Assignment 04

COS30017 - Software Development for Mobile Devices

Daniel Parker 971328X

September 25, 2014

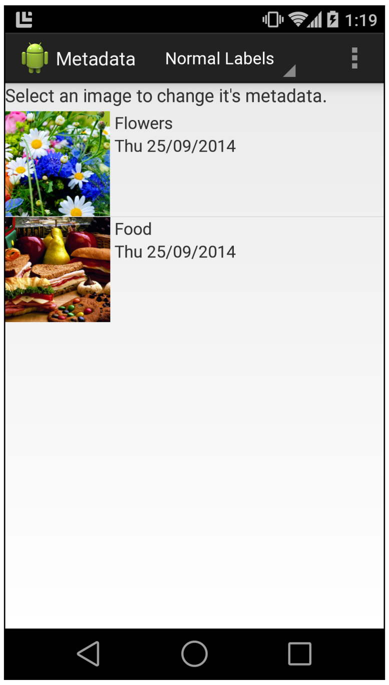
1. MetaData App

The Metadata app contains two activities. The main activity contains a listview of all the ImageData objects and the list item layout is specified in a separate xml layout file. The wiring of these two activities uses the parcelable protocol. The ImageData class implements the Parcelable interface as it is the data that will be sent between the two activities. The EditMetadata activity is called using StartActivityForResult because the edited ImageData object needs to be returned and the updated metadata shown in the listview. The label text sizes are set using styles. The user can change the size of the text by selecting a different option from the dropdown list in the EditMetadata activity, which changes the theme of the activity and subsequently the text size for the labels.

1.1. What is the Parcelable protocol

The Parcelable protocol is a method for serializing object data so that it can be sent between activities as context. In the case of this Metadata app, when a user selects an image to edit, the underlying ImageData object is serialized as per the Parcelable interface implementation and sent to the EditMetadata activity. That activity will deserialize the object from using the parcelable interface so that it can be accessed like a normal object and edited. The same is done when the object is returned as a result from the EditMetadata activity. The reason we do not use the Serializable interface is that the Parcelable interface is significantly quicker to serialize and deserialize. This is desirable for Android apps considering that they run on lower powered devices and time is critical with these simple user interface wiring tasks.

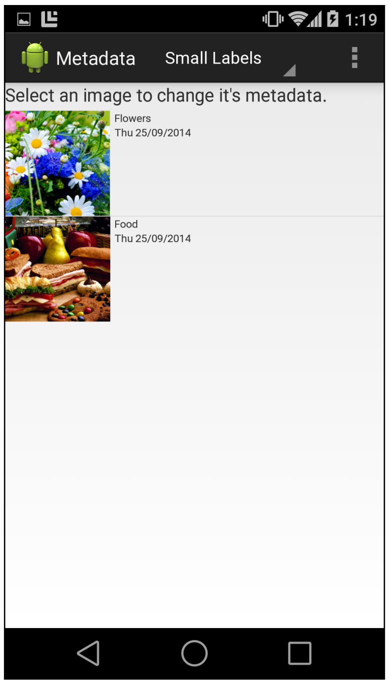
1.2. screenshots



Select an image to change it's metadata.

Flowers
Thu 25/09/2014

Food
Thu 25/09/2014



Select an image to change it's metadata.

Flowers
Thu 25/09/2014

Food
Thu 25/09/2014

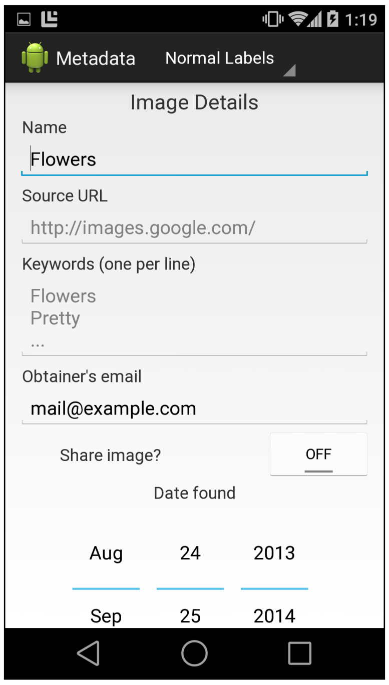


Image Details

Name

Flowers

Source URL

http://images.google.com/

Keywords (one per line)

Flowers
Pretty
...

