Camera and Email Intent

Elena Fortini

Android's Intent

An Intent is a messaging object you can use to request an action from another app component. Although intents facilitate communication between components in several ways, there are three fundamental use-cases:

- To start an Activity
- To start a Service
- To start a Broadcast

Types of intents

There are two types of intents:

- **Explicit intents** specify the component to start by name (the fully-qualified class name). You'll typically use an explicit intent to start a component in your own app, because you know the class name of the activity or service you want to start.

- **Implicit intents** do not name a specific component, but instead declare a general action to perform, which allows a component from another app to handle it

When you create an explicit intent to start an activity or service, the system immediately starts the app component specified in the Intent object. When you create an implicit intent, the Android system finds the appropriate component to start by comparing the contents of the intent to the intent filters.

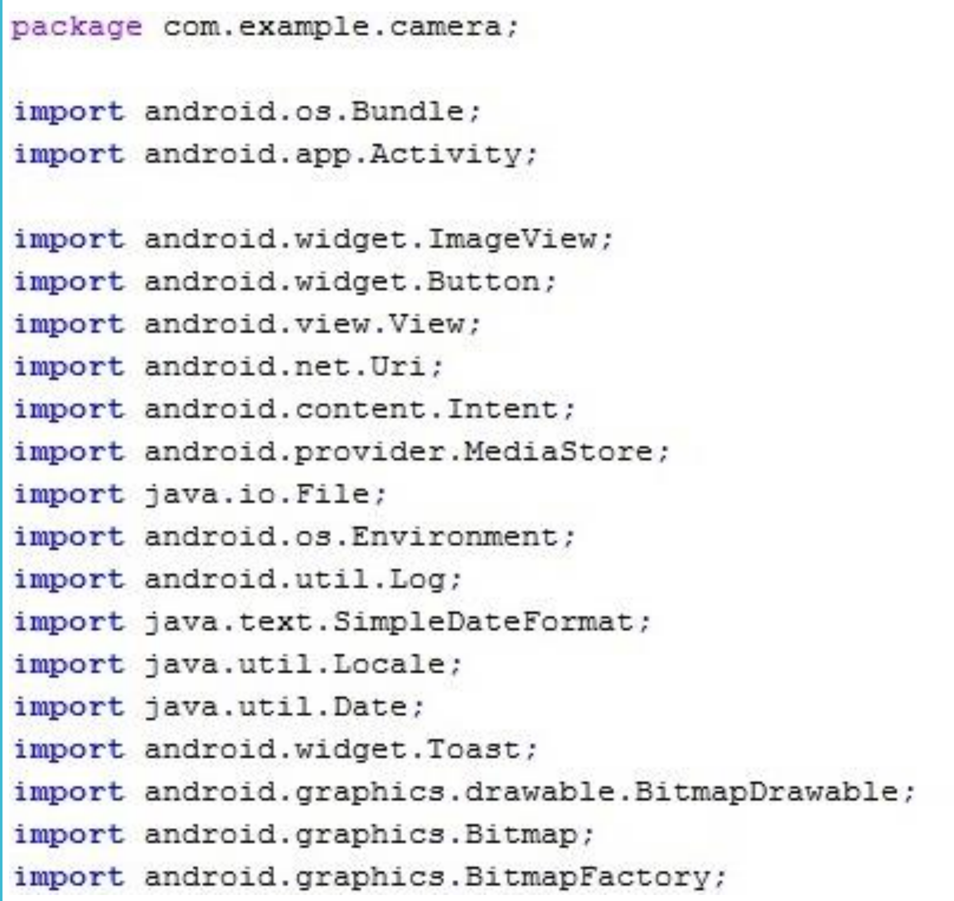
Camera Intent

Camera Intent

A quick way to enable taking pictures or videos in your application without a lot of extra code is to use an Intent to invoke an existing Android camera. To invoke the camera intent you need to follow these steps:

