Announce: Project 2

- Posted
- Grader re-checking some of the code
- Can read the project
- But wait for my email before starting
The Mobile Camera
Interesting application
Word Lens Feature of Google Translate

- Word Lens: translates text/signs in foreign Language in real time
- Example use case: tourist can understand signs, restaurant menus
- Uses Optical Character Recognition technology
- Google bought company in 2014, now part of Google Translate
Camera: Taking Pictures
Taking Pictures with Camera
Ref: https://developer.android.com/training/camera/photobasics.html

- How to take photos from your app using Android Camera app
- 4 Steps:
  1. Request the camera feature
  2. Take a Photo with the Camera App
  3. Get the Thumbnail
  4. Save the Full-size Photo
1. Request the Smartphone Camera Feature

Ref: https://developer.android.com/training/camera/photobasics.html

- If your app takes pictures using the phone’s Camera, you can allow only devices with a camera find your app while searching Google Play Store
- How?
- Make the following declaration in AndroidManifest.xml

```xml
<manifest ...>
  <uses-feature android:name="android.hardware.camera"
     android:required="true" />
... 
</manifest>
```
2. Capture an Image with the Camera App

Ref: https://developer.android.com/training/camera/photobasics.html

- To take picture, your app needs to send *implicit Intent* requesting for a picture to be taken (i.e. action = capture an image)
- Call `startActivityForResult()` with Camera intent since picture sent back
- Potentially, multiple apps/activities can handle this/take a picture
- Check that at least 1 Activity that can handle request to take picture using `resolveActivity`

**Diagram:**

- **Your App**
- **Android Camera app**
- `startActivityForResult`
- `onActivityResult`
Code to Take a Photo with the Camera App
Ref: https://developer.android.com/training/camera/photobasics.html

```java
static final int REQUEST_IMAGE_CAPTURE = 1;

private void dispatchTakePictureIntent() {
    Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    if (takePictureIntent.resolveActivity(getPackageManager()) != null) {
        startActivityForResult(takePictureIntent, REQUEST_IMAGE_CAPTURE);
    }
}
```

1. Build Intent, action = capture an image
2. Check that there’s at least 1 Activity that can handle request to capture an image (Avoids app crashing if no camera app available)
3. Send Intent requesting an image to be captured (usually handled by Android’s Camera app)
3. Get the Thumbnail

Ref: https://developer.android.com/training/camera/photobasics.html

- Android Camera app returns thumbnail of photo (small bitmap)

- Thumbnail bitmap returned in “extra” of Intent delivered to onActivityResult()

```java
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == REQUEST_IMAGE_CAPTURE && resultCode == RESULT_OK) {
        Bundle extras = data.getExtras();
        Bitmap imageBitmap = (Bitmap) extras.get("data");
        mImageView.setImageBitmap(imageBitmap);
    }
}
```
4. Save Full-Sized Photo
Ref: https://developer.android.com/training/basics/data-storage/files.html

- Android Camera app saves full-sized photo in a filename you give it
- We need phone owner’s permission to write to external storage
- Android systems have:
  - **Internal storage**: data stored here is available by only your app
  - **External storage**: available stored here is available to all apps
- Would like all apps to read pictures this app takes, so use external storage
Save Full-Sized Photo
Ref: https://developer.android.com/training/basics/data-storage/files.html

- Android Camera app can save full-size photo to
  1. **Public external storage** (shared by all apps)
     - getExternalStoragePublicDirectory()
     - Need to get permission
  2. **Private storage** (Seen by only your app, deleted when your app uninstalls):
     - getExternalFilesDir()

- Either way, need phone owner’s permission to write to external storage
- In AndroidManifest.xml, make the following declaration

```xml
<manifest ...>
  <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
  ...
</manifest>
```
Saving Full Sized Photo
Ref: https://developer.android.com/training/camera/photobasics.html

```java
static final int REQUEST_TAKE_PHOTO = 1;

private void dispatchTakePictureIntent() {
    Intent takePictureIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    // Ensure that there's a camera activity to handle the intent
    if (takePictureIntent.resolveActivity(getPackageManager()) != null) {
        // Create the File where the photo should go
        File photoFile = null;
        try {
            photoFile = createImageFile();
        } catch (IOException ex) {
            // Error occurred while creating the File
            ...
        }
        // Continue only if the File was successfully created
        if (photoFile != null) {
            Uri photoURI = FileProvider.getUriForFile(this,
                    "com.example.android.fileprovider",
                    photoFile);
            takePictureIntent.putExtra(MediaStore.EXTRA_OUTPUT, photoURI);
            startActivityForResult(takePictureIntent, REQUEST_TAKE_PHOTO);
        }
    }
}
```
Taking Pictures: Bigger Example
Taking Pictures with Intents
Ref: Ch 16 Android Nerd Ranch 3rd edition

- Would like to take picture of “Crime” to document it
- Use implicit intent to start Camera app from our CrimeIntent app
- **Recall:** Implicit intent used to call component in different activity
Create Placeholder for Picture