Obtainer's email

mail@example.com

Share image? OFF

Date found

Aug 24 2013

Sep 25 2014

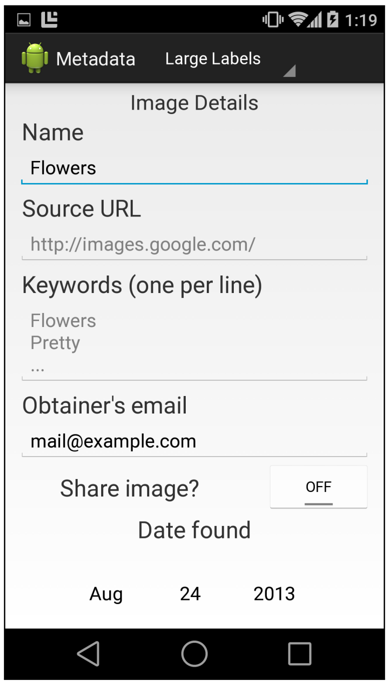


Image Details

Name

Flowers

Source URL

http://images.google.com/

Keywords (one per line)

Flowers
Pretty
...

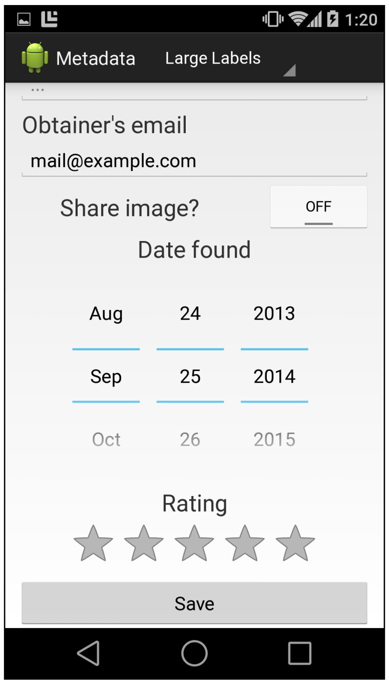
Obtainer's email

mail@example.com

Share image? OFF

Date found

Aug 24 2013


1:20
Metadata
Large Labels
...
Obtainer's email
mail@example.com
Share image?
OFF
Date found
Aug
24
2013
Sep
25
2014
Oct
26
2015
Rating
Save

1.3. Source

1.3.1. Gallery.java (Startup activity)

```java
package au.net.danielparker.metadata;

import android.app.ActionBar;
import android.app.ListActivity;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.SpinnerAdapter;

import org.androidannotations.annotations.EActivity;

import java.util.ArrayList;

@EActivity(R.layout.activity_gallery)
public class Gallery extends ListActivity {

    private boolean navFired = false;

    private ArrayList<ImageData> galleryData = new ArrayList<ImageData>();
    private ArrayAdapter<ImageData> adapter;
```

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

    Bundle bundledData = getIntent().getExtras();

    if (bundledData != null && bundledData.containsKey("Theme")) {
        setTheme(bundledData.getInt("Theme"));
    }

    // Create actionbar dropdown spinner from stringarray
    SpinnerAdapter mSpinnerAdapter = ArrayAdapter.createFromResource(
                                     getActionBar().getThemedContext(),
                                     R.array.text_sizes,
                                     android.R.layout.simple_spinner_dropdown_item);

    ActionBar.OnNavigationListener mOnNavigationListener = new ActionBar.OnNavigationListener() {
        @Override
        public boolean onNavigationItemSelected(int position, long itemId) {