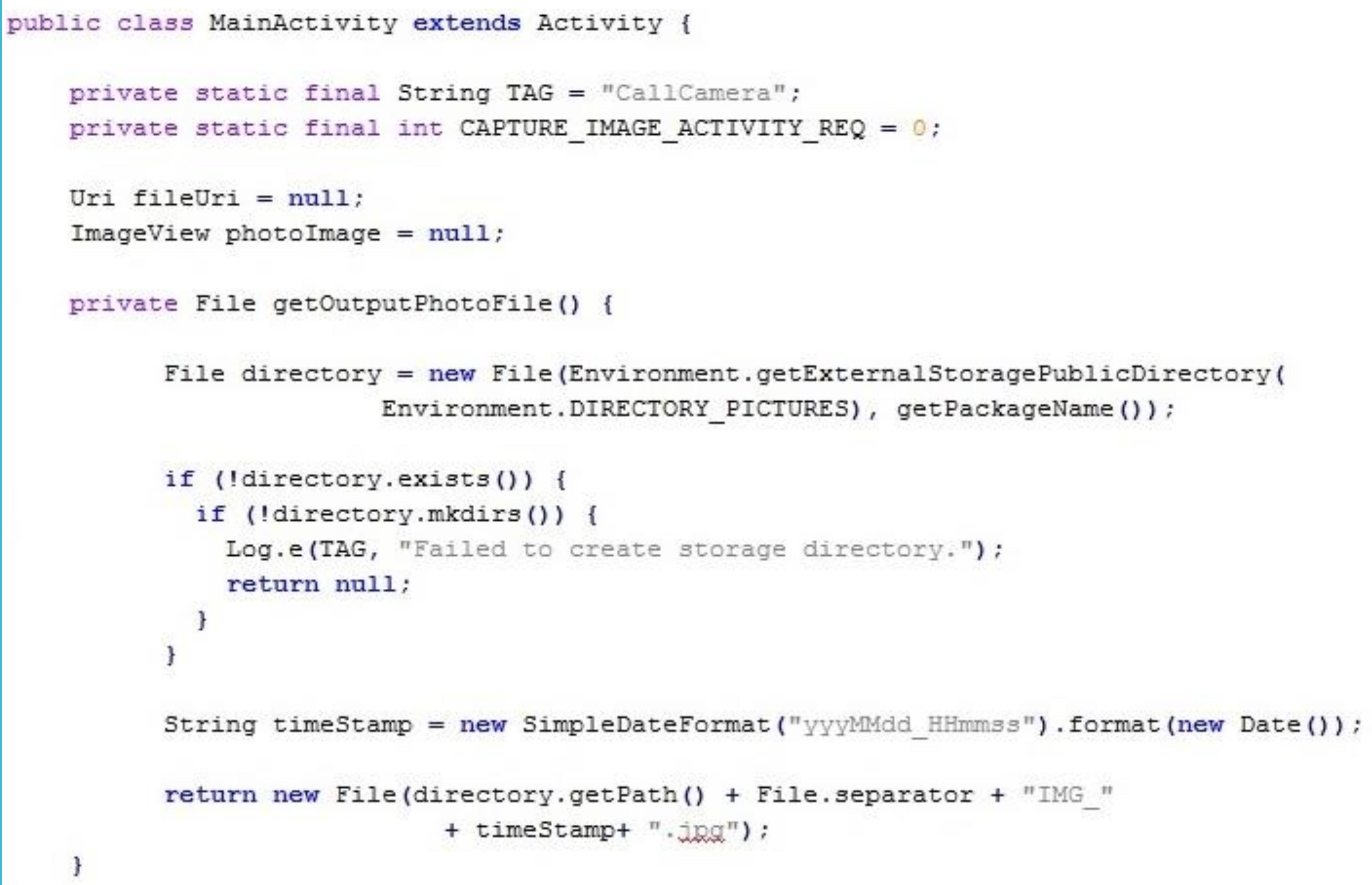
1. **Compose a Camera Intent** - Create an Intent that requests an image or video(there are two types of intent MediaStore.ACTION_IMAGE_CAPTURE and MediaStore.ACTION_VIDEO_CAPTURE);
2. **Start the Camera Intent** - Use the startActivityForResult() method. After you start the intent, the Camera application user interface appears on the device screen and the user can take a picture or video.
3. **Receive the Intent Result** - Set up an onActivityResult() method in your application to receive the callback and data from the camera intent.

Camera Intent – an example (1/5)



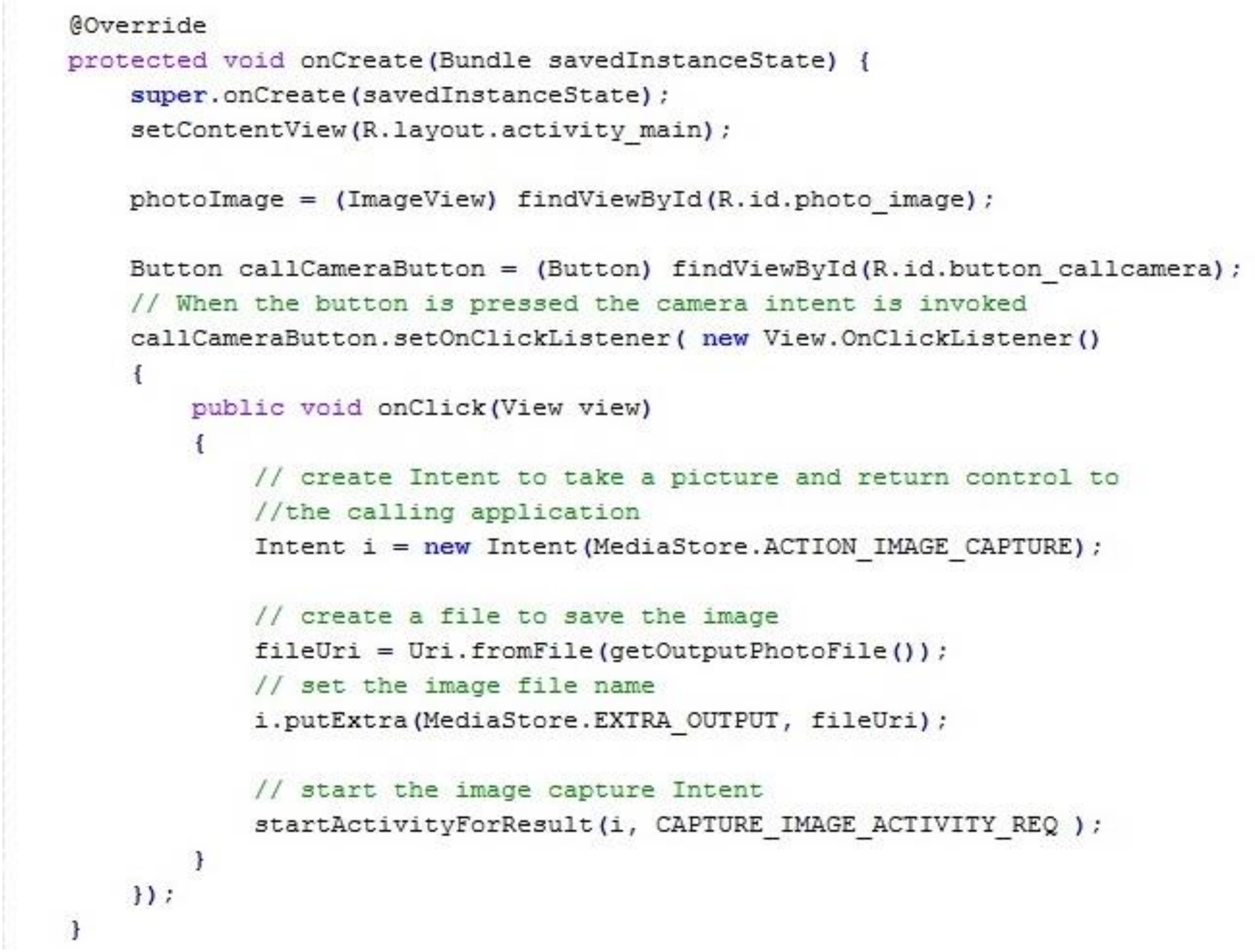
```
package com.example.camera;

import android.os.Bundle;
import android.app.Activity;

import android.widget.ImageView;
import android.widget.Button;
import android.view.View;
import android.net.Uri;
import android.content.Intent;
import android.provider.MediaStore;
import java.io.File;
import android.os.Environment;
import android.util.Log;
import java.text.SimpleDateFormat;
import java.util.Locale;
import java.util.Date;
import android.widget.Toast;
import android.graphics.drawable.BitmapDrawable;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
```

