



# LẬP TRÌNH DI ĐỘNG

---

## Bài 12: Media Services (2) + Location Base Services



# Nhắc lại bài trước

---

- Làm việc với chức năng viễn thông (telephony)
  - Đọc trạng thái của điện thoại: để ứng dụng có cách ứng xử phù hợp
  - Gửi/Nhận/Đọc tin nhắn
  - Quay số và nhận cuộc gọi
- Các chức năng đa phương tiện
  - MediaStore: cơ sở dữ liệu của hệ thống về các file đa phương tiện (hình ảnh, âm thanh, video,...) trên thiết bị
  - Chơi file audio
  - Ghi âm



# Nội dung

---

## 1. Media Services (part II)

- Video
- TTS
- Camera

## 2. Location Base Services

- Global Positioning Services
- Geocoding Locations
- Mapping Locations



Phần 1.1

# Video



# Video playback

---

- Android OS có 2 cách để chơi lại các tập tin video
  - Sử dụng **VideoView** kết hợp với **MediaController**
  - Sử dụng **MediaPlayer** và **SurfaceView**
- Chơi lại video không yêu cầu quyền gì đặc biệt, nhưng nếu file video ở ngoài internet, thì ứng dụng cần có quyền truy cập internet
- Phương pháp thứ 2 cho phép lập trình viên thiết lập các bộ filter cho hình ảnh phát ra thông qua hàm **setPreviewCallback(filter)**, cần có kiến thức tốt về video nếu muốn viết filter



# VideoView + MediaController

---

- VideoView là view dùng để hiển thị dữ liệu video
- VideoView cung cấp các hàm để điều khiển quá trình chơi video: **start**, **pause**, **suspend**, **resume**, **stopPlayback**, **seekTo**(millis)
- MediaController là widget cung cấp các điều khiển cơ bản cho video, ngoài ra cũng cho lập trình viên tùy biến điều khiển các nút **next** và **prev**
- VideoView và MediaController được thiết kế để làm việc với nhau và cùng đáp ứng trải nghiệm người dùng khi chơi video (xử lý các sự kiện chạm)



# VideoView + MediaController

---

<RelativeLayout

```
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
xmlns:tools="http://schemas.android.com/tools"
```

```
android:layout_width="match_parent"
```

```
android:layout_height="match_parent" >
```

<VideoView

```
android:id="@+id/videoView1"
```

```
android:layout_width="fill_parent"
```

```
android:layout_height="match_parent"
```

```
android:layout_alignParentBottom="true"
```

```
android:layout_alignParentLeft="true"
```

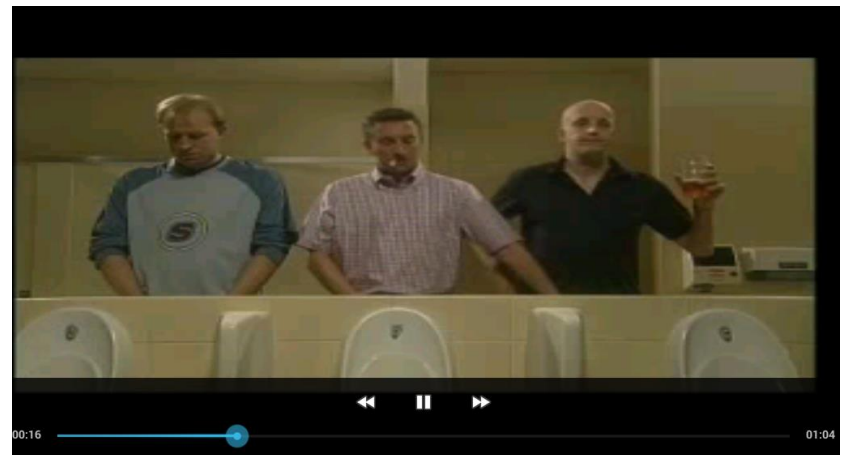
```
android:layout_alignParentRight="true"
```

```
android:layout_alignParentTop="true" />
```