            Log.d("METADATA", "Position: " + position + " ItemId: " + itemId);
            Intent intent = getIntent();

            if (navFired == true ) {
                switch (position) {
                    case 0:
                        intent.putExtra("Theme", R.style.NormalText);
                        finish();
                        startActivity(intent);
                        break;
                case 1:
                    intent.putExtra("Theme", R.style.SmallText);
                    finish();
                    startActivity(intent);
                    break;
                case 2:
                    intent.putExtra("Theme", R.style.LargeText);
                    finish();
                    startActivity(intent);
                    break;
            }
        } else {
            navFired = true;
        }

        return true;
    }
};

getActionBar().setNavigationMode(ActionBar.NAVIGATION_MODE_LIST);
getActionBar().setListNavigationCallbacks(mSpinnerAdapter, mOnNavigationListener);

if (bundledData != null && bundledData.containsKey("Theme")) {
    switch (bundledData.getInt("Theme")){
        case R.style.NormalText:
            getActionBar().setSelectedNavigationItem(0);
            break;
        case R.style.SmallText:
            getActionBar().setSelectedNavigationItem(1);
            break;
            case R.style.LargeText:
                getActionBar().setSelectedNavigationItem(2);
                break;
        }

    } else {
        getActionBar().setSelectedNavigationItem(0);
    }

    setContentView(R.layout.activity_gallery);

    initialiseUI();
}

@Override
public void onListItemClick(ListView l, View v, int position, long id) {
    ImageData selectedItem = (ImageData) getListView().getItemAtPosition(position);
    editMetadataForItem(selectedItem, position);
}

private void editMetadataForItem(ImageData selectedItem, int position) {
    Intent intent = new Intent(this, EditMetadata_.class );
    intent.setAction(Intent.ACTION_EDIT);
    intent.putExtra("ImageData", selectedItem);
    intent.putExtra("position", position);

    startActivityForResult(intent, 1);
}

@Override
```

```java
protected void onActivityResult(int requestCode, int responseCode, Intent data) {
    if (requestCode == 1) {
        if (responseCode == RESULT_OK) {
            ImageData editedSelection = data.getParcelableExtra("ImageData");
            int position = data.getIntExtra("position", -1);
            if (position != -1) {
                galleryData.set(position, editedSelection);
                adapter.notifyDataSetChanged();
            } else {
                Log.e("METADATA", "Error getting edited metadata: Invalid index");
            }
        }
    }
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action bar if it is present.
    getMenuInflater().inflate(R.menu.contact_list, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar will
    // automatically handle clicks on the Home/Up button, so long
    // as you specify a parent activity in AndroidManifest.xml.
    int id = item.getItemId();
    if (id == R.id.action_settings) {
        return true;
        }
        return super.onOptionsItemSelected(item);
    }

    private void initialiseUI() {
        this.galleryData = loadGalleryItems();
        adapter = new GalleryAdapter(this, galleryData);

        setListAdapter(adapter);
    }

    private ArrayList<ImageData> loadGalleryItems() {
        ArrayList<ImageData> images = new ArrayList<ImageData>();
        try {
            images.add(new ImageData("Flowers", "mail@example.com", R.drawable.flowers));
            images.add(new ImageData("Food", "mail@example.com", R.drawable.food));
        } catch (Exception e ) {

        }

        return images;
    }
}
```

1.3.2. EditMetadata.java

```
package au.net.danielparker.metadata;

import android.app.ActionBar;
import android.app.Activity;
import android.app.Service;
import android.content.Intent;
import android.media.Rating;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.util.Patterns;
import android.view.inputmethod.InputMethodManager;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.MultiAutoCompleteTextView;
import android.widget.RatingBar;
import android.widget.SpinnerAdapter;
import android.widget.Toast;
import android.widget.ToggleButton;

import org.androidannotations.annotations.AfterViews;
import org.androidannotations.annotations.Click;
import org.androidannotations.annotations.EActivity;
import org.androidannotations.annotations.ViewById;

import java.io.BufferedReader;
import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.Calendar;
```

```
/**
 * Created by danielparker on 14/09/14.
 */
@EActivity(R.layout.activity_edit_metadata)
public class EditMetadata extends Activity {

    private boolean navFired = false;

    private ImageData selectedImage;
    private int position;

    @ViewById(R.id.name)
    EditText name;

    @ViewById(R.id.source_url)
    EditText sourceURL;

    @ViewById(R.id.keywords)
    MultiAutoCompleteTextView keywords;

    @ViewById(R.id.source_email)
    EditText sourceEmail;

