

위치 기반 서비스 Location Based Service



목차

LBS 개요

LBS 사용 준비

- ◆ Google Play Service
- ◆ Location Permissions

기기 위치 확인

Geocoding

외부 지도 앱의 사용

위치 기반 서비스 (LBS: Location Based Service)

개요

- ◆ 위치 정보를 수신 받아 다른 정보와 결합하여 위치 특성을 반영한 정보를 제공하는 서비스

지도와의 결합

- ◆ 위치 정보를 특정 지도 위치에 표시
- ◆ 네비게이션: 위치 정보와 지도 정보를 결합

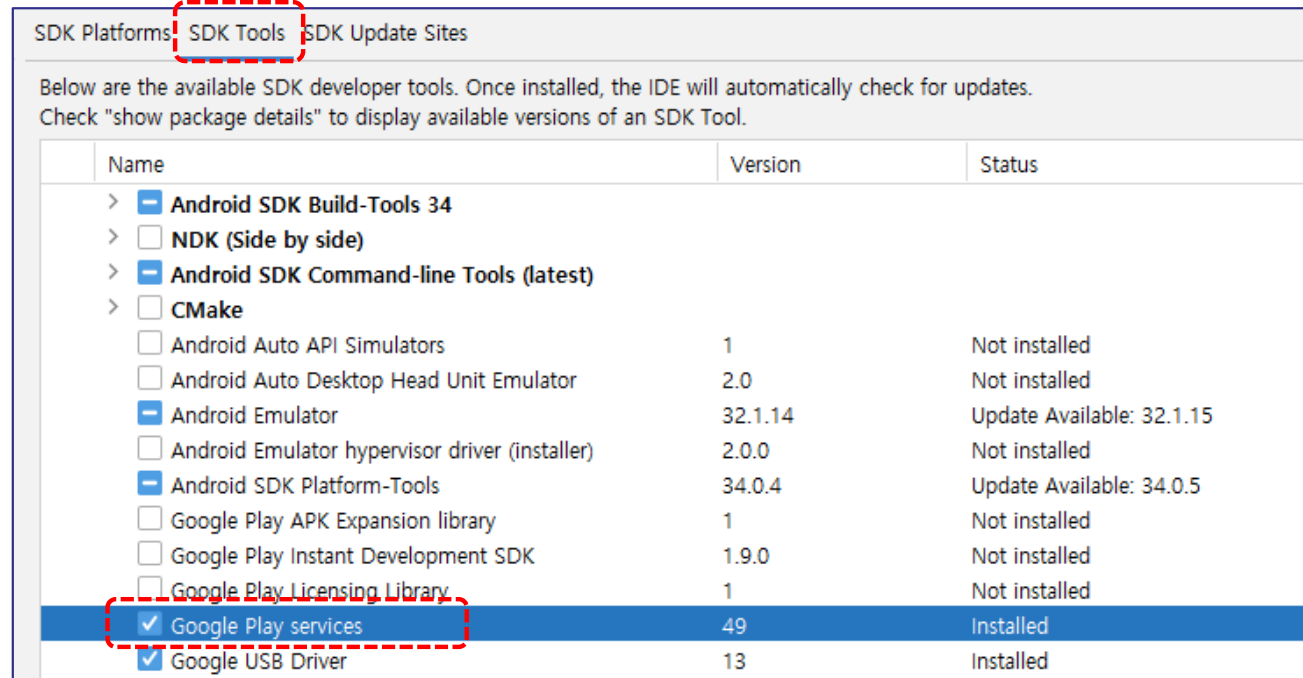
스마트폰의 위치 정보 확인 방법

- ◆ GNSS(Global Navigation Satellite System): 위성으로부터 위치 정보 수신 (GPS, Galileo, GLONASS, Beidou 등)
- ◆ 네트워크: 모바일통신이나 Wi-Fi 연결 정보를 토대로 위치 파악
- ◆ 그 밖에 Beacon, RFID 또는 Bluetooth 를 사용한 초근거리 위치 확인 기술 등이 존재

LBS 사용 준비 01 – Google Play Service 설정

Google Play Services 설치 확인

◆ [Tools] → [SDK Manager] → [SDK Tools] 탭



Dependency 추가 – build.gradle (Module: ...)