</RelativeLayout>

# VideoView + MediaController

```
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    setContentView(R.layout.activity_main);  
    VideoView videoView = (VideoView) findViewById(R.id.videoView1);  
    videoView.setMediaController(new MediaController(this));  
    videoView.setVideoURI(Uri.parse("android.resource://" +  
        getPackageName() + "/" + R.raw.teamwork));  
    videoView.start();  
}
```







# VideoView + MediaController

---

- MediaController mặc định là ẩn, không cần đặt lên layout khi thiết kế
- MediaController dùng hàm `setAnchorView(v)` để xác định nó sẽ được gắn vào view nào khi xuất hiện, view mặc định chính là VideoView mà nó điều khiển
- MediaController được ẩn đi sau 5s nếu không có tác động của người dùng
- VideoView mặc định giữ nguyên tỉ lệ khung hình của video mà nó chạy; muốn thiết lập tỉ lệ khác, chỉ cần điều chỉnh kích thước của VideoView



# MediaPlayer + SurfaceView

---

- MediaPlayer là bộ giải mã, luôn chạy ngầm, hỗ trợ nhiều chuẩn và giao thức video, audio, mạng
- SurfaceView là view được thiết kế với mục đích để thể hiện các hình ảnh cần có tốc độ cập nhật cao, đặc biệt thích hợp với việc thể hiện dữ liệu từ video, camera và hoạt hình
- SurfaceView có những hạn chế nhất định:
  - Thread có cơ chế cập nhật thẳng vào SurfaceView không cần qua đồng bộ
  - SurfaceView luôn chiếm một vùng màn hình và không thể bị che hoặc có bóng mờ



# MediaPlayer + SurfaceView

---

```
public class MainActivity extends Activity
    implements SurfaceHolder.Callback, OnPreparedListener {

    private MediaPlayer mediaPlayer;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        SurfaceView vidSurface = new SurfaceView(this);
        vidSurface.getHolder().addCallback(this);
        setContentView(vidSurface);
    }
    public void surfaceChanged(SurfaceHolder s, int a, int b, int c) {
    }
    public void surfaceDestroyed(SurfaceHolder arg0) {
    }
}
```



# MediaPlayer + SurfaceView

---

```
public void surfaceCreated(SurfaceHolder arg0) {  
    try {  
        mediaPlayer = new MediaPlayer();  
        mediaPlayer.setDisplay(arg0);  
        mediaPlayer.setDataSource(this,  
            Uri.parse("android.resource://" +  
                getPackageName() + "/" + R.raw.teamwork));  
        mediaPlayer.prepare();  
        mediaPlayer.setOnPreparedListener(this);  
        mediaPlayer.setAudioStreamType(AudioManager.STREAM_MUSIC);  
    } catch (Exception e) { }  
}
```

```
public void onPrepared(MediaPlayer mp) { mediaPlayer.start(); }  
}
```



Phần 1.2

# Text to speech



# Text to speech

---

- Cho phép chuyển đổi từ text sang âm thanh
- Cần tìm hiểu kỹ về chọn ngôn ngữ, loại giọng, tùy biến ngữ điệu
- Có thể lựa chọn engine khác với engine mặc định
- Có thể điều chỉnh nguồn phát ra tai nghe, alarm,...
- Hai class hữu ích:
  - TextToSpeech: phát âm
  - TextToSpeechService: customize engine



# Text to speech

---

```
// gọi activity mặc định để kiểm tra có dữ liệu cho TTS chưa
```

```
Intent intent = new Intent(Engine.ACTION_CHECK_TTS_DATA);  
startActivityForResult(intent, CODE);
```

```
// xử lý dữ liệu do activity trả về
```

```
protected void onActivityResult(int req, int result, Intent data) {  
    if (req == CODE)  
        if (result != Engine.CHECK_VOICE_DATA_PASS)  
            startActivity(new Intent(Engine.ACTION_INSTALL_TTS_DATA));  
        else {  
            // dữ liệu đã có, tạo đối tượng TTS mặc định  
        }  
}
```



