matching item is set in the adapter.

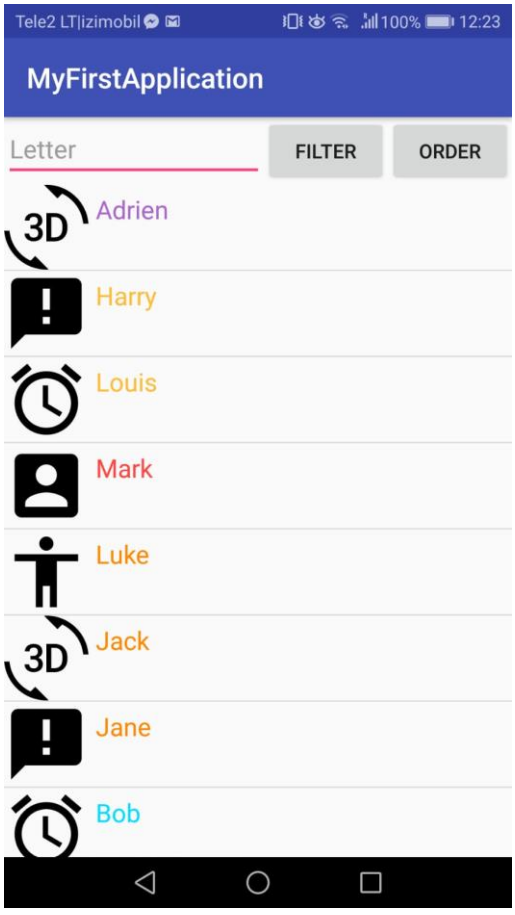


8. Special Task

```
public class ListAdapter extends ArrayAdapter<ListItem> {
    private int [] colorArray ={android.R.color.holo_red_light,
            android.R.color.holo_orange_dark,
            android.R.color.holo_orange_light,
            android.R.color.holo_green_light,
            android.R.color.holo_blue_bright,
            android.R.color.holo_purple,
    };
    private Random random = new Random();

    public ListAdapter(Context context, List<ListItem> objects) {
        super(context, R.layout.listitemdesign, objects);

    }
```

```
int color = random.nextInt(colorArray.length);
while (color < 0){
    color = random.nextInt( bound: 5);
}
TextView title =(TextView) v.findViewById(R.id.title);
title.setTextColor(getContext().getResources().getColor(colorArray[color]));

TextView description =(TextView) v.findViewById(R.id.description);
description.setTextColor(colorArray[color]);
```

ListAdapter.java



To make the second activity colourful I used an array of int. Every int inside represent a colour.

To choose the colour I used a random function.