사용 가능
버전 확인

```
implementation 'com.google.android.gms:play-services-location:21.0.1'
```

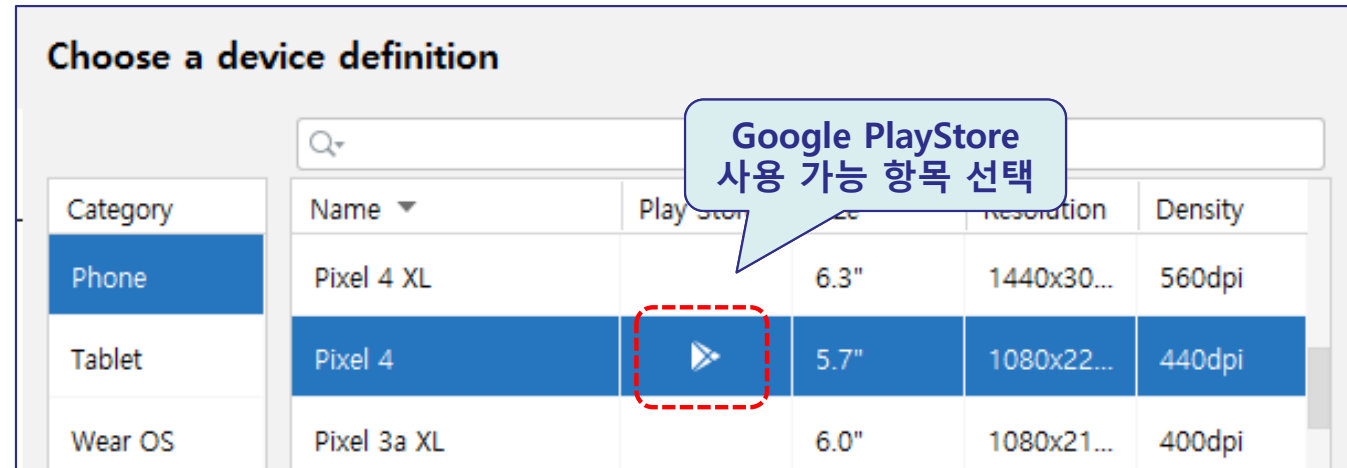
Import 시 시스템 클래스 또는 GooglePlay 제공 클래스 확인

```
import com.google.android.gms.location.*
```

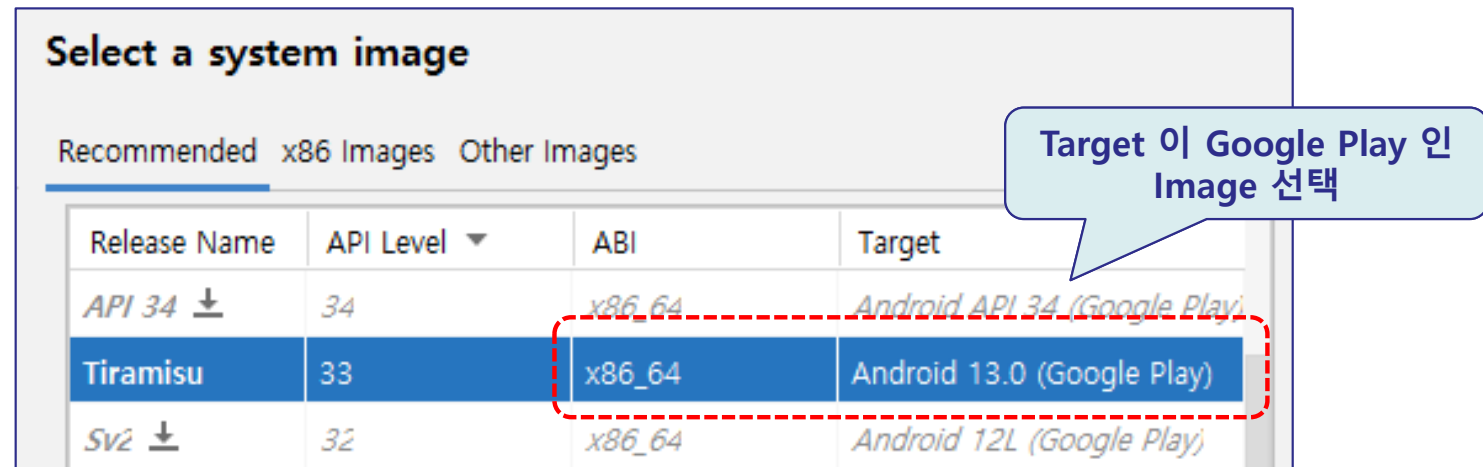
LBS 사용 준비 02 - Android 기기 준비

Android Emulator 생성 시 (PlayStore 사용 가능 버전)