# Text to speech

---

```
TextToSpeech talker = new TextToSpeech(this, new OnInitListener() {  
    public void onInit(int status) {  
        talker.speak("Hi! There!", TextToSpeech.QUEUE_FLUSH, null);  
    }  
})
```

```
TextToSpeech tts = new TextToSpeech(this, new OnInitListener() {  
    public void onInit(int status) {  
        if (status != TextToSpeech.SUCCESS) return;  
        tts.setLanguage(Locale.ENGLISH);  
        tts.setPitch(0.8f);  
        tts.setSpeechRate(1.0f);  
    }  
});
```



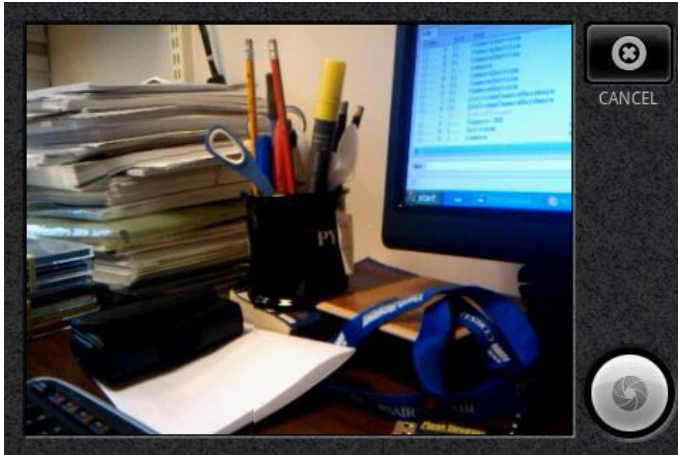


Phần 1.3

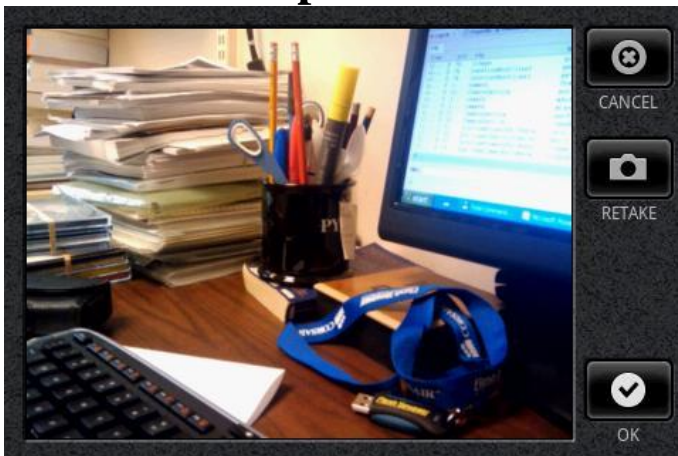
# Camera

# Chụp ảnh bằng camera activity

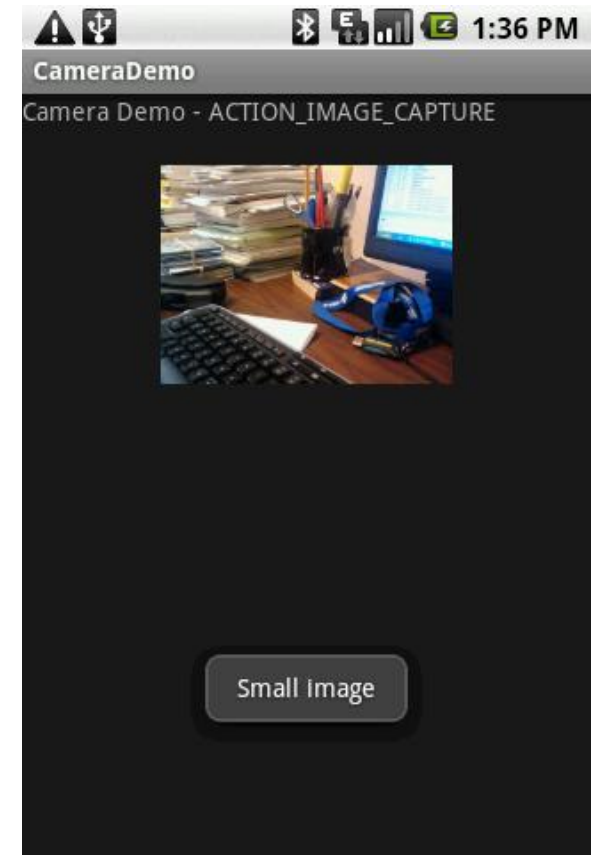
## 1. Previewing



## 2. After shutter is pressed



*These views are Provided by the Built-in ACTION\_IMAGE\_CAPTURE Intent.*



**3. Image transferred from CAMERA to the application**



# Chụp ảnh bằng camera activity

---

```
public class CameraDemo extends Activity {
    ImageView mImageView;
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        mImageView = (ImageView) findViewById(R.id.mImageView);
        Intent mIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
        startActivityForResult(mIntent, 101);
    }
    private void showToast(Context context, String text) {
        Toast.makeText(context, text, 1).show();
    }
}
```



# Chụp ảnh bằng camera activity

---

```
protected void onActivityResult(int req, int res, Intent intent)
{
    super.onActivityResult(req, res, intent);
    if (res == RESULT_CANCELED) return;
