

# **HW6:**

## **ContactManager2**

Martin Mudenda Bbela

2582912

# Overview:

Contact management app version 2

- Like version 1, the layout for the main activity should have a button (at the bottom of the layout) for users to add a new contact, and the new contact information should be passed back to the main activity on exit of that activity
- The created contacts should be displayed as a ListView in the main activity
- In the ListView, each contact's full name should be displayed
- In the main activity, a user could click any row of the contact, a third activity would open to display the contact details

## Method:

### ContactConstants

This is a simple class that will hold constants that will be used to reference values that are passed through intents .

```
public class ContactConstants {  
    public static final String CONTACT_NAME = "contact_name";  
    public static final String CONTACT_EMAIL = "contact_email";  
    public static final String CONTACT_PHONE = "contact_phone";  
}
```

### Contact

A custom object known as contact has been created to house all the information about a specific contact. This will make it easier to handle transferring, creating and showing contacts.

```
public class Contact {  
    private String name, phone, email;  
  
    public String getName() { return name; }  
    public void setName(String name) { this.name = name; }  
    public String getPhone() { return phone; }  
    public void setPhone(String phone) { this.phone = phone; }  
    public String getEmail() { return email; }  
    public void setEmail(String email) { this.email = email; }  
}
```

## CreateContactActivity

Create Contact has a Linear Layout with each Edit Texts to add a name email and Phone. The set Contact button will will return these values to the Main activity activity

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:orientation="horizontal">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Name" />

    <EditText
        android:id="@+id/editTextTextPersonName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="3"
        android:ems="10"
        android:hint="Name"
        android:inputType="textPersonName" />

</LinearLayout>
```

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_create_contact);
    btnFinish = findViewById(R.id.buttonFinish);
    etPersonName = findViewById(R.id.editTextTextPersonName);
    etEmailAddress = findViewById(R.id.editTextTextEmailAddress);
    etPhone = findViewById(R.id.editTextTextPhone);

    btnFinish.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent returnContact = new Intent();
            returnContact.putExtra(ContactConstants.CONTACT_NAME, etPersonName.getText().toString());
            returnContact.putExtra(ContactConstants.CONTACT_EMAIL, etEmailAddress.getText().toString());
            returnContact.putExtra(ContactConstants.CONTACT_PHONE, etPhone.getText().toString());
            setResult(RESULT_OK, returnContact);
            finish();
        }
    });
}
```

## ShowContactActivity

Similar to the previous Contact Manager App, it will contain three TextViews that will be wired to show the values of the Contact Information sent to it through an intent.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res-auto"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ShowContactActivity">

    <TextView
        android:id="@+id/textViewName"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView"
        app:layout_constraintBottom_toTopOf="@+id/textViewEmail"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.501"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.733" />

    <TextView
        android:id="@+id/textViewEmail"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.501"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.733" />

    <TextView
        android:id="@+id/textViewPhone"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="TextView"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.501"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.733" />

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

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_show_contact);

    Intent contactInfo = savedInstanceState;
    btnReturn = findViewById(R.id.buttonReturn);
    tvName = findViewById(R.id.textViewName);
    tvPhone = findViewById(R.id.textViewPhone);
    tvEmail = findViewById(R.id.textViewEmail);

    tvName.setText(contactInfo.getStringExtra(ContactConstants.CONTACT_NAME));
    tvPhone.setText(contactInfo.getStringExtra(ContactConstants.CONTACT_PHONE));
    tvEmail.setText(contactInfo.getStringExtra(ContactConstants.CONTACT_EMAIL));

    btnReturn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) { finish(); }
    });
}
```

## Main Activity

The main activity layout is a List view Within a Linear Layout and an add contact button Below it.

In the Main activity Class we wire a n ArrayAdapter with the *simple\_list\_item\_1* and a list of the contactNames. This will allow us to view the contact names on the Main activity. The new adapter is attached to the list view and An onItemClickListener is attached to the listview as well. The onItemClickListener will start the *ShowContactsActivity for the specific contact*.

An arrayList is used to store the contactNames and a hash Map has the contact name as a key for the Contact object. The values of the contact object are extracted and added to the intent to initialize the show contact intent.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".MainActivity" >

    <ListView
        android:id="@+id/listViewContacts"
        android:layout_width="match_parent"
        android:layout_height="addsp"
        android:layout_above="@+id/buttonAddContact"
        android:layout_marginBottom="5dp" />

    <Button
        android:id="@+id/buttonAddContact"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_marginEnd="15dp"
        android:layout_marginBottom="15dp"
        android:layout_weight="1"
        android:gravity="center"
        android:text="Add Contact" />

</RelativeLayout>
```

```
lvContacts = findViewById(R.id.listViewContacts);
lvContacts.setAdapter(contactListAdapter);
lvContacts.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView? adapterView, View view, int i, long l) {
        Contact c = contacts.get(contactNames.get(i));
        //TODO get View Contact

        Intent showContactIntent = new Intent(getApplicationContext(), ShowContactActivity.class);
        showContactIntent.putExtra(ContactConstants.CONTACT_NAME, c.getName());
        showContactIntent.putExtra(ContactConstants.CONTACT_EMAIL, c.getEmail());
        showContactIntent.putExtra(ContactConstants.CONTACT_PHONE, c.getPhone());

        startActivity(showContactIntent);
    }
});
```

Using the new Android Pattern for activity results. I created an activityResultsLauncher with an activity result Contract. This will get the information after a contact has been created in the create contact activity and add it to the contacts HashMap, ContactNames list and will finally tell the adapter to update the list. We will then a listener to the btnAddContact to launch the CreateContactActivity

```

mGetContactInformation = registerForActivityResult(new ActivityResultContracts.StartActivityForResult(), new ActivityResult
@Override
public void onActivityResult(ActivityResult result) {
    Contact c = new Contact();
    c.setName(result.getData().getStringExtra(ContactConstants.CONTACT_NAME));
    c.setPhone(result.getData().getStringExtra(ContactConstants.CONTACT_PHONE));
    c.setEmail(result.getData().getStringExtra(ContactConstants.CONTACT_EMAIL));

    contacts.put(c.getName(), c);
    contactNames.add(c.getName());
    contactsAdapter.notifyDataSetChanged();
}
});

btnAddContact.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View view) {
    Intent startCreateContentIntent = new Intent(getApplicationContext(), CreateContactActivity.class);
    mGetContactInformation.launch(startCreateContentIntent);
}
});

```

## RESULTS

User is able to create contacts, populating the list view in the main activity and can further view these contacts in another activity on clicking on them.