Camera Intent – an example (2/5)



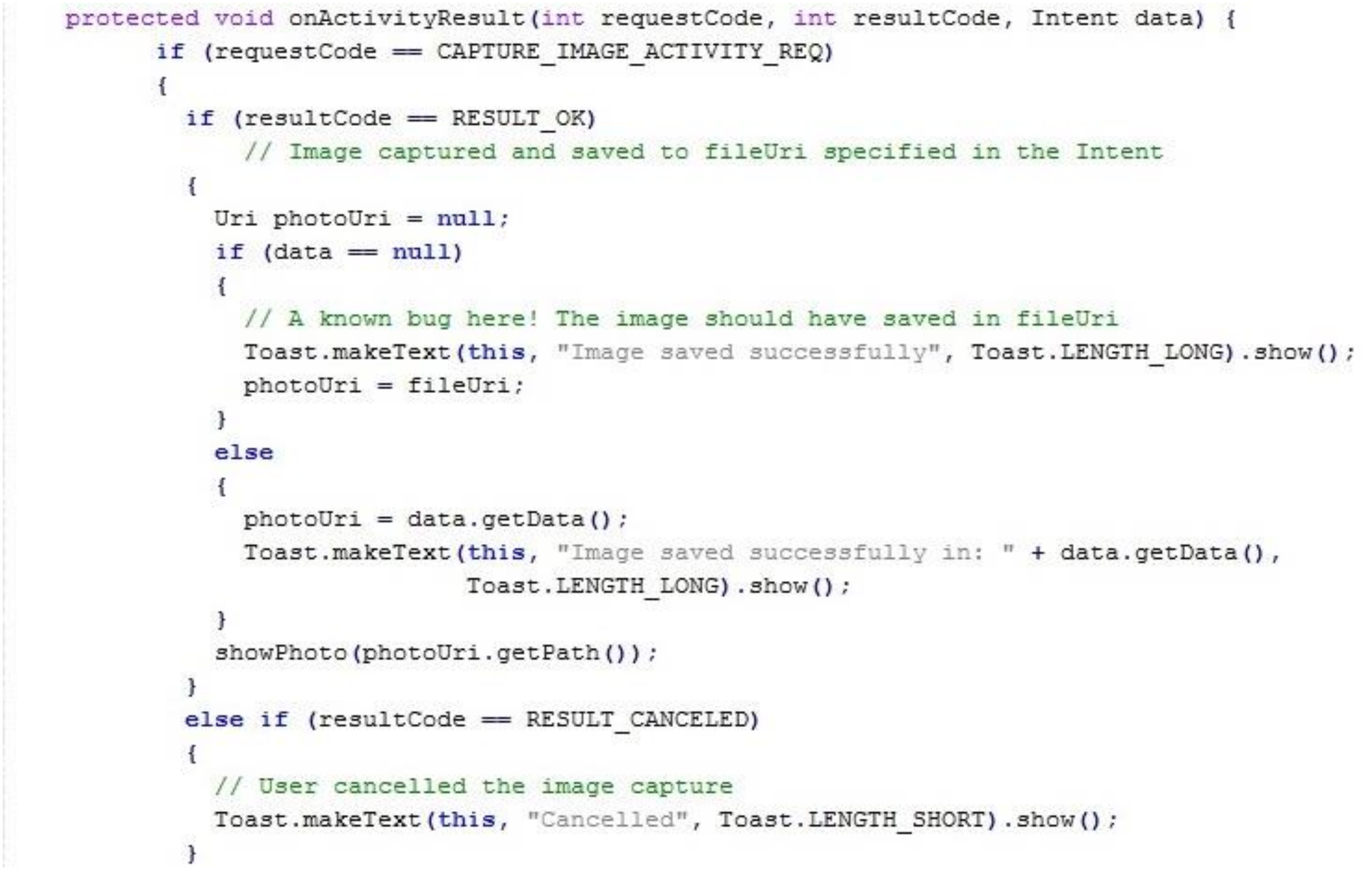
```
public class MainActivity extends Activity {

    private static final String TAG = "CallCamera";
    private static final int CAPTURE_IMAGE_ACTIVITY_REQ = 0;

    Uri fileUri = null;
    ImageView photoImage = null;

    private File getOutputPhotoFile() {

          File directory = new File(Environment.getExternalStoragePublicDirectory(
                        Environment.DIRECTORY_PICTURES), getPackageName());

          if (!directory.exists()) {
            if (!directory.mkdirs()) {
              Log.e(TAG, "Failed to create storage directory.");
              return null;
            }
          }

          String timeStamp = new SimpleDateFormat("yyyMMdd_HHmmss").format(new Date());

          return new File(directory.getPath() + File.separator + "IMG_"
                            + timeStamp+ ".jpg");
    }
```