    if (req == 101) {
        Bundle b = intent.getExtras();
        Bitmap bm = (Bitmap) b.get("data");
        mImageView.setImageBitmap(bm);
    }
}
}
```



# Camera

---

- Lớp Camera hỗ trợ kết nối và ngắt kết nối tới dịch vụ camera, để cho phép bạn có thể: chụp, quay và sử dụng các tiện ích camera cung cấp
- Dùng hàm `open()` để lấy về một đối tượng Camera
- Cần khai báo trong Android Manifest để cấp quyền và các tính năng sử dụng Camera
- Xem demo ứng dụng chụp ảnh để hiểu rõ hơn

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

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

```
<uses-feature android:name="android.hardware.camera.autofocus" />
```



# The takePicture Method

---

```
public final void takePicture ( Camera.ShutterCallback shutter,  
                               Camera.PictureCallback raw,  
                               Camera.PictureCallback jpeg )
```

Triggers an asynchronous image capture. The camera service will initiate a series of *callbacks* to the application as the image capture progresses.

1. The *shutter callback* occurs after *the image is captured*. This can be used to trigger a sound to let the user know that image has been captured.
2. The *raw callback* occurs when *the raw image data is available* (NOTE: the data may be null if the hardware does not have enough memory to make a copy).
3. The *jpeg callback* occurs when the *compressed image is available*. If the application does not need a particular callback, a null can be passed instead of a callback method.