- Modify layout to include
  - ImageView for picture
  - Button to take picture
Create Layout for Thumbnail and Button

- First, build out left side
Create Title and Crime Entry EditText

- Build out right side
Get Handle of Camera Button and ImageView

- To respond to Camera Button click, in camera fragment, need handles to
  - Camera button
  - ImageView

```java
private Button mSuspectButton;
private Button mReportButton;
private ImageButton mPhotoButton;
private ImageView mPhotoView;
...

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
Bundle savedInstanceState) {
    ...
    PackageManager packageManager = getActivity().getPackageManager();
    if (packageManager.resolveActivity(pickContact,
            PackageManager.MATCH_DEFAULT_ONLY) == null) {
        mSuspectButton.setEnabled(false);
    }

    mPhotoButton = (ImageButton) v.findViewById(R.id.crime_camera);
    mPhotoView = (ImageView) v.findViewById(R.id.crime_photo);

    return v;
}
Firing Camera Intent

private static final int REQUEST_DATE = 0;
private static final int REQUEST_CONTACT = 1;
private static final int REQUEST_PHOTO = 2;

... 
@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
    ...
    mPhotoButton = (ImageButton) v.findViewById(R.id.crime_camera);
    final Intent captureImage = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    boolean canTakePhoto = mPhotoFile != null &&
            captureImage.resolveActivity(getPackageManager()) != null;
    mPhotoButton.setEnabled(canTakePhoto);
    mPhotoButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Uri uri = FileProvider.getUriForFile(getActivity(),
                    "com.bignerdranch.android.criminalintent.fileprovider",
                    mPhotoFile);
            captureImage.putExtra(MediaStore.EXTRA_OUTPUT, uri);
            List<ResolveInfo> cameraActivities = getActivity().
                    getPackageManager().queryIntentActivities(captureImage,
                    PackageManager.MATCH_DEFAULT_ONLY);

            for (ResolveInfo activity : cameraActivities) {
                try {
                    activity.activityInfo.packageName;
                }
            }
            startActivityForResult(captureImage, REQUEST_PHOTO);
        }
    });
    mPhotoView = (ImageView) v.findViewById(R.id.crime_photo);
    return v;
}

Create new intent for image capture

Check with PackageManager that a Camera exists on this phone

Build Uri location to store image, Put image URI into Intents extra

Take picture
Declaring Features

- Declaring “uses-features”. But “android:required=false” means app prefers to use this feature.
- Phones without a camera will still “see” and on Google Play Store and can download this app.

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.bignerdranch.android.criminalintent">

    <uses-feature android:name="android.hardware.camera"
        android:required="false"/>

</manifest>
```
Face Recognition
Face Recognition

- Answers the question:

Who is this person in this picture?

Example answer: John Smith

- Compares unknown face to database of faces (or facial attributes) with known identity
- Neural networks/deep learning now makes comparison faster
FindFace App: Stalking on Steroids?

- See stranger you like? Take a picture
- App searches 1 billion pictures using neural networks < 1 second
- Finds person’s picture, identity, link on VK (Russian Facebook)
  - You can send friend Request
- ~ 70% accurate!
- Can also upload picture of celebrity you like
- Finds 10 strangers on Facebook who look similar, can send friend request
FindFace App

- Also used in law enforcement
  - Police identify criminals on watchlist

Face Detection
Mobile Vision API
https://developers.google.com/vision/

- **Face Detection**: Are there [any] faces in this picture?
- **How**? Locate face in photos and video and
  - **Facial landmarks**: Eyes, nose and mouth
  - **State of facial features**: Eyes open? Smiling?
Face Detection: Google Mobile Vision API
Ref: https://developers.google.com/vision/face-detection-concepts

- Detects faces:
  - reported at a position, with size and orientation
  - Can be searched for landmarks (e.g. eyes and nose)

<table>
<thead>
<tr>
<th>Orientation</th>
<th>Landmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y = 60^\circ, r = 45^\circ$</td>
<td><img src="image1" alt="Landmarks" /></td>
</tr>
<tr>
<td>$y = 0^\circ, r = 45^\circ$</td>
<td></td>
</tr>
<tr>
<td>$y = -60^\circ, r = 45^\circ$</td>
<td></td>
</tr>
<tr>
<td>$y = 0^\circ, r = 0^\circ$</td>
<td></td>
</tr>
<tr>
<td>$y = 60^\circ, r = 0^\circ$</td>
<td></td>
</tr>
<tr>
<td>$y = -60^\circ, r = 0^\circ$</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Euler Y angle</th>
<th>detectable landmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt; -36$ degrees</td>
<td>left eye, left mouth, left ear, nose base, left cheek</td>
</tr>
<tr>
<td>$-36$ degrees to $-12$ degrees</td>
<td>left mouth, nose base, bottom mouth, right eye, left eye, left cheek, left ear tip</td>
</tr>
<tr>
<td>$-12$ degrees to $12$ degrees</td>
<td>right eye, left eye, nose base, left cheek, right cheek, left mouth, right mouth, bottom mouth</td>
</tr>
<tr>
<td>$12$ degrees to $36$ degrees</td>
<td>right mouth, nose base, bottom mouth, left eye, right eye, right cheek, right ear tip</td>
</tr>
<tr>
<td>$&gt; 36$ degrees</td>
<td>right eye, right mouth, right ear, nose base, right cheek</td>
</tr>
</tbody>
</table>
Google Mobile Vision API