Camera Intent – an example (3/5)



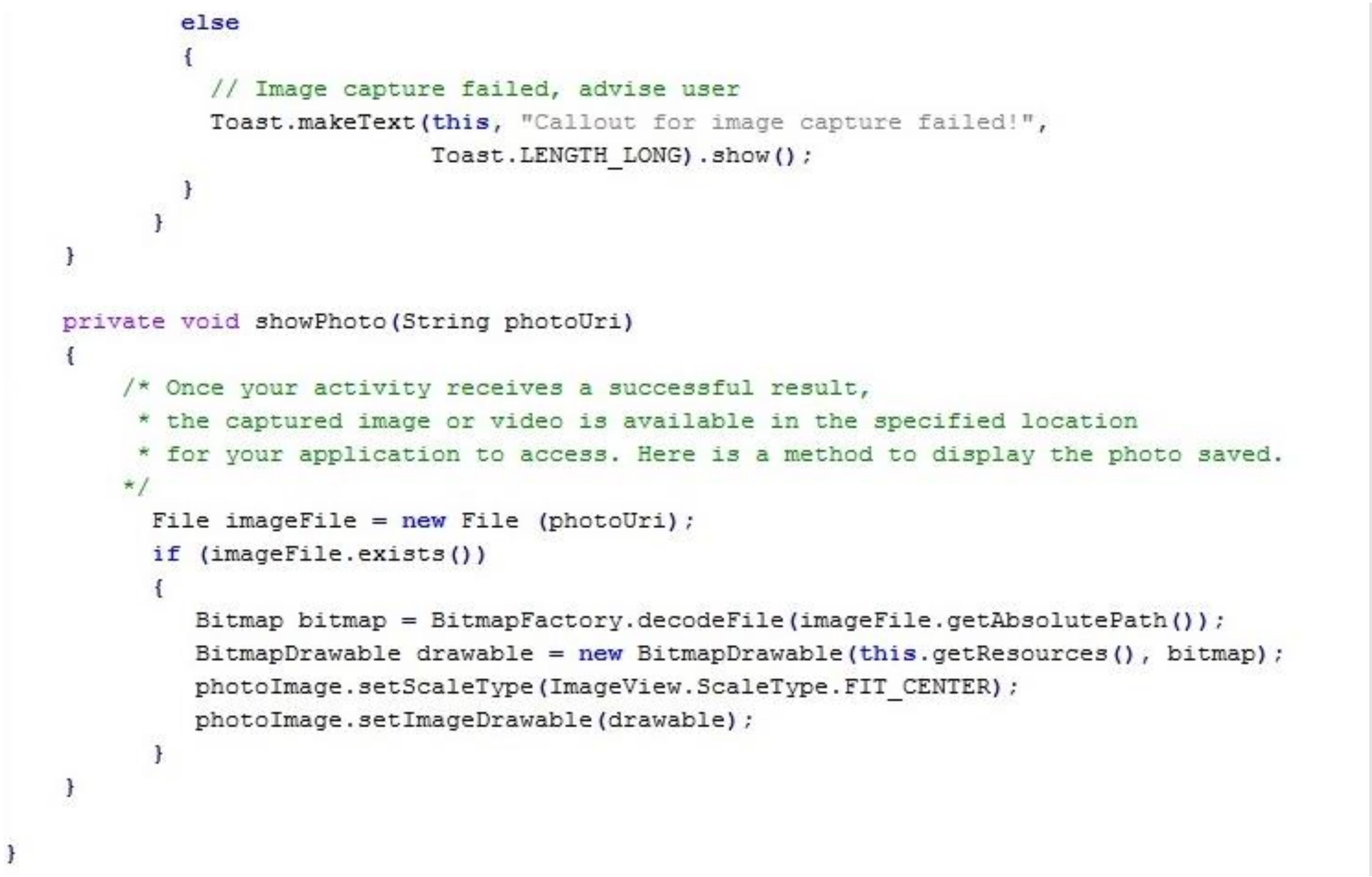
```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    photoImage = (ImageView) findViewById(R.id.photo_image);

    Button callCameraButton = (Button) findViewById(R.id.button_callcamera);
    // When the button is pressed the camera intent is invoked
    callCameraButton.setOnClickListener( new View.OnClickListener()
    {
        public void onClick(View view)
        {
            // create Intent to take a picture and return control to
            //the calling application
            Intent i = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

            // create a file to save the image
            fileUri = Uri.fromFile(getOutputPhotoFile());
            // set the image file name
            i.putExtra(MediaStore.EXTRA_OUTPUT, fileUri);

            // start the image capture Intent
            startActivityForResult(i, CAPTURE_IMAGE_ACTIVITY_REQ );
        }
    });
}
```