## Parameters

*shutter* callback after the image is captured, may be null

*raw* callback with raw image data, may be null

*jpeg* callback with jpeg image data, may be null



Phần 2.1

# Global Positioning Services



# Global Positioning Services

---

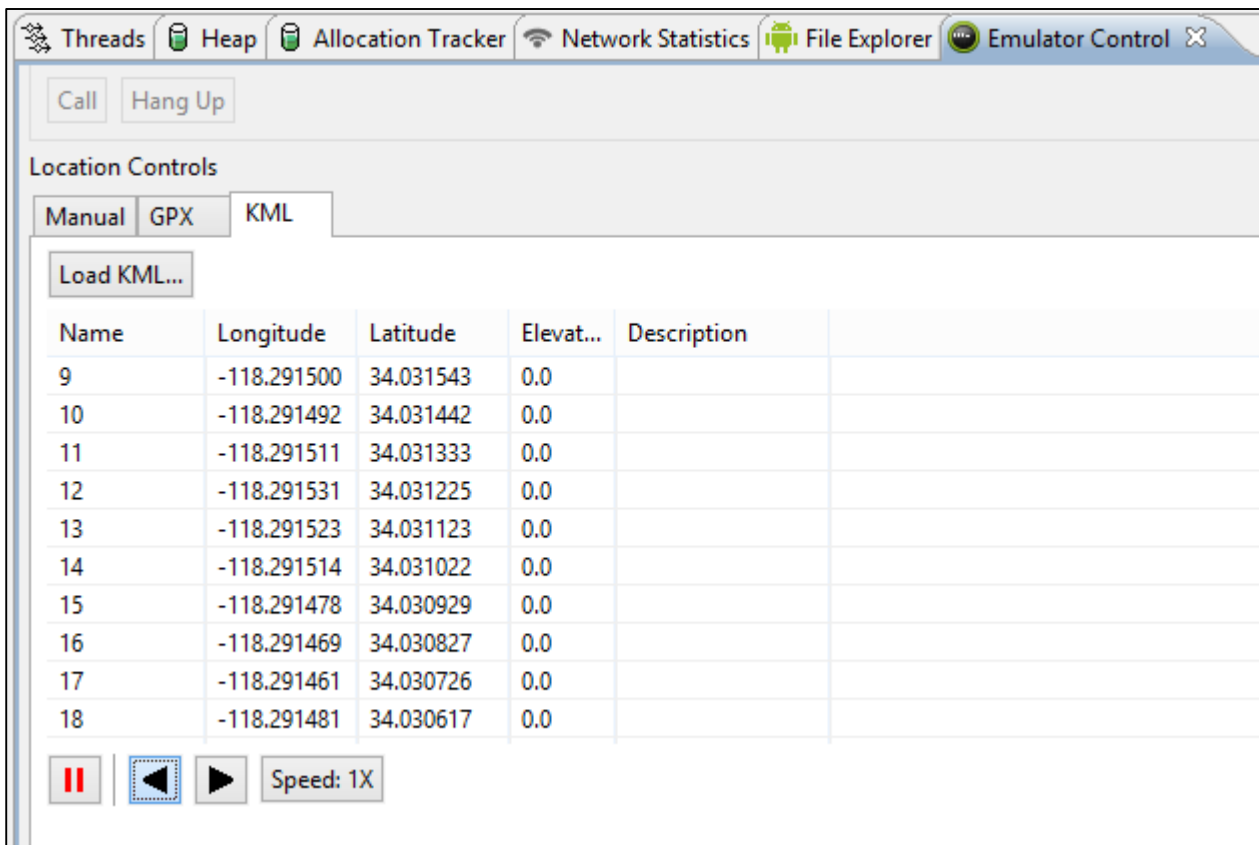
- Các bước để xác định vị trí của thiết bị:
  - Cấp quyền truy cập các dịch vụ phù hợp (trong [AndroidManifest.xml](#))
  - Khởi tạo đối tượng LocationManager bằng việc gọi phương thức `getSystemService` với tham số truyền vào là `LOCATION_SERVICE`
  - Lựa chọn một provider bằng cách gọi phương thức `getAllProviders()` hoặc `getBestProvider()`
  - Implement interface `LocationListener`, viết các hàm xử lý phù hợp với các sự kiện location liên quan
  - Gọi phương thức `requestLocationUpdates()` với provider được chọn ở bước trên



# Global Positioning Services

```
<uses-feature android:name="android.hardware.Location" />
```

```
<uses-feature android:name="android.hardware.Location.gps" />
```

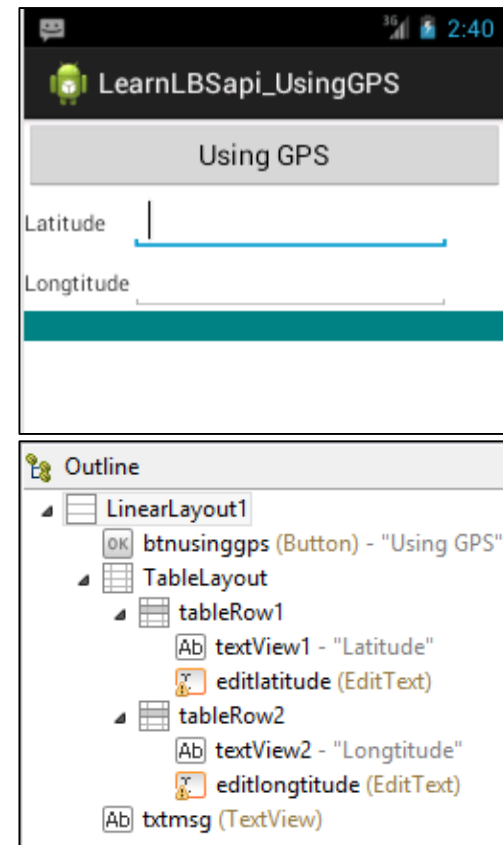


Location Controls

Manual GPX KML

Load KML...