◆ 기기 선택



◆ System Image 선택



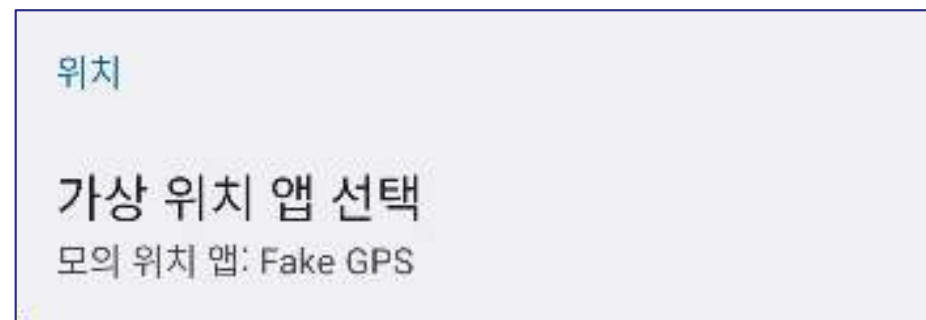
LBS 사용 준비 03

MockUp GPS 앱 설치

- ◆ Google Plays Store 에서 가상의 GPS 정보를 생성하는 Mockup GPS 앱 선택

Mockup GPS 정보 지정

- ◆ [개발자 옵션] 활성화
 - [설정] → [기기 정보] → [빌드 번호] 7번 클릭
- ◆ MockGPS 사용 설정
 - [설정] → [시스템] → [개발자 옵션] → [가상 위치 앱 선택]
 - 설치한 Mockup GPS 앱 지정



LBS 사용 준비 02 – Permission

Permission 추가

- ◆ 위치 관련 Permission 추가
- ◆ 사용하고자 하는 위치 정확도에 따라 선택

필요 Permission

- ◆ ACCESS_COARSE_LOCATION (거리 블록 정도의 정확도 수준)
- ◆ ACCESS_FINE_LOCATION (정확한 위치 정확도)

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

LBS 사용 준비 03 – Permission 실시간 요청

Permission 실시간 요청

- ◆ 중요한 Permission의 경우 실시간 확인 → Marshmallow 버전 이후
- ◆ 사용자는 앱 실행 중 권한 요구를 선택 또는 취소하고 언제든지 변경할 수 있음

실시간 Permission 확인 절차

1. `checkSelfPermission()` 으로 필요 권한 확인
2. `ActivityResultContracts.RequestMultiplePermission()` 또는 `.RequestPermission()` 으로 필요 권한 요청
3. 사용자의 응답에 따라 앱의 실행 여부 결정

권한 확인의 예

◆ checkSelfPermission() 과 requestPermissions()

```
fun checkPermissions () {  
    if (checkSelfPermission(ACCESS_FINE_LOCATION)  
        == PackageManager.PERMISSION_GRANTED  
        && checkSelfPermission(ACCESS_COARSE_LOCATION)  
            == PackageManager.PERMISSION_GRANTED) {  
        showData("Permissions are already granted")  
        // textView에 출력  
    } else {  
        locationPermissionRequest.launch(arrayOf(  
            ACCESS_FINE_LOCATION,  
            ACCESS_COARSE_LOCATION))  
    }  
}
```

checkSelfPermission (권한) 결과가
PackageManager.PERMISSION_GRANTED
또는 .PERMISSION_DENIED 인지 확인

권한 요청 작업을 실행하는
ActivityResultLauncher 객체
→ registerActivityResult() 가 생성

LBS 사용 준비 05 – Permission 실시간 요청

권한요청의 사용자 응답 처리

Activity 결과 요청 등록 →
ActivityResultLauncher 생성

ActivityResultContract
: 요청하는 Activity 정보 지정

```
val locationPermissionRequest
= registerForActivityResult( ActivityResultContracts.RequestMultiplePermissions() ) {
    permissions ->
        when {
            permissions.getDefault(ACCESS_FINE_LOCATION, false) -> {
                showData("FINE_LOCATION is granted")
            }
            permissions.getDefault(ACCESS_COARSE_LOCATION, false) -> {
                showData("COARSE_LOCATION is granted")
            }
            else -> {
                showData("Location permissions are required")
            }
        }
    }
}
```