Camera Intent – an example (4/5)



```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
        if (requestCode == CAPTURE_IMAGE_ACTIVITY_REQ)
        {
          if (resultCode == RESULT_OK)
              // Image captured and saved to fileUri specified in the Intent
          {
            Uri photoUri = null;
            if (data == null)
            {
              // A known bug here! The image should have saved in fileUri
              Toast.makeText(this, "Image saved successfully", Toast.LENGTH_LONG).show();
              photoUri = fileUri;
            }
            else
            {
              photoUri = data.getData();
              Toast.makeText(this, "Image saved successfully in: " + data.getData(),
                                  Toast.LENGTH_LONG).show();
            }
            showPhoto(photoUri.getPath());
          }
          else if (resultCode == RESULT_CANCELED)
          {
            // User cancelled the image capture
            Toast.makeText(this, "Cancelled", Toast.LENGTH_SHORT).show();
          }
```

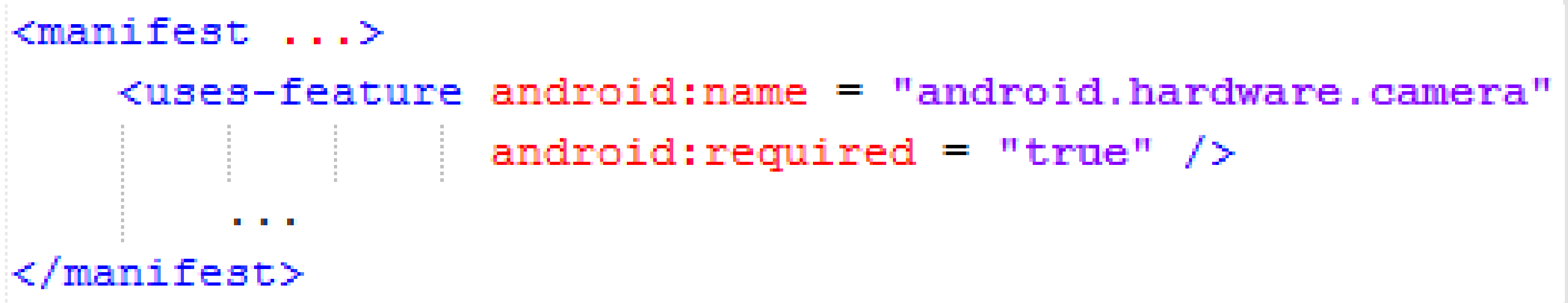
Camera Intent – an example (5/5)



```
            else
            {
              // Image capture failed, advise user
              Toast.makeText(this, "Callout for image capture failed!",
                                     Toast.LENGTH_LONG).show();
            }
          }
    }

    private void showPhoto(String photoUri)
    {
        /* Once your activity receives a successful result,
         * the captured image or video is available in the specified location
         * for your application to access. Here is a method to display the photo saved.
        */
          File imageFile = new File (photoUri);
          if (imageFile.exists())
          {
             Bitmap bitmap = BitmapFactory.decodeFile(imageFile.getAbsolutePath());
             BitmapDrawable drawable = new BitmapDrawable(this.getResources(), bitmap);
             photoImage.setScaleType(ImageView.ScaleType.FIT_CENTER);
             photoImage.setImageDrawable(drawable);
          }
    }

}
```

Camera Intent - The camera is essential!!!!

If an essential function of your application is taking pictures, then restrict its visibility on Google Play to devices that have a camera. To advertise that your application depends on having a camera, put a <uses-feature> tag in your manifest file:



```
<manifest ...>
    <uses-feature android:name = "android.hardware.camera"
                  android:required = "true" />
        ...
</manifest>
```

Email Intent

Email Intent

- In android we can use Intent.ACTION_SEND to call an existing email client to send an Email. If no email clients are configured then, android system displays an error such “No application can perform this action”.

- If we want to ensure that your intent is handled only by an email app (and not other text messaging or social apps), then use the ACTION_SENDTO action and include the "mailto:" data scheme.

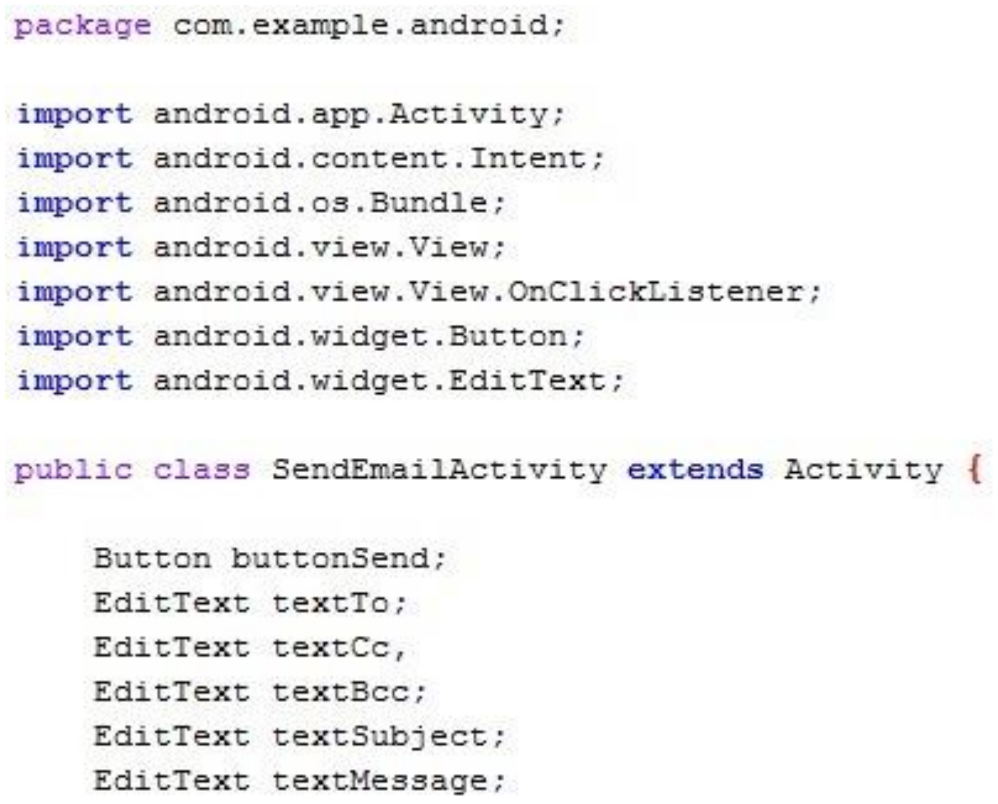
- In addittion, if we want to sent an email with multiple attachment we can use the ACTION_SEND_MULTIPLE.