Name	Longitude	Latitude	Elevat...	Description
9	-118.291500	34.031543	0.0	
10	-118.291492	34.031442	0.0	
11	-118.291511	34.031333	0.0	
12	-118.291531	34.031225	0.0	
13	-118.291523	34.031123	0.0	
14	-118.291514	34.031022	0.0	
15	-118.291478	34.030929	0.0	
16	-118.291469	34.030827	0.0	
17	-118.291461	34.030726	0.0	
18	-118.291481	34.030617	0.0	



LearnLBSapi\_UsingGPS

Using GPS

Latitude

Longitude

Outline

- LinearLayout1
  - btnusinggps (Button) - "Using GPS"
  - TableLayout
    - tableRow1
      - textView1 - "Latitude"
      - editlatitude (EditText)
    - tableRow2
      - textView2 - "Longitude"
      - editlongitude (EditText)
  - txtmsg (TextView)



# AndroidManifest.xml

---

```
<uses-permission android:name=  
    "android.permission.ACCESS_GPS" />  
<uses-permission android:name=  
    "android.permission.ACCESS_LOCATION" />  
<uses-permission android:name=  
    "android.permission.ACCESS_FINE_LOCATION" />  
<uses-permission android:name=  
    "android.permission.ACCESS_COARSE_LOCATION" />  
<uses-permission android:name=  
    "android.permission.INTERNET" />  
<uses-feature android:name=  
    "android.hardware.Location" />  
<uses-feature android:name=  
    "android.hardware.Location.gps" />
```



# Location services: example

---

```
public class MainActivity extends Activity
    implements LocationListener {
    Button btnusinggps;
    EditText editlatitude,editlongitude;
    TextView txtmsg;
    LocationManager locationManager=null;
    Location lastlocation=null;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btnusinggps=(Button) findViewById(R.id.btnusinggps);
        btnusinggps.setOnClickListener(new View.OnClickListener() {
            public void onClick(View arg0) {
                doUsingGPS();
            }
        });
        editlatitude=(EditText) findViewById(R.id.editlatitude);
        editlongitude=(EditText) findViewById(R.id.editlongitude);
        txtmsg=(TextView) findViewById(R.id.txtmsg);
    }
}
```



# Location services: example

---

```
public void doUsingGPS ()
{
    try{
        locationManager =(LocationManager)
            getSystemService (LOCATION_SERVICE);
        Criteria criteria = new Criteria();
        criteria.setAccuracy(Criteria.NO_REQUIREMENT);
        criteria.setPowerRequirement (Criteria.NO_REQUIREMENT);
        String bestProvider = locationManager
            .getBestProvider(criteria, true);
        Location loc = locationManager.
            getLastKnownLocation (bestProvider);
        if(loc==null) {/**//process null*/}
        else{
            // "Requesting network location updates..."
            locationManager.requestLocationUpdates
                (bestProvider, 1000, 0, this);
        }
    }
    catch (Exception e) {e.printStackTrace();}
}
```

# Location services: example

```
public void onLocationChanged(Location location) {
    String locInfo = String.
        format("Current loc = (%f, %f) @ (%f meters up)",
            location.getLatitude(),
            location.getLongitude(),
            location.getAltitude() );
    if (lastlocation != null)
    {
        float distance = location.distanceTo(lastlocation);
        locInfo += String.
            format("\n Distance from last = %f meters",
                distance);
    }
    lastlocation = location;
    txtmsg.setText(locInfo);
}
public void onProviderDisabled(String provider) {□}
public void onProviderEnabled(String provider) {□}
public void onStatusChanged(String provider, int status, Bund
}
```



