Lecture 8

- Camera
- Location
- Google Maps

Camera

Your app requests the camera app that is installed in the device.

To check that there is a camera app installed, you can create a camera intent, and then call the getResolveActivity()

```
Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
if (intent.resolveActivity(getPackageManager()) != null) {
    // prepare intent
    activity.startActivityForResult(intent, appCode);
}
```

The intent contains the necessary data to start the camera application.

You can also check this without an intent:

```
if(activity.getPackageManager().hasSystemFeature(
                PackageManager.FEATURE_CAMERA)) {
    // Prepare intent and start the camera app
    activity.startActivityForResult(intent, appCode);
}
```

Camera - thumbnail

You can choose to get a thumbnail or a picture out of the camera app.
Getting a thumbnail:

```
static final int THUMBNAIL = 1;


Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
if (intent.resolveActivity(getPackageManager()) != null) {
           activity.startActivityForResult(intent, THUMBNAIL);
}
```

onActivityResult provides the resulting Thumbnail as a bitmap.

```
protected void onActivityResult(int requestCode, int resultCode, Intent
    data) {
    super.onActivityResult(requestCode, resultCode, data);
    if(requestCode==THUMBNAIL && resultCode== Activity.RESULT_OK) {
        Bitmap thumbnail = data.getParcelableExtra("data");
        ivThumbnail.setImageBitmap(thumbnail); // gör något med thumbnailen
    } else {

    }
}
```

Camera - picture

Choosing a picture requires to provide a URI, pointing to the place in the device where the picture will be stored.

```
static final int PICTURE = 2;
:
Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

if (intent.resolveActivity(getPackageManager()) != null) {
    String timeStamp = new SimpleDateFormat("yyyyMMdd_HHmmss")
                                   .format(new Date());
    String filename = "JPEG_" + timeStamp + ".jpg";
    File dir = getExternalFilesDir(Environment.DIRECTORY_PICTURES)
    pictureUri = Uri.fromFile(new File(dir, filename));
    intent.putExtra(MediaStore.EXTRA_OUTPUT, pictureUri);
    startActivityForResult(intent, PICTURE);

}
```

Camera - picture

Retrieve the picture (as a Bitmap) from the URI in onActivityResult.

```
protected void onActivityResult(int requestCode, int resultCode, Intent
    data) {

    super.onActivityResult(requestCode, resultCode, data);
    if(requestCode==THUMBNAIL && resultCode== Activity.RESULT_OK) {
        // thumbnail
    }
    else if(requestCode==PICTURE && resultCode== Activity.RESULT_OK){

        String pathToPicture = pictureUri.getPath();
        ivPicture.setImageBitmap(getScaled(pathToPicture,500,500));
    }
```

Resize the resulting bitmap:

```
private Bitmap getScaled(String pathToPicture,int width,int height) {
    BitmapFactory.Options bmOptions = new BitmapFactory.Options();
    bmOptions.inJustDecodeBounds = true;
    BitmapFactory.decodeFile(pathToPicture, bmOptions);

    int photoW = bmOptions.outWidth;
    int photoH = bmOptions.outHeight;
    int scaleFactor = Math.min(photoW/targetW, photoH/targetH);
    bmOptions.inJustDecodeBounds = false;
    bmOptions.inSampleSize = scaleFactor;
    Bitmap bitmap = BitmapFactory.decodeFile(pathToPicture, bmOptions);
    return bitmap;
    }
```

Camera - picture

You will need to add this permission to the manifest in order to handle the camera from your app.

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

Location

With classes Location Manager and Location and interface

Location Listener can obtain the phone's position.

The position can be obtained using:

- Mobile network or WiFi
- GPS

Using the network requires the following permission:

```
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
```

Using the GPS requires a different permission:

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

which includes the permission ACCESS_COARSE_LOCATION

Location

A LocationManager is obtained by calling getSystemService.

```
private LocationManager locationManager;


locationManager = (LocationManager)
                      getSystemService(Context.LOCATION_SERVICE);
```

Add a listener to onResume so that the position can be retrieved every time the Activity is resumed:

```
protected void onResume() {
    super.onResume();
    locationManager.requestLocationUpdates(
        LocationManager.GPS_PROVIDER, 1000, 0, locationListener);
}
```

Remove the listener in onPause.

```
protected void onPause() {
    locationManager.removeUpdates(locationListener);
    super.onPause();
}
```

Location

Then create a class that implements the locationListener:

```
private LocationManager locationManager;
private LocationListener locationListener;


locationListener = new LocList();

private class LocList implements LocationListener {
    // Called when the location has changed.
    public void onLocationChanged(Location location) {
        double latitude = location.getLatitude();
        double longitude = location.getLongitude();
        Log.d("onLocChanged","Lng="+longitude+",Lat="+latitude);
    }

    // Called when the provider is disabled by the user.
    public void onStatusChanged(String provider, int status, Bundle extras) {}

    // Called when the provider is enabled by the user.
    public void onProviderEnabled(String provider) {}

    // Called when the provider status changes.
    public void onProviderDisabled(String provider) {}
}
```

Location

Since Android 6.0, the permission must be declared in the manifest and requested from the source code.

```
if (ContextCompat.checkSelfPermission(this,
android.Manifest.permission.ACCESS_FINE_LOCATION)== PackageManager.PERMISSION_DENIED) {
    ActivityCompat.requestPermissions(this, new
          String[]{android.Manifest.permission.ACCESS_FINE_LOCATION},
          REQUEST_ACCESS_FINE_LOCATION);
}

else {
    locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0, 0,
          locationListener);
    locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0,
          locationListener);
}
```

Location

The answer to the permission request is handled with the following callback:

```
public void onRequestPermissionsResult(int requestCode, String[] permissions, int[]
grantResults) {
    switch (requestCode) {
        case REQUEST_ACCESS_FINE_LOCATION :
            if (ContextCompat.checkSelfPermission(this,
          android.Manifest.permission.ACCESS_FINE_LOCATION) ==
          PackageManager.PERMISSION_GRANTED) {
                locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0,
                    0, locationListener);
                locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0,
                    locationListener);
            }
            break;
    }
}
```

Google Maps

In order to add Google Maps to your app:

- Install Google Play Services and add it to your Project (Gradle config file)
- Create a Google Maps Project in Android Studio.
- Get an API key following the instructions in the **google_maps_api.xml** file (in your Project, under res/values).
- Place the API key in the manifest (this is done automatically if you created the Google Maps Project)

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="@string/google_maps_key" />
```

Google Maps - MapFragment

Include a Fragment in your layout for displaying the map:

```
<fragment
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:name="com.google.android.gms.maps.MapFragment"
            android:id="@+id/map"
            android:layout_gravity="center_horizontal"    />
```

Retrieve the Fragment from the code as a SupportMapFragment:

```
SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
        .findFragmentById(R.id.map);
mapFragment.getMapAsync(this);
```

Override this callback. Otherwise, you will not get the map when it’s ready:

```
@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;
    mapReady = true;
}
```

Google Maps - GoogleMap

Add a marker and center the map using addMarker and moveCamera.

Locations are stored using LatLng variables:

```
private void addMarker(LatLng latLng) {
    addMarker = false;
    MarkerOptions mo = new MarkerOptions().position(latLng).title("My position");
    mMap.addMarker(mo);
    mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(latLng, 3));

}
```

MarkerOptions can be set to include icons:

```
MarkerOptions mo = new MarkerOptions()
        .position(latLng)
        .title("My thumbnail")
        .icon(BitmapDescriptorFactory.fromBitmap(thumbnail));
```