Email Intent - Extras

With the method Intent.putExtra(·) is it possible to add more specific detail such as:

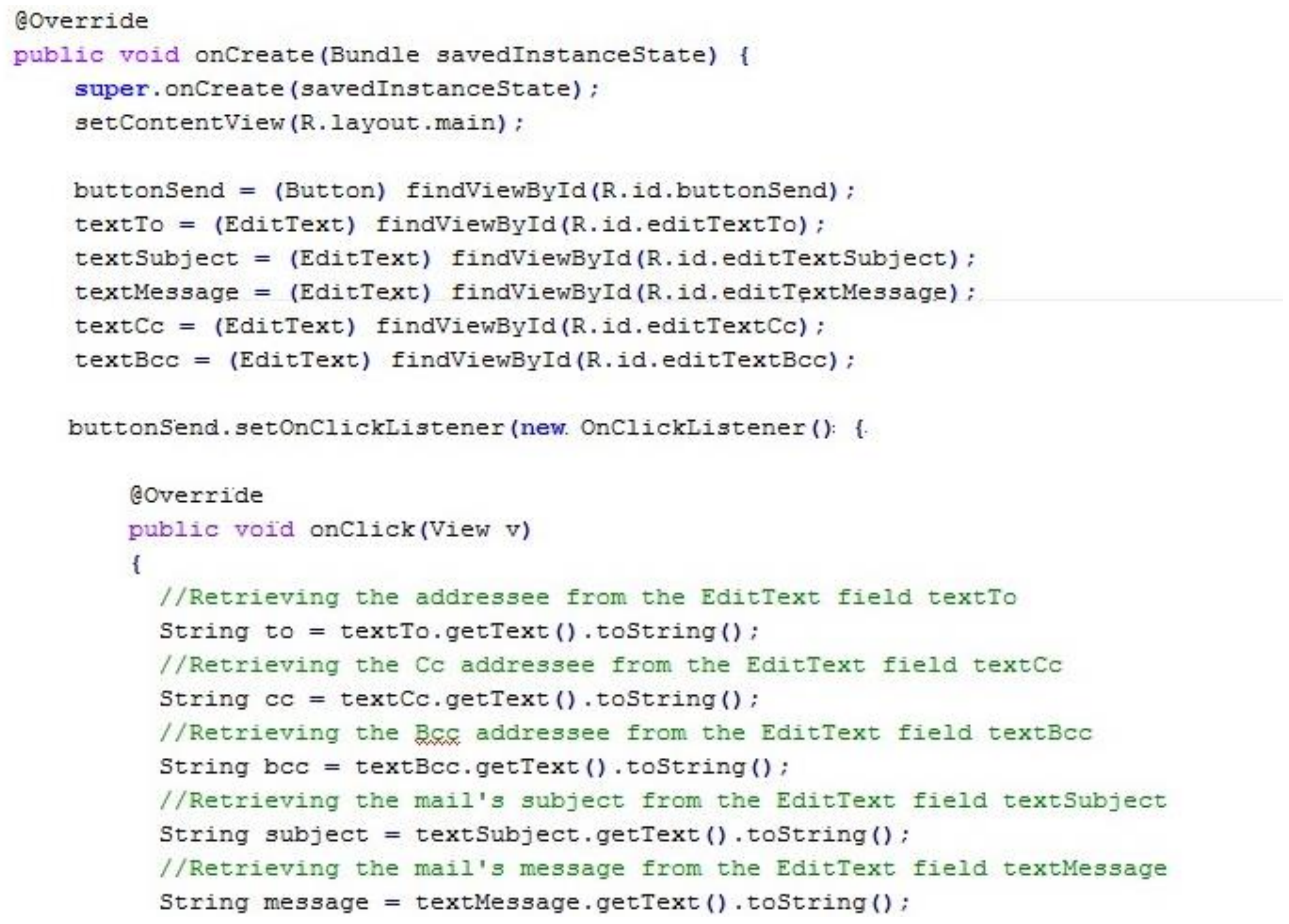
- **Intent.EXTRA_EMAIL**: A string array of all "To" recipient email addresses.
- **Intent.EXTRA_CC**: A string array of all "CC" recipient email addresses.
- **Intent.EXTRA_BCC**: A string array of all "BCC" recipient email addresses.
- **Intent.EXTRA_SUBJECT**: A string with the email subject.
- **Intent.EXTRA_TEXT**: A string with the body of the email.
- **Intent.EXTRA_STREAM**: A Uri pointing to the attachment. If using the ACTION_SEND_MULTIPLE action, this should instead be an ArrayList containing multiple Uri objects.