    @ViewById(R.id.share_toggle)
    ToggleButton shareToggle;

    @ViewById(R.id.date)
    DatePicker date;

    @ViewById(R.id.rating)
```

```java
RatingBar rating;

@ViewById(R.id.save_button)
Button saveButton;

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    Bundle bundledData = getIntent().getExtras();

    if (bundledData.containsKey("Theme")) {
        setTheme(bundledData.getInt("Theme"));
    }
    selectedImage = bundledData.getParcelable("ImageData");
    position = bundledData.getInt("position");

    // Create actionbar dropdown spinner from stringarray
    SpinnerAdapter mSpinnerAdapter = ArrayAdapter.createFromResource(
                                     getActionBar().getThemedContext(), R.array.text_sizes,
                                     android.R.layout.simple_spinner_dropdown_item);

    ActionBar.OnNavigationListener mOnNavigationListener = new ActionBar.OnNavigationListener() {
        @Override
        public boolean onNavigationItemSelected(int position, long itemId) {

            Log.d("METADATA", "Position: " + position + " ItemId: " + itemId);
            Intent intent = getIntent();

            if (navFired == true ) {
                switch (position) {
```

```
                case 0:
                    intent.putExtra("Theme", R.style.NormalText);
                    finish();
                    startActivity(intent);
                    break;
                case 1:
                    intent.putExtra("Theme", R.style.SmallText);
                    finish();
                    startActivity(intent);
                    break;
                case 2:
                    intent.putExtra("Theme", R.style.LargeText);
                    finish();
                    startActivity(intent);
                    break;
            }
        } else {
            navFired = true;
        }

        return true;
    }
};

getActionBar().setNavigationMode(ActionBar.NAVIGATION_MODE_LIST);
getActionBar().setListNavigationCallbacks(mSpinnerAdapter, mOnNavigationListener);

if (bundledData.containsKey("Theme")) {
    switch (bundledData.getInt("Theme")){
        case R.style.NormalText:
                getActionBar().setSelectedNavigationItem(0);
                break;
            case R.style.SmallText:
                getActionBar().setSelectedNavigationItem(1);
                break;
            case R.style.LargeText:
                getActionBar().setSelectedNavigationItem(2);
                break;
        }

    } else {
        getActionBar().setSelectedNavigationItem(0);
    }
}

@AfterViews
public void restoreData() {
    this.name.setText(selectedImage.getName());
    this.sourceURL.setText(selectedImage.getUrl());
    this.keywords.setText(selectedImage.getKeyWords());
    this.sourceEmail.setText(selectedImage.getSourceEmail());
    this.shareToggle.setChecked(selectedImage.getShare());
    this.date.updateDate(selectedImage.getDate().get(Calendar.YEAR),
                         selectedImage.getDate().get(Calendar.MONTH),
                         selectedImage.getDate().get(Calendar.DAY_OF_MONTH));
    this.rating.setRating((int)selectedImage.getRating().getStarRating());
}

/**
 * Need to override back press to have data saved
 * when the back arrow is pressed.
 */
@Override
public void onBackPressed(){
    Log.d("METADATA", "Back button pressed. Saving data...");
    onSave();
}

@Click(R.id.save_button)
public void onSave() {
    try {
        InputMethodManager imm =
                (InputMethodManager)this.getSystemService(Service.INPUT_METHOD_SERVICE);
        if (!Patterns.WEB_URL.matcher(sourceURL.getText().toString()).matches()){
            Toast urlMismatchToast = Toast.makeText(getApplicationContext(),
                                        "The source url is invalid", Toast.LENGTH_LONG);
            sourceURL.requestFocus();
            imm.showSoftInput(sourceURL, 0);

            urlMismatchToast.show();
        } else if (!Patterns.EMAIL_ADDRESS.matcher(sourceEmail.getText().toString()).matches()) {
            Toast emailMismatchToast = Toast.makeText(getApplicationContext(),
                                        "The email address is invalid", Toast.LENGTH_LONG);
            sourceEmail.requestFocus();
            imm.showSoftInput(sourceEmail, 0);