# Location services: giải thích

---

- Thông qua LocationManager có thể lấy được nhiều provider có năng lực location service (tùy vào thiết bị có những sensor này hay không)
  - `List<String> getAllProviders()`: lấy tất cả provider
  - `String getBestProviders(Criteria, enableOnly)`: lấy provider phù hợp nhất với điều kiện
    - Criteria: power, accuracy, speed, altitude
  - `LocationProvider getProvider(name)`: Lấy provider theo tên
  - `requestLocationUpdate(providerName, minTime, minDistance, listener)`: Yêu cầu gọi listener mỗi minTime mili-giây hoặc provider phát hiện dịch chuyển midDistance mét



Phần 2.2

# Geocoding Locations



# Geocoding Locations

---

- Google API cho phép người dùng có thể xác định địa chỉ và tọa độ thông qua sử dụng đối tượng Geocoder
- Sử dụng đối tượng Geocoder khá đơn giản và không yêu cầu đăng ký thêm quyền trong manifest
- Service này hoạt động dựa trên remote service của Google, vì vậy đôi khi bị trục trặc (phải liệu mà xử lý lỗi phù hợp)
- Các phương thức thường được sử dụng như:
  - `List<Address> getFromLocation`
  - `List<Address> getFromLocationName`



# Geocoding Locations - example

```
public void doGeoCodingFromLocationName(String locName)
{
    if (Geocoder.isPresent()) {
        Geocoder coder = new Geocoder(this);
        try {
            List<Address> addresses = coder
                .getFromLocationName(locName, 3);
            if (addresses != null) {
                StringBuilder locInfo = new StringBuilder("Results:\n");
                double lat = 0f;
                double lon = 0f;
                for (Address namedLoc : addresses) {
                    lat = namedLoc.getLatitude();
                    lon = namedLoc.getLongitude();
                    locInfo.append("Location: ").append(lat)
                        .append(", ").append(lon).append("\n");
                }
                //process locInfo here
            }
        } catch (IOException e) {
            Log.e("GPS", "Failed to get address", e);
        }
    } else { /*No geocoding available*/ }
}
```



# Geocoding Locations - example

---

```
public void doGeoCodingFromLocation()
{
    //You should get location from onLocationChanged()
    Location location=new Location("Fake location");
    location.setLatitude(-113.115);
    location.setLongitude(100.114);
    if (Geocoder.isPresent()) {
        Geocoder coder = new Geocoder(this);
        try {
            List<Address> addresses = coder.getFromLocation(
                location.getLatitude(),
                location.getLongitude(), 3);
            if (addresses != null) {
                String locInfo="";
                for (Address namedLoc : addresses) {
                    String placeName = namedLoc.getLocality();
                    String featureName = namedLoc.getFeatureName();
                    String country = namedLoc.getCountryName();
                    String road = namedLoc.getThoroughfare();
```



# Geocoding Locations - example

---

```
        locInfo+=(String.format("[%s] [%s] [%s] [%s]\n",
                                placeName, featureName, road, country));
        int addIdx = namedLoc.getMaxAddressLineIndex();
        for (int idx = 0; idx <= addIdx; idx++) {
            String addLine = namedLoc.getAddressLine(idx);
            locInfo+=(String.format("Line %d: %s\n", idx,
                                    addLine));
        }
        Toast.makeText(this, locInfo, Toast.LENGTH_LONG).show();
    }
} catch (IOException e) {
    Log.e("GPS", "Failed to get address", e);
}
} else {
    Toast.makeText(MainActivity.this, "No geocoding available",
        Toast.LENGTH_LONG).show();
}
}
```



Phần 2.3

# Mapping Locations

# Mapping Locations

- Google API cung cấp hai cách để làm việc với Google Map
  - Hiện thị vị trí lên Google Maps bên ngoài ứng dụng thông qua Intent
  - Nhúng widget MapView vào bên trong ứng dụng