Default : false

사용자 응답 처리



기기 위치 확인 01

FusedLocationProviderClient

◆ Google Play Services 에서 제공하는 위치확인 관련 클래스

- 시스템 제공클래스인 LocationManager 역할 대체
- 최종 위치 확인
- 현재 위치 확인

객체 생성

```
private lateinit var fusedLocationClient : FusedLocationProviderClient
```

Context

```
fusedLocationClient = LocationServices.getFusedLocationProviderClient(this)
```

기기 위치 확인 02

위치 확인 수행 준비

- ◆ Permission 확인 및 획득 수행
- ◆ LocationRequest 구현
 - 위치정보 수신 조건 지정
- ◆ LocationCallback 구현
 - 시스템이 전달해주는 위치 정보를 수신
- ◆ FusedLocationProviderClient 객체 생성
 - 위치 확인을 수행

위치 확인 수행

- ◆ FusedLocationProviderClient 클래스의 requestLocationUpdates() 사용

기기 위치 확인 03

LocationRequest 구현

- ◆ 위치정보 수신을 위한 조건 지정 : Builder(ms) 로 생성
- ◆ setIntervalMillis(ms)
 - 위치정보 업데이트 간격을 ms 로 지정
- ◆ setMinUpdateIntervalMillis(ms)
 - 위치정보 업데이트의 최소간격 지정
- ◆ setPriority(): 요청 우선순위 및 source 지정
 - PRIORITY_HIGH_ACCURACY : GPS
 - PRIORITY_BALANCED_POWER_ACCURACY : Wi-Fi 또는 Cell network

수신간격 ms

```
val locRequest = LocationRequest.Builder(10000)
    .setMinUpdateIntervalMillis(5000)
    .setPriority(Priority.PRIORITY_BALANCED_POWER_ACCURACY)
    .build()
```

기기 위치 확인 04

LocationCallback 구현

- ◆ 위치정보의 수신 결과를 전달 받는 클래스
- ◆ `onLocationResult(LocationResult locationResult)`를 재정의하여 위치정보 수신 시 수행할 동작 지정
- ◆ `requestLocationUpdates()`에 전달

```
val locCallback : LocationCallback
    = object : LocationCallback() {
        override fun onLocationResult(locResult: LocationResult) {
            val currentLoc : Location = locResult.locations[0]
            showData("위도: ${currentLoc.latitude}, 경도: ${currentLoc.longitude}")
        }
    }
```

위치정보들을
보관하는 클래스

Location
: 위치정보를 표현

기기 위치 확인 05

☞ 위치정보 수신 실행 사전 확인

- ◆ 위치 제공자가 사용가능한지 확인: network 와 GPS
- ◆ 위치 관련 Permission 설정 확인
- ◆ 위치 요청관련 설정 정보 확인

☞ 위치 확인 실행

- ◆ requestLocationUpdates() 실행

FusedLocationProviderClient 객체

```
private fun startLocUpdates() {
    fusedLocationClient.requestLocationUpdates(
        locRequest,          // LocationRequest 객체
        locCallback,         // LocationCallback 객체
        Looper.getMainLooper() // System 메시지 수신 Looper
    )
}
```

시스템의 메시지를
전달받는 Looper 객체

기기 위치 확인 06

📄 위치 확인 종료

- ◆ removeLocationUpdates() 실행

FusedLocationProviderClient 객체

LocationCallback 객체

fusedLocationClient.removeLocationUpdates(locCallback)

📄 사용자에게 의해 종료가 안 될 경우 대비

- ◆ Activity 의 생명주기 함수인 onPause()에서 호출

```
override fun onPause() {
    super.onPause()
    fusedLocationClient.removeLocationUpdates(locCallback)
}
```


기기 위치 확인 07

최종 위치의 사용

- ◆ 기기가 마지막으로 확인한 위치를 확인
- ◆ 앱 실행 시 위치정보를 수신하지 못 할 경우 기본 위치 또는 마지막 수신위치로 위치 표시

최종 위치 확인

- ◆ 각 Listener 는 Task<TResult> 객체 반환

```
private fun getLastLocation() {
    fusedLocationClient.lastLocation.addOnSuccessListener { location: Location? ->
        if (location != null) {
            showData(location.toString())
        }
    }
    fusedLocationClient.lastLocation.addOnFailureListener { e: Exception ->
        Log.d(TAG, e.toString())
    }
}
```