            emailMismatchToast.show();
        } else {
            this.selectedImage.setName(name.getText().toString());
            this.selectedImage.setUrl(Uri.parse(sourceURL.getText().toString()));
            this.selectedImage.setKeyWords(stringToArrayList(keywords.getText().toString()));
            this.selectedImage.setSourceEmail(sourceEmail.getText().toString());
            this.selectedImage.setShare(shareToggle.isChecked());
            Calendar givenDate = Calendar.getInstance();
            givenDate.set(date.getYear(), date.getMonth(), date.getDayOfMonth());
            this.selectedImage.setDate(givenDate);
            this.selectedImage.setRating(Rating.newStarRating(Rating.RATING_5_STARS,
                                                              rating.getNumStars()));
            Log.d("METADATA", new Integer(rating.getNumStars()).toString());

            Intent resultIntent = new Intent();
            resultIntent.putExtra("ImageData", selectedImage);
            resultIntent.putExtra("position", position);

            setResult(Activity.RESULT_OK, resultIntent);

            finish();
        }
    } catch ( ImageData.NameEmptyException e ) {
        Toast nameToast = Toast.makeText(getApplicationContext(),
                                         "Image must have a name", Toast.LENGTH_LONG);
        nameToast.show();
    } catch ( ImageData.EmailEmptyException e ) {
        Toast emailToast = Toast.makeText(getApplicationContext(),
                                          "Email field must not be empty", Toast.LENGTH_LONG);
        emailToast.show();
    }
}

public ArrayList<String> stringToArrayList(String s) {
        ArrayList<String> list = new ArrayList<String>();
        BufferedReader streamReader = new BufferedReader(new InputStreamReader(
                                                         new ByteArrayInputStream(s.getBytes())));

        String temp;
        try {
            while ((temp = streamReader.readLine()) != null) {
                list.add(temp);
            }
        } catch (IOException e) {
            Log.e("METADATA", "Error converting keywords to ArrayList");
        }

        return list;
    }
}
```

1.3.3. ImageData.java

```
package au.net.danielparker.metadata;

import android.media.Rating;
import android.net.Uri;
import android.os.Parcel;
import android.os.Parcelable;

import java.util.ArrayList;
import java.util.Calendar;

/**
```

```
 * Created by danielparker on 14/09/14.
 */
public class ImageData implements Parcelable{
    private int imageId;
    private String name;
    private Uri url;
    private ArrayList<String> keyWords = new ArrayList<String>();
    private Calendar date = Calendar.getInstance();
    private String sourceEmail;
    private Boolean share = false;
    private Rating rating = Rating.newUnratedRating(Rating.RATING_5_STARS);

    public static final Parcelable.Creator<ImageData> CREATOR
            = new Parcelable.Creator<ImageData>() {
        public ImageData createFromParcel(Parcel in) {
            return new ImageData(in);
        }

        public ImageData[] newArray(int size) {
            return new ImageData[size];
        }
    };

    public ImageData(Parcel in) {
        this.imageId = in.readInt();
        this.name = in.readString();
        this.url = (Uri)in.readValue(Uri.class.getClassLoader());
        this.keyWords = in.readArrayList(String.class.getClassLoader());
        date.setTimeInMillis(in.readLong());
        this.sourceEmail = in.readString();
        this.share = (Boolean)in.readValue(Boolean.class.getClassLoader());
        this.rating = (Rating)in.readValue(Rating.class.getClassLoader());
    }

    public ImageData(String name, String sourceEmail, int imageId) throws NameEmptyException,
                                                                          EmailEmptyException {
        setName(name);
        setSourceEmail(sourceEmail);
        setImageId(imageId);
    }

    public void writeToParcel(Parcel out, int flags) {
        out.writeInt(imageId);
        out.writeString(name);
        out.writeValue(url);
        out.writeList(keyWords);
        out.writeLong(date.getTimeInMillis());
        out.writeString(sourceEmail);
        out.writeValue(share);
        out.writeValue(rating);
    }

    public int describeContents() {
        return 0;
    }

    public int getImageId() {
        return imageId;
    }
```

```
public void setImageId(int imageId) {
    this.imageId = imageId;
}

public Boolean getShare() {
    return share;
}

public void setShare(Boolean share) {
    this.share = share;
}

public String getName() {
    return name;
}

public void setName(String name) throws NameEmptyException{
    if (name.isEmpty()) {
        throw new NameEmptyException();
    }
    this.name = name;
}

public String getUrl() {
    if (url != null) {
        return url.toString();
    } else {
        return "";
    }
}
```

