

컬리 사전인터뷰



Kotlin을 사용한 Android 앱 개발에 지도 사용 필요.
구글 / 네이버 / 카카오 세가지 비교.
핀/마커 커스텀 가능 여부와 범위 확인 필요



좋아요

아래는 **Kotlin 기반 Android 앱**에서 사용할 수 있는
Google Maps / Naver Map / Kakao Map SDK 3종을 비교한 정리입니다.

Android 지도 SDK 비교 (2025 기준)

항목	Google Maps SDK	Naver Map SDK	Kakao Map SDK
SDK 추 가 방식	Play Services (<code>com.google.android.gms:play- services-maps</code>)	<code>com.naver.maps:map- sdk</code>	<code>libs</code> 또는 <code>mavenCentral()</code> (SDK Key 필요)
지도 사용 범위	전세계 지원 (글로벌 지도)	한국 중심 + 일부 해외 (제한적)	한국 중심, 해외 미지원
핀/마커 커스텀	완전 커스텀 가능 (Bitmap, Vector, Layout 기반)	완전 커스텀 가능 (View나 Drawable로 가 능)	제한적 (일부 속성만 변경 가능, 완전 커스텀은 비권장)
클러스터 링 (여러 마커 그룹 화)	지원 (<code>MarkerCluster</code> or 3rd- party libs)	지원 (직접 구현 용이)	직접 구현 필요
지도 스타 일 변경 (색상, 다 크모드 등)	JSON 스타일 지원	Light/Dark Theme 및 Custom Map 지원	제한적 (일부 색상만 변경 가능)
지도 이 동/줌 제 어	자유롭게 API 제어 가능	완전 제어 가능	가능하지만 제스처 세밀제어는 제한 적



항목	Google Maps SDK	Naver Map SDK	Kakao Map SDK
오프라인 지도 지원	✗ (온라인 필요)	✗ (온라인 필요)	⚠ 일부 캐시 지원
위치 추적 / 현재 위치 표시	✓ 쉽게 가능 (<code>MyLocationLayer</code>)	✓ 쉽게 가능 (<code>LocationOverlay</code>)	✓ 가능 (<code>CurrentLocationTrackingMode</code>)
사용 편의성 (Kotlin)	★★★★★ (공식 문서 완벽)	★★★★★ (Kotlin-friendly DSL)	★★★ (Java 중심 문법)
라이선스 / 정책	무료(일정 요청량 내), 이후 과금	무료(네이버 계정/콘솔 등록 필요)	무료(비상업용), 상업용은 별도 계약 필요
상용 서비스 적합성	✓ 전세계 앱에서 표준	✓ 한국 시장에 강점	⚠ 국내 한정, 상용 시 제약 많음
지도 로딩 속도 (한국 내)	★★★	★★★★★	★★★★★

📍 마커(핀) 커스텀 가능 여부 상세

Google Maps	Naver Map	Kakao Map
✓ (<code>MarkerOptions.icon(BitmapDescriptor)</code>)	✓ (<code>Marker.icon = OverlayImage.fromResource(...)</code>)	✓ (<code>MapPOIItem.markerType</code>)
✓ (<code>Marker 대신 GroundOverlay 나 CustomView</code>)	✓ (<code>OverlayImage.fromView(view)</code>)	⚠ 제한 (뷰 기반 불가)
✓ (<code>setOnMarkerClickListener</code>)	✓ (<code>setOnClickListener</code>)	✓ (<code>setPOIItemEventListener</code>)



Google Maps

Naver Map

Kakao Map

✓ (marker.position = newLatLng)

✓ (marker.position = ...)

⚠ 성능 저하 위험

🔍 선택 가이드

목적 / 상황	추천 지도
글로벌 서비스용 앱	 Google Maps SDK
한국 중심 서비스, 네이버 스타일 UI 원활 때	 Naver Map SDK
국내 내부용 앱, 단순 지도 표시만 필요	 Kakao Map SDK
핀/마커 완전 커스텀 UI 필요	 Naver Map SDK
사용자 위치 추적/경로/교통정보	 Google Maps SDK
비즈니스 목적 + 무료 유지 원함 (한국 한정)	 Naver Map SDK

💡 요약 결론

순위	SDK	이유
🥇	Naver Map SDK	Kotlin 친화적, 마커 커스텀 자유도 높고, 한국 지리 정밀도 우수
🥈	Google Maps SDK	글로벌 지도 및 다양한 기능, 그러나 API 과금 주의
🥉	Kakao Map SDK	단순 표시용엔 충분하지만, 개발 유연성 낮음



원하신다면 제가 세 지도 SDK 각각의 마커 커스텀 예제 (**Kotlin 코드**) 도 바로 보여드릴 수 있습니다.
예를 들어 👉

“Google Map / Naver Map / Kakao Map 각각에서 커스텀 마커를 표시하는 코드 예제”

이렇게 드릴까요?



Compose 적용도 모두 가능한지,
가능하다면 Naver map 사용 가이드 및 링크 필요



네 — Naver 지도 SDK는 Jetpack Compose 환경에서도 사용이 가능하고, 커스텀 래퍼 라이브러리도 존재합니다.

아래에 **Compose 적용 가능 여부 + Naver Map SDK 사용 가이드 및 링크**를 정리해 드릴게요.

✓ Compose 적용 가능 여부

- 공식적으로 Naver Map SDK 자체 문서가 Compose 전용 API를 완전히 제공한다고 명시하지 않았지만, 커뮤니티 기반 “Compose 래퍼 라이브러리”가 존재합니다. 예컨대 naver-map-compose 라이브러리가 그 예입니다. [GitHub](#) ±2
- 라이브러리 설치 후 `@Composable` 함수 안에서 지도를 직접 그릴 수 있으며, Compose UI 흐름 안에서 지도를 제어하는 예제들이 존재합니다. [디션의 개발 이모...](#) ±1
- 따라서 **Compose 환경(Android + Kotlin + Compose UI)**에서 Naver 지도 사용은 충분히 가능하다고 보시면 됩니다.

🧭 Naver Map SDK 사용 가이드 및 링크

공식 문서

- “시작하기” 가이드 (한국어):
[시작하기 · 네이버 지도 안드로이드 SDK ↗](#) — 저장소 설정, 클라이언트 ID 등록, SDK 의존성 추가 등 기본 내용 포함 [Naver Maps](#)
- 공식 영문 문서도 사용 가능: [Get Started · NAVER Map Android SDK ↗](#) [Naver Maps](#) ±1

Compose 래퍼 라이브러리

- GitHub 저장소: [fornewid/naver-map-compose ↗](#) — Compose 전용 래퍼 라이브러리로 설치 및 사용 설명이 나와 있음. [GitHub](#)



- Maven Repository: `io.github.fornewid:naver-map-compose:<version>` 형태로依存성 추가 가능.
[Maven Repository](#) +1
- 최신 버전 확인 가능: 예컨대 version 1.7.3 등이 존재함. [Maven Repository](#)

간단한 사용 흐름 (Compose)

아래는 Compose 환경에서 Naver 지도를 사용하는 기본 흐름 예시입니다:

```
@Composable
fun NaverMapScreen() {
    // 예: Modifier, MapProperties 등을 설정
    NaverMap(
        modifier = Modifier.fillMaxSize()
    ) {
        // 마커 추가, 이벤트 처리 등
    }
}
```

또한, 설정에 따라 MapProperties 등의 매개변수를 `remember { mutableStateOf(...) }` 형태로 받아서 줌 범위, 카메라 위치, 지도 타입 등을 제어할 수 있음. [디션의 개발 이모...](#) +1

의존성 추가 및 저장소 설정

- 루트 프로젝트(예: `settings.gradle.kts` 또는 `build.gradle`)에 Maven 저장소 추가해야 함:

```
dependencyResolutionManagement {
    repositories {
        google()
        mavenCentral()
        maven("https://repository.map.naver.com/archive/maven")
    }
}
```

[Naver Maps](#) +1

- 앱 모듈 `build.gradle.kts`에 기본 SDK 및 Compose 라퍼 추가:

```
implementation("com.naver.maps:map-sdk:<latest-version>")
implementation("io.github.fornewid:naver-map-compose:<latest-version>")
```

[Naver Maps](#) +1

- 클라이언트 ID 설정: `AndroidManifest.xml`에 `<meta-data`

`android:name="com.naver.maps.map.NCP_KEY_ID" android:value="YOUR_KEY"/>` 형태로 등록.
[Naver Maps](#)



필요하시면 **Compose** 환경에서 마커(핀) 커스텀 하는 예제 코드도 작성해 드릴까요? 그렇게 하면 귀하의 프로젝트(Android + Kotlin + Compose)에도 바로 적용하실 수 있어요.



우선 xml 뷰스타일로 naver지도 사용시 아래 기능 내용 정리해줘
1. 내위치 표시
2. 지도 이동 및 원근 설정
3. 마커 커스텀(텍스트입력 희망)
4. 정보창 커스텀
5. 폴리라인 (내가 지나간 길 표시 희망)



좋습니다 😊

아래는 **Kotlin + XML 기반 Naver Map SDK** 사용 시

요청하신 기능들 — **1. 내 위치 표시, 2. 지도 이동/줌, 3. 마커 커스텀(텍스트 포함), 4. 정보창 커스텀, 5.**

폴리라인 표시 — 의 구현 방법과 핵심 코드 요약입니다.

🧭 1 내 위치 표시 (현재 위치 아이콘)



Naver Map SDK는 `LocationOverlay` 객체로 사용자의 현재 위치를 지도 위에 표시할 수 있습니다.

💡 구현 요약

```
val locationOverlay = naverMap.locationOverlay
locationOverlay.isVisible = true
locationOverlay.position = LatLng(latitude, longitude)
```

- `isVisible = true` → 내 위치 아이콘 표시
- `position` → GPS 등으로 받은 현재 위치 지정
- 현재 위치 추적을 자동화하려면 `FusedLocationSource` 사용

예시 (Activity/Fragment)

```
private lateinit var locationSource: FusedLocationSource

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = ActivityMainBinding.inflate(layoutInflater)
```



```
setContentView(binding.root)

locationSource = FusedLocationSource(this, LOCATION_PERMISSION_REQUEST_CODE)

binding.mapView.getMapAsync { naverMap ->
    naverMap.locationSource = locationSource
    naverMap.locationTrackingMode = LocationTrackingMode.Follow
}
}
```

2 지도 이동 및 원근(줌/틸트/회전) 설정

지도 제어 API

```
val cameraUpdate = CameraUpdate.scrollTo(LatLng(37.5665, 126.9780)) // 서울 좌표
naverMap.moveCamera(cameraUpdate)
```

추가 옵션

```
val cameraUpdate = CameraUpdate.scrollTo(LatLng(37.5665, 126.9780), 15.0)
    .animate(CameraAnimation.Fly, 1000L)
naverMap.moveCamera(cameraUpdate)
```

- `CameraAnimation.Fly` → 부드럽게 이동
- `zoomTo()` / `tiltTo()` / `rotateTo()` 조합으로 원근감 조절 가능

3 마커 커스텀 (텍스트 입력 포함)

기본 마커

```
val marker = Marker()
marker.position = LatLng(37.5665, 126.9780)
marker.map = naverMap
```

커스텀 마커 (텍스트 포함)

- 마커 아이콘을 직접 뷰(View)로 만들어서 이미지로 변환 가능

```
fun createCustomMarker(context: Context, text: String): OverlayImage {
    val view = LayoutInflater.from(context).inflate(R.layout.view_marker, null)
    view.findViewById<TextView>(R.id.textView).text = text
```



```

    val bitmap = viewToBitmap(view)
    return OverlayImage.fromBitmap(bitmap)
}

fun viewToBitmap(view: View): Bitmap {
    view.measure(
        View.MeasureSpec.makeMeasureSpec(0, View.MeasureSpec.UNSPECIFIED),
        View.MeasureSpec.makeMeasureSpec(0, View.MeasureSpec.UNSPECIFIED)
    )
    view.layout(0, 0, view.measuredWidth, view.measuredHeight)
    val bitmap = Bitmap.createBitmap(view.measuredWidth, view.measuredHeight,
        Bitmap.Config.ARGB_8888)
    val canvas = Canvas(bitmap)
    view.draw(canvas)
    return bitmap
}

```

그리고:

```

marker.icon = createCustomMarker(context, "여기!")
marker.width = Marker.SIZE_AUTO
marker.height = Marker.SIZE_AUTO

```

◆ `view_marker.xml` 예시:

```

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:padding="4dp"
    android:background="@drawable/marker_bg"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">

    <TextView
        android:id="@+id/textView"
        android:text="텍스트마커"
        android:textColor="@android:color/white"
        android:textStyle="bold"
        android:textSize="14sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>

```

💬 4 정보창(InfoWindow) 커스텀

✓ 기본 InfoWindow

```
val infoWindow = InfoWindow()
infoWindow.adapter = object : InfoWindow.DefaultTextAdapter(context) {
    override fun getText(infoWindow: InfoWindow): CharSequence {
        return "이곳은 서울!"
    }
}
infoWindow.open(marker)
```

✓ 커스텀 레이아웃 InfoWindow

```
infoWindow.adapter = object : InfoWindow.ViewAdapter() {
    override fun getView(infoWindow: InfoWindow): View {
        val view = LayoutInflater.from(context).inflate(R.layout.view_infowindow, null)
        view.findViewById<TextView>(R.id.title).text = "커스텀 정보창!"
        return view
    }
}
```

◆ view_infowindow.xml 예시:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:background="@drawable/infowindow_bg"
    android:padding="8dp"
    android:orientation="vertical"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <TextView
        android:id="@+id/title"
        android:textColor="@android:color/black"
        android:text="정보창 내용"
        android:textSize="14sp" />
</LinearLayout>
```

5 폴리라인 (이동 경로 표시)

✓ 개요

PolylineOverlay 를 사용하면 사용자가 지나온 경로를 선(Line)으로 표시 가능.

```
val path = mutableListOf<LatLng>()
path.add(LatLng(37.5665, 126.9780))
path.add(LatLng(37.5651, 126.98955))
path.add(LatLng(37.5643, 126.9980))
```



```
val polyline = PolylineOverlay()
polyline.coords = path
polyline.color = Color.BLUE
polyline.width = 8
polyline.map = naverMap
```

📍 위치가 실시간으로 변할 때는 `path.add(LatLng(...))` 후 `polyline.coords = path` 를 갱신하면 자동 업데이트됩니다.

🔗 공식 문서 링크

- 📘 [NAVER Map Android SDK 가이드 \(공식\)](#) ↗
- 📘 [Overlay\(마커, 정보창, 폴리라인 등\) 가이드](#) ↗

원하신다면 위 기능들을 **하나의 XML + Activity** 코드 전체 예제로 구성해드릴 수도 있습니다.

예: “내 위치 추적 + 텍스트 마커 + 경로 표시 포함한 `MainActivity.kt` / `activity_main.xml` 전체 코드 예시”

그렇게 만들어드릴까요?



이외 나머지 모든 기능을 적용한 샘플 하나만 알려줘



좋습니다 👍

아래는 **Naver Map SDK (XML 뷰 기반, Kotlin)** 의 거의 모든 주요 기능을 한 화면에 통합한 샘플 예제입니다.

- ✓ 내 위치 표시
- ✓ 지도 이동/줌
- ✓ 커스텀 마커 (텍스트)
- ✓ 커스텀 정보창
- ✓ 폴리라인(이동 경로 표시)
- ✓ 카메라 애니메이션 / 지도 클릭 이벤트 / 마커 클릭 이벤트
- ✓ 권한 처리 및 `FusedLocationSource`

🗺 완성 예제



activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <com.naver.maps.map.MapView
        android:id="@+id/mapView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

</LinearLayout>
```

MainActivity.kt

```
package com.example.naverfullsample

import android.Manifest
import android.content.pm.PackageManager
import android.graphics.*
import android.os.Bundle
import android.view.*
import android.widget.TextView
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import com.example.naverfullsample.databinding.ActivityMainBinding
import com.naver.maps.geometry.LatLng
import com.naver.maps.map.*
import com.naver.maps.map.overlay.*
import com.naver.maps.util.FusedLocationSource

class MainActivity : AppCompatActivity(), OnMapReadyCallback {

    private lateinit var binding: ActivityMainBinding
    private lateinit var naverMap: NaverMap
    private lateinit var locationSource: FusedLocationSource
    private val path = mutableListOf<LatLng>()
    private lateinit var polyline: PolylineOverlay

    companion object {
        private const val LOCATION_PERMISSION_REQUEST_CODE = 1000
    }
}
```

```

}

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = ActivityMainBinding.inflate(layoutInflater)
    setContentView(binding.root)

    // 위치 권한 소스
    locationSource = FusedLocationSource(this, LOCATION_PERMISSION_REQUEST_CODE)

    binding.mapView.onCreate(savedInstanceState)
    binding.mapView.getMapAsync(this)
}

override fun onMapReady(map: NaverMap) {
    naverMap = map
    naverMap.locationSource = locationSource
    naverMap.locationTrackingMode = LocationTrackingMode.Follow

    // 초기 카메라 위치
    val startPos = LatLng(37.5665, 126.9780) // 서울시청
    naverMap.moveCamera(CameraUpdate.scrollTo(startPos, 15.0))

    // 지도 설정
    naverMap.uiSettings.isZoomControlEnabled = true
    naverMap.uiSettings.isCompassEnabled = true

    // 내 위치 오버레이
    val locationOverlay = naverMap.locationOverlay
    locationOverlay.isVisible = true
    locationOverlay.position = startPos

    // 폴리라인 준비
    polyline = PolylineOverlay()
    polyline.color = Color.BLUE
    polyline.width = 10
    polyline.map = naverMap

    // 마커 추가
    val marker = Marker()
    marker.position = startPos
    marker.icon = createTextMarker("서울시청")
    marker.map = naverMap

    // 정보창 커스텀
    val infowindow = Infowindow()

```



```

infoWindow.adapter = object : InfoWindow.ViewAdapter() {
    override fun getView(infoWindow: InfoWindow): View {
        val v = layoutInflater.inflate(R.layout.view_infowindow, null)
        v.findViewById<TextView>(R.id.title).text = "여기는 서울시청!"
        return v
    }
}
marker.setOnClickListener {
    infoWindow.open(marker)
    true
}

// 지도 클릭 시 폴리라인 경로 추가
naverMap.setOnMapClickListener { _, coord ->
    path.add(coord)
    polyline.coords = path

    val newMarker = Marker()
    newMarker.position = coord
    newMarker.icon = createTextMarker("지점 ${path.size}")
    newMarker.map = naverMap
}

// 카메라 움직임 애니메이션 예시
val cameraUpdate = CameraUpdate.scrollTo(LatLng(37.5651, 126.98955), 14.0)
    .animate(CameraAnimation.Fly, 2000L)
naverMap.moveCamera(cameraUpdate)
}

// 커스텀 마커 (텍스트 포함)
private fun createTextMarker(text: String): OverlayImage {
    val view = LayoutInflater.from(this).inflate(R.layout.view_marker, null)
    view.findViewById<TextView>(R.id.textView).text = text
    return OverlayImage.fromBitmap(viewToBitmap(view))
}

private fun viewToBitmap(view: View): Bitmap {
    view.measure(
        View.MeasureSpec.makeMeasureSpec(0, View.MeasureSpec.UNSPECIFIED),
        View.MeasureSpec.makeMeasureSpec(0, View.MeasureSpec.UNSPECIFIED)
    )
    view.layout(0, 0, view.measuredWidth, view.measuredHeight)
    val bitmap = Bitmap.createBitmap(view.measuredWidth, view.measuredHeight,
    Bitmap.Config.ARGB_8888)
    val canvas = Canvas(bitmap)
    view.draw(canvas)
}

```



```

        return bitmap
    }

    // 권한 처리
    override fun onRequestPermissionsResult(requestCode: Int, permissions: Array<out String>, grantResults: IntArray) {
        if (requestCode == LOCATION_PERMISSION_REQUEST_CODE) {
            if (locationSource.onRequestPermissionsResult(requestCode, permissions, grantResults)) {
                if (!locationSource.isActivated) {
                    naverMap.locationTrackingMode = LocationTrackingMode.None
                }
                return
            }
        }
        super.onRequestPermissionsResult(requestCode, permissions, grantResults)
    }

    override fun onStart() { super.onStart(); binding.mapView.onStart() }
    override fun onResume() { super.onResume(); binding.mapView.onResume() }
    override fun onPause() { super.onPause(); binding.mapView.onPause() }
    override fun onStop() { super.onStop(); binding.mapView.onStop() }
    override fun onDestroy() { super.onDestroy(); binding.mapView.onDestroy() }
    override fun onLowMemory() { super.onLowMemory(); binding.mapView.onLowMemory() }
}

```

view_marker.xml (커스텀 마커)

```

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:padding="4dp"
    android:background="@drawable/marker_bg"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content">
    <TextView
        android:id="@+id/textView"
        android:text="마커"
        android:textColor="@android:color/white"
        android:textStyle="bold"