- Mobile Vision API also does:
  - **Face tracking:** detects faces in consecutive video frames
  - **Classification:** Eyes open? Face smiling?

- Classification:
  - Determines whether a certain facial characteristic is present
  - API currently supports 2 classifications: eye open, smiling
  - Results expressed as a confidence that a facial characteristic is present
    - Confidence > 0.7 means facial characteristic is present
    - E.g. > 0.7 confidence means it’s likely person is smiling

- Mobile vision API does face **detection** but **NOT recognition**
Face Detection

- **Face detection**: Special case of object-class detection
- **Object-class detection task**: find locations and sizes of all objects in an image that belong to a given class.
  - E.g: bottles, cups, pedestrians, and cars
- **Object matching**: Objects in picture compared to objects in database of labelled pictures
Mobile Vision API: Other Functionality

- Barcode scanner
- Optical Character Recognition (OCR): Recognize text
Face Detection Using Google’s Mobile Vision API
Getting Started with Mobile Vision Samples
https://developers.google.com/vision/android/getting-started

- **New**: Mobile vision API now part of ML kit
- Get **Android Play Services SDK** level 26 or greater
- Download mobile vision samples from github
Creating the Face Detector
Ref: https://developers.google.com/vision/android/detect-faces-tutorial

- In app’s **onCreate** method, create face detector

```java
FaceDetector detector = new FaceDetector.Builder(context)
    .setTrackingEnabled(false)
    .setLandmarkType(FaceDetector.ALL_LANDMARKS)
    .build();
```

- **detector** is base class for implementing specific detectors. E.g. face detector, bar code detector
- Tracking finds same points in multiple frames (continuous)
- Detection works best in single images when **trackingEnabled** is false
Detecting Faces and Facial Landmarks

- Create Frame (image data, dimensions) instance from bitmap supplied

```java
Frame frame = new Frame.Builder().setBitmap(bitmap).build();
```

- Call detector synchronously with frame to detect faces

```java
SparseArray<Face> faces = detector.detect(frame);
```

- Detector takes Frame as input, outputs array of Faces detected
- Face is a single detected human face in image or video
- Iterate over array of faces, landmarks for each face, and draw the result based on each landmark’s position

```java
for (int i = 0; i < faces.size(); ++i) {
    Face face = faces.valueAt(i);
    for (Landmark landmark : face.getLandmarks()) {
        int cx = (int) (landmark.getPosition().x * scale);
        int cy = (int) (landmark.getPosition().y * scale);
        canvas.drawCircle(cx, cy, 10, paint);
    }
}
```
Other Stuff

- To count faces detected, call `faces.size()`. E.g.

```java
TextView faceCountView = (TextView) findViewById(R.id.face_count);
faceCountView.setText(faces.size() + " faces detected");
```

- Querying Face detector’s status

```java
if (!detector.isOperational()) {
    // ...
}
```

- Releasing Face detector (frees up resources)

```java
detector.release();
```
Detect & Track Multiple Faces in Video

- Can also track multiple faces in image sequences/video, draw rectangle round each one
Face Interpretation
Visage Face Interpretation Engine

- Real-time face interpretation engine for smart phones
  - Tracking user’s 3D head orientation + facial expression

- Facial expression, affect, emotion
  - angry, disgust, fear, happy, neutral, sad, surprise
  - Use? Can be used in Mood Profiler app

Facial Expression Inference

- Active appearance model
  - Describes 2D image as triangular mesh of landmark points
- 7 expression classes: angry, disgust, fear, happy, neutral, sad, surprise
- Extract triangle shape, texture features
- Classify features using Machine learning

Classification Accuracy

<table>
<thead>
<tr>
<th>Expressions</th>
<th>Anger</th>
<th>Disgust</th>
<th>Fear</th>
<th>Happy</th>
<th>Neutral</th>
<th>Sadness</th>
<th>Surprise</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy(%)</td>
<td>82.16</td>
<td>79.68</td>
<td>83.57</td>
<td>90.30</td>
<td>89.93</td>
<td>73.24</td>
<td>87.52</td>
</tr>
</tbody>
</table>
References

- Busy Coder’s guide to Android version 4.4
- CS 65/165 slides, Dartmouth College, Spring 2014
- CS 371M slides, U of Texas Austin, Spring 2014
- Android Nerd Ranch, 1st edition