```
public void setUrl(Uri url) {
    this.url = url;
}

public String getKeyWords() {
    StringBuilder stringBuilder = new StringBuilder();
    for (String s: keyWords) {
        stringBuilder.append(s + '\n');
    }
    return stringBuilder.toString();
}

public void setKeyWords(ArrayList<String> keyWords) {
    this.keyWords = keyWords;
}

public Calendar getDate() { return date; }

public void setDate(Calendar date) {
    this.date = date;
}

public String getSourceEmail() {
    return sourceEmail;
}

public void setSourceEmail(String sourceEmail) throws EmailEmptyException{
    if (sourceEmail.toString().isEmpty()) {
        throw new EmailEmptyException();
        }
        this.sourceEmail = sourceEmail;
    }

    public Rating getRating() {
        return rating;
    }

    public void setRating(Rating rating) {
        this.rating = rating;
    }

    public static class NameEmptyException extends Exception {
        public NameEmptyException() {

        }
    }

    public static class EmailEmptyException extends Exception {
        public EmailEmptyException() {

        }
    }
}
```

1.3.4. activity_gallery.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <TextView
        android:id="@+id/instructionText"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="@string/instructions"
        android:textSize="@dimen/instruction_size"
        android:paddingBottom="@dimen/std_padding"/>
    <ListView
        android:id="@android:id/list"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/instructionText">
    </ListView>
</LinearLayout>
```

1.3.5. gallery_item.xml

```
<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <ImageView
        android:id="@+id/Image"
        android:layout_width="@dimen/photo_height"
        android:layout_height="@dimen/photo_width"/>
    <TextView
        android:id="@+id/Image_Title"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingLeft="@dimen/std_padding"
        android:paddingRight="@dimen/std_padding"
        android:layout_toRightOf="@id/Image"
        android:textSize="?android:attr/labelTextSize"/>
    <TextView
        android:id="@+id/Image_Date"
        android:layout_width="fill_parent"
        android:layout_height   ="wrap_content"
        android:paddingLeft="@dimen/std_padding"
        android:paddingRight="@dimen/std_padding"
        android:layout_below="@id/Image_Title"
        android:layout_toRightOf="@id/Image"
        android:textSize="?android:attr/labelTextSize"/>
</RelativeLayout>
```

1.3.6. activity_edit_metadata.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content">
        <LinearLayout
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="@dimen/edge_margin"
```

```
android:layout_marginRight="@dimen/edge_margin">

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/metadata"
    android:textSize="@dimen/title_size"
    android:gravity="center"
    android:paddingTop="@dimen/std_padding"/>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="?android:labelTextSize"
    android:paddingLeft="@dimen/label_padding"
    android:text="@string/label_name"/>
<EditText
    android:id="@+id/name"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:singleLine="true"
    android:hint="@string/hint_name"
    android:layout_marginBottom="@dimen/std_padding"/>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="?android:attr/labelTextSize"
    android:paddingLeft="@dimen/label_padding"
    android:text="@string/label_location"/>
<EditText
    android:id="@+id/source_url"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:singleLine="true"
    android:hint="@string/hint_location"
    android:layout_marginBottom="@dimen/std_padding"
    android:inputType="textWebEditText" />
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="?android:attr/labelTextSize"
    android:paddingLeft="@dimen/label_padding"
    android:text="@string/label_keywords"/>
<MultiAutoCompleteTextView
    android:id="@+id/keywords"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/hint_keywords"
    android:layout_marginBottom="@dimen/std_padding">
</MultiAutoCompleteTextView>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textSize="?android:attr/labelTextSize"
    android:paddingLeft="@dimen/label_padding"
    android:text="@string/label_source"/>
<EditText
    android:id="@+id/source_email"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/hint_source"
    android:singleLine="true"
    android:inputType="textEmailAddress" />
<RelativeLayout
    android:id="@+id/share_section"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal">
    <TextView
        android:id="@+id/share_label"
        android:layout_width="fill_parent"
        android:layout_height="50dp"
        android:text="@string/label_share"
        android:textSize="?android:attr/labelTextSize"
        android:textAlignment="center"
        android:gravity="center_vertical"
        android:paddingLeft="40dp"
        android:paddingRight="40dp"/>
    <ToggleButton
        android:id="@+id/share_toggle"
        android:layout_width="100dp"
        android:layout_height="wrap_content"
        android:layout_alignParentRight="true" />
</RelativeLayout>