# Google Maps bên ngoài

---

```
public void doMappingIntent ()
{
    try
    {
        Location locFake=new Location("FAKE Location");
        locFake.setLatitude(-118.344);
        locFake.setLongitude(34.945);
        String geoURI = String.format("geo:%f,%f",
            locFake.getLatitude(),
            locFake.getLongitude());
        Uri geo = Uri.parse(geoURI);
        Intent geoMap = new Intent(Intent.ACTION_VIEW, geo);
        startActivity(geoMap);
    }
    catch(Exception e)
    {
        Toast.makeText(this, e.toString(),
            Toast.LENGTH_LONG).show();
    }
}
```



# MapView

---

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

```
<com.google.android.maps.MapView  
    android:id="@+id/map"  
    android:layout_width="fill_parent"  
    android:layout_height="wrap_content"  
    android:apiKey="yourMapKey" />
```

```
<application  
    android:allowBackup="true"  
    android:icon="@drawable/ic_launcher"  
    android:label="@string/app_name"  
    android:theme="@style/AppTheme" >  
    <activity<br/>  
    <uses-library android:name=  
        "com.google.android.maps" />  
</application>
```

```
MapView mapView = (MapView) findViewById(R.id.mapview);  
mapView.setBuiltInZoomControls(true);  
MapController mapController = mapView.getController();  
mapController.setZoom(10);
```



# Google Maps: giải thích

---

- V1: bắt buộc làm việc với MapActivity là một activity chứa MapView mặc định
- V2 (API 12 trở lên): cung cấp đối tượng MapView là một ViewGroup tương tự các view khác, cung cấp sẵn các năng lực như đáp ứng key presses, touch, zooming, ngoài ra:
  - Hiển thị dữ liệu vector nhanh hơn
  - Tốn ít băng thông internet hơn
  - Caching làm việc hiệu quả hơn
  - Hỗ trợ 3D map
- V3: mới có bản javascript chưa có native android





# Google Maps: giải thích

---

- Dịch vụ Google Maps nằm trong gói Google Play Services, vì vậy cần thêm thư viện này vào dự án
- Đây là dịch vụ của Google, nên cần key do Google cung cấp để truy cập dịch vụ
- Lấy Google Maps API Key:  
<https://code.google.com/apis/console>
- Trong AndroidManifest.xml, phần <application>, bổ sung thêm các dòng sau:
  1. `<meta-data android:name="com.google.android.maps.v2.API_KEY" android:value="KEY"/>`
  2. `<meta-data android:name="com.google.android.gms.version" android:value="@integer/google_play_services_version" />`



# Google Maps: giải thích

---

- Quyền (bắt buộc):

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

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

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

```
<uses-permission  
android:name="com.google.android.providers.gsf.permission.READ_GSER  
VICES"/>
```

- Quyền (optional):

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

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



# Google Maps: giải thích

---

- GoogleMap tự động xử lý:
  - Kết nối đến Google Maps service + tải dữ liệu bản đồ
  - Hiển thị bản đồ
  - Hiển thị các control điều khiển như zoom và dịch chuyển
  - Xử lý các action zoom và dịch chuyển
- Tùy biến GoogleMap bằng cách viết các listener phù hợp và đăng kí với GoogleMap :
  - `setOnMapClickListener`
  - `setOnMapLongClickListener`
  - `setOnMarkerClickListener`
  - `setOnMarkerDragListener`



# Google Maps: giải thích

---

- Từ MapView có thể lấy GoogleMap bằng method `getMap()`

```
GoogleMap mMap = ((SupportMapFragment) getSupportFragmentManager().  
findFragmentById(R.id.map)).getMap();
```

- Lập trình viên có thể tác động vào Google Map:
  - Thêm một đánh dấu (`addMarker`)  
`mMap.addMarker(new MarkerOptions().position(x).title("XXX"));`
  - Vẽ đa giác, hình tròn với vị trí theo tọa độ địa lý
  - Điều chỉnh góc nhìn
  - Điều chỉnh loại bản đồ
  - Điều chỉnh vị trí (camera)