Email Intent – A simple example (1/3)



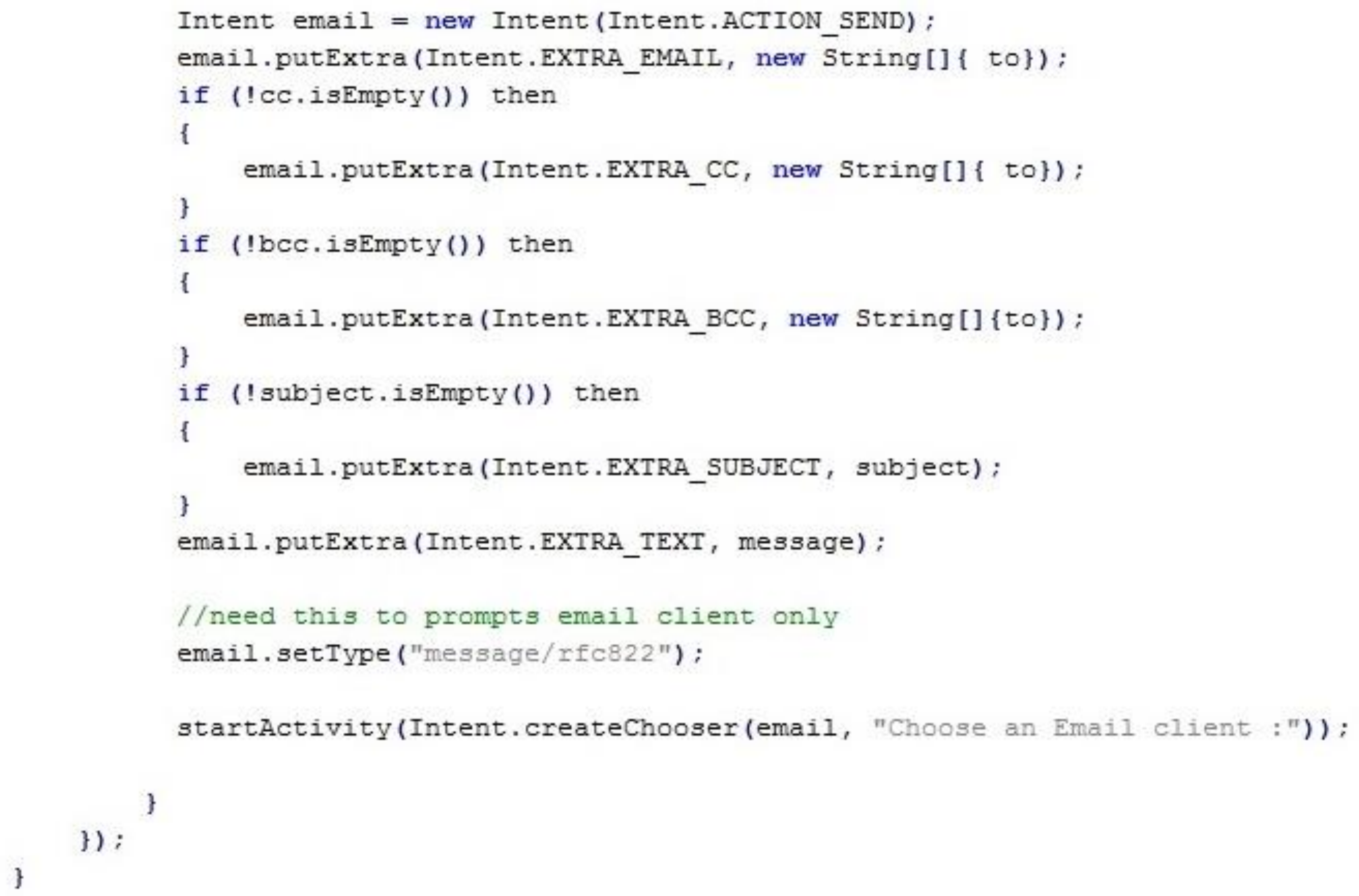
```java
package com.example.android;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

public class SendEmailActivity extends Activity {

    Button buttonSend;
    EditText textTo;
    EditText textCc,
    EditText textBcc;
    EditText textSubject;
    EditText textMessage;
```

Email Intent – A simple example (2/3)



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

    buttonSend = (Button) findViewById(R.id.buttonSend);
    textTo = (EditText) findViewById(R.id.editTextTo);
    textSubject = (EditText) findViewById(R.id.editTextSubject);
    textMessage = (EditText) findViewById(R.id.editTextMessage);
    textCc = (EditText) findViewById(R.id.editTextCc);
    textBcc = (EditText) findViewById(R.id.editTextBcc);

    buttonSend.setOnClickListener(new OnClickListener() {

        @Override
        public void onClick(View v)
        {
          //Retrieving the addressee from the EditText field textTo
          String to = textTo.getText().toString();
          //Retrieving the Cc addressee from the EditText field textCc
          String cc = textCc.getText().toString();
          //Retrieving the Bcc addressee from the EditText field textBcc
          String bcc = textBcc.getText().toString();
          //Retrieving the mail's subject from the EditText field textSubject
          String subject = textSubject.getText().toString();
          //Retrieving the mail's message from the EditText field textMessage
          String message = textMessage.getText().toString();
```

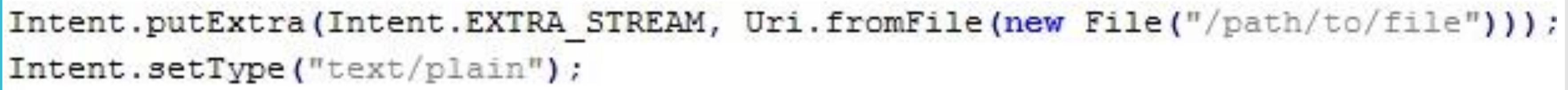
Email Intent – A simple example (3/3)