<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/label_date"
    android:textSize="?android:attr/labelTextSize"
    android:gravity="center"/>
```

```
            <DatePicker
                android:id="@+id/date"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:calendarViewShown="false"/>
            <TextView
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="@string/label_rating"
                android:textSize="?android:attr/labelTextSize"
                android:gravity="center_horizontal"/>
            <RatingBar
                android:id="@+id/rating"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_gravity="center_horizontal"
                android:numStars="5"
                android:stepSize="1"/>
            <Button
                android:id="@+id/save_button"
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:text="@string/button_save"/>
        </LinearLayout>
    </ScrollView>
</LinearLayout>
```

1.3.7. styles.xml

```xml
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="android:Theme.Holo.Light.DarkActionBar">
        <!-- Customize your theme here. -->
    </style>

    <style name="SmallText" parent="AppTheme">
        <item name="android:labelTextSize">10sp</item>
    </style>

    <style name="NormalText" parent="AppTheme">
        <item name="android:labelTextSize">16sp</item>
    </style>

    <style name="LargeText" parent="AppTheme">
        <item name="android:labelTextSize">22sp</item>
    </style>

</resources>
```

2. Usability Test

2.1. Objective

The purpose of this usability test was to identify what label font size felt most comfortable visually for users and also what information display font size was the best. It isn't always obvious what users will prefer, but doing a usability test like this one can help to identify the best font size for the situation.

2.2. Method

1. The three font sizes were selected by first selecting a font size that looked best in my own opinion. That value was then used as the middle font size and font sizes slightly smaller and larger were selected as the other two font sizes for the usability test.
2. The three selected font sizes were:
   - Small: 10sp
   - Medium: 16sp
   - Large: 22sp
3. The code used three different themes to change the font size. The three themes simple overrode the android value for labelTextSize, which was referenced as the text size for labels using the attribute *android:textSize="?android:attr/labelTextSize"*. The user can then select the size they want within the EditMetadata activity and the theme will be set and activity reloaded.

2.3. Participants

1. - Age: 55
   - Occupation: Personal Assistant
   - Computer usage: Daily
   - Smartphone: Owns a smartphone
2. - Age: 56
   - Occupation: Software Engineer
   - Computer usage: Daily
   - Smartphone: Does not own a smartphone

3.
- Age: 24
- Occupation: Waitress
- Computer usage: Once a month
- Smartphone: Owns a smartphone

2.4. Results

2.4.1. Participant 1

	Small	Medium	Large
Can you read the text?	Only Just	Yes	Yes
Do you prefer this font size for the labels?	No	Yes	No
Do you prefer this font size for the information display?	No	Yes	No

2.4.2. Participant 2

	Small	Medium	Large
Can you read the text?	No	Yes	Yes
Do you prefer this font size for the labels?	No	Yes	No
Do you prefer this font size for the information display?	No	Yes	Yes

2.4.3. Participant 3

	Small	Medium	Large
Can you read the text?	Yes	Yes	Yes
Do you prefer this font size for the labels?	No	Yes	No
Do you prefer this font size for the information display?	No	Yes	No

2.5. Recommendation

The recommendation from the results of the usability test is that the Medium 16sp font size is used for labels and information display. This is due to all of the tested users saying that they could read and preferred that size for the label and display information text.

2.6. Reflection

If the usability test were to be run again, the users would be show all the fonts first and then asked to only give one preferred out of the three fonts for the labels and display information.