getLastLocation() : Task<TResult>

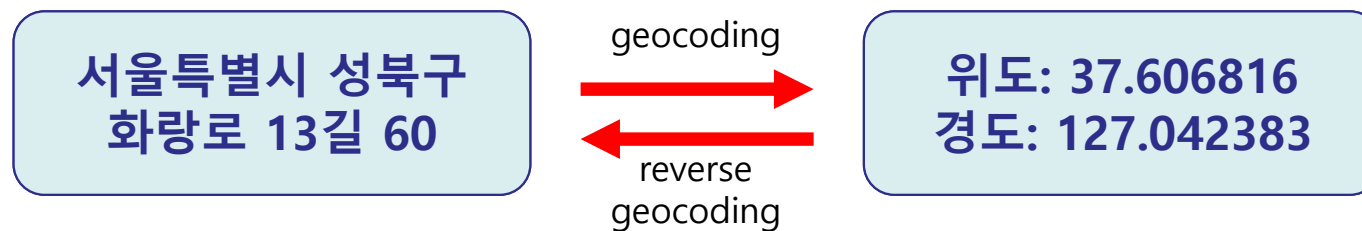
최종 위치 확인 성공 시

최종 위치 확인 실패 시

Geocoding 01

Geocoding

- ◆ Geocode: 일반 주소를 위도(latitude)/경도(longitude) 로 변환
- ◆ Reverse geocode: 위도/경도를 일반 주소로 변환



Geocoder 확인

- ◆ 안드로이드 기본 프레임워크가 아닌 Backend 서비스에 포함되어 있으므로 사용가능 가부의 확인 필요
- ◆ Geocoder.isPresent() 로 확인 → 위치 권한 필요

Geocoding 02

Geocoder 의 생성

```
private lateinit var geocoder : Geocoder
```

기기 지정
지역

```
geocoder = Geocoder(this, Locale.getDefault())
```

- ◆Geocoder는 네트워크를 통해 정보를 획득하며, 별도의 실행흐름에서 수행 필요

Geocoding

- ◆Geocoder.getLocation(위도, 경도, 개수, Geocoder.GeocoderListener)

위도

경도

결과 개수

GeocoderListener
람다 함수

```
geocoder.getLocation(37.505816, 127.042383, 5) { addresses ->
    CoroutineScope(Dispatchers.Main).launch {
        showData(addresses.get(0).getAddressLine(0).toString())
    }
}
```

첫 번째 Address 객체의 주소 확인

Geocoding 03

Reverse Geocoding

- ◆ `getFromLocationName(주소명, 개수, Geocoder.GeocoderListener)`

```
geocoder.getFromLocationName("동덕여자대학교", 5 ) { addresses ->
    CoroutineScope(Dispatchers.Main).Launch {
        showData("위도: ${addresses.get(0).Latitude}, " +
            "경도: ${addresses.get(0).Longitude}")
    }
}
```

첫 번째 Address 객체의 위도/경도


지도 앱 사용하기

안드로이드 기본 지도앱 사용

```
fun callExternalMap() {
    val locLatLng // 위도/경도 정보로 지도 요청 시
        = String.format("geo:%f,%f?z=%d", 37.606320, 127.041808, 17)
    val locName // 위치명으로 지도 요청 시
        = "https://www.google.co.kr/maps/place/" + "Hawolmok-dong"
    val route // 출발-도착 정보 요청 시
        = String.format("https://www.google.co.kr/maps?saddr=%f,%f&daddr=%f,%f",
            37.606320, 127.041808, 37.601925, 127.041530)
    val uri = Uri.parse(locLatLng)
    val intent = Intent(Intent.ACTION_VIEW, uri)
    startActivity(intent)
}
```

 GPS 에서 전달하는 현재 위치의 위도, 경도를 읽어와 해당 위치의 실주소를 출력하는 앱을 구현하시오.

- ◆ GPS 사용
- ◆ 위치가 바뀔 때마다 현재의 위도/경도 확인 후 이를 Geocoding 을 통해 주소로 바꾸어 출력

 앱을 종료 후 다시 실행시키면 가장 마지막에 수신한 위치 정보를 실주소로 출력하도록 위의 앱을 수정하시오.

참고

위치 정보 처리

- ◆ <https://developer.android.com/training/location?hl=ko>

위치 정보 접근 권한 요청

- ◆ <https://developer.android.com/training/location/permissions?hl=ko>

Activity에서 결과 가져오기

- ◆ <https://developer.android.com/training/basics/intents/result?hl=ko>

구글맵 스키마 관련 정보

- ◆ <https://developers.google.com/maps/documentation/urls/android-intents?hl=ko>