```
          Intent email = new Intent(Intent.ACTION_SEND);
          email.putExtra(Intent.EXTRA_EMAIL, new String[]{ to});
          if (!cc.isEmpty()) then
          {
              email.putExtra(Intent.EXTRA_CC, new String[]{ to});
          }
          if (!bcc.isEmpty()) then
          {
              email.putExtra(Intent.EXTRA_BCC, new String[]{to});
          }
          if (!subject.isEmpty()) then
          {
              email.putExtra(Intent.EXTRA_SUBJECT, subject);
          }
          email.putExtra(Intent.EXTRA_TEXT, message);

          //need this to prompts email client only
          email.setType("message/rfc822");

          startActivity(Intent.createChooser(email, "Choose an Email client :"));

        }
    });
}
```

Email Intent – How to send attachments (1/2)

To attach a single file, we add some extended data to ou intent:

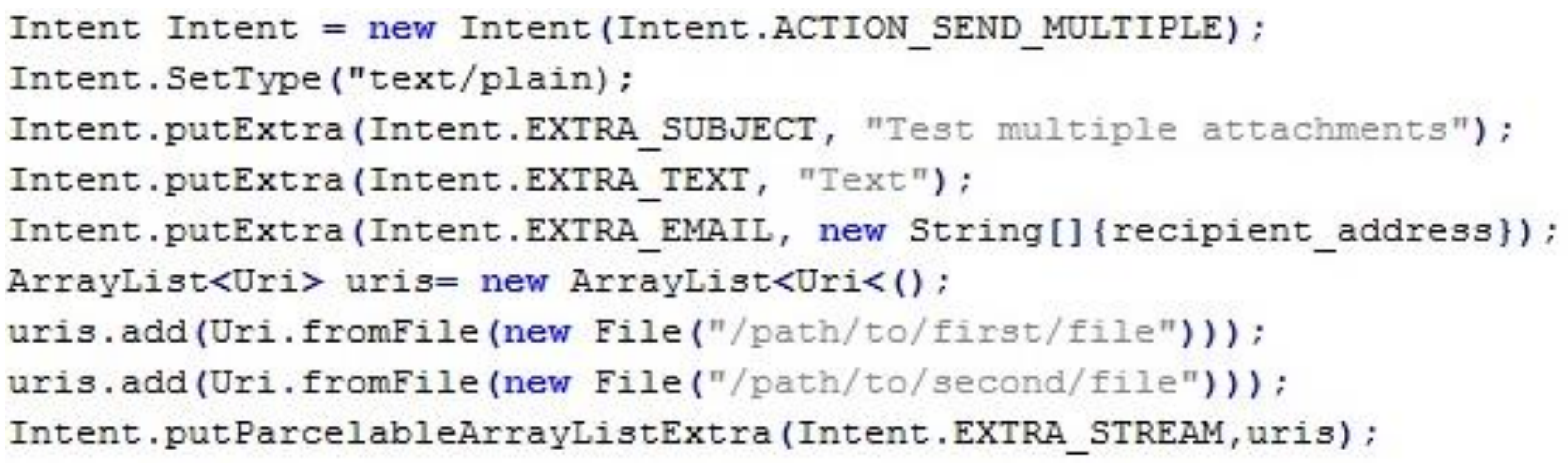


```
Intent.putExtra(Intent.EXTRA_STREAM, Uri.fromFile(new File("/path/to/file")));
Intent.setType("text/plain");
```

The MIMe type can always be set as text/plain but you may want to be more specific so application parsing your message will work properly. For istance if you're including a JPEG image you should write image/jpeg.

Email Intent – How to send attachments (2/2)

To send an email with multiple attachment the procedure is slightly different.



```
Intent Intent = new Intent(Intent.ACTION_SEND_MULTIPLE);
Intent.SetType("text/plain);
Intent.putExtra(Intent.EXTRA_SUBJECT, "Test multiple attachments");
Intent.putExtra(Intent.EXTRA_TEXT, "Text");
Intent.putExtra(Intent.EXTRA_EMAIL, new String[]{recipient_address});
ArrayList<Uri> uris= new ArrayList<Uri<();
uris.add(Uri.fromFile(new File("/path/to/first/file")));
uris.add(Uri.fromFile(new File("/path/to/second/file")));
Intent.putParcelableArrayListExtra(Intent.EXTRA_STREAM,uris);
```

If sending different type of files you may want to use multipart/mixed as MIME type

For practice…

Exercise

There are some exercises that you can solve to take confidence with what we've seen today:

- Complete the first two examples whit the XML file;
- Try to create an application for your phone that:
  - Has 2 button: a send botton and a picture one. When you press the second one a camera application is invoked. Then the picture is saved and ready to be sent as attachment. When you press the send botton the email Intent is invoked and the email is sent.
  - Has 3 EditText field: one for the addressee, one for the email's subject and one for the email's message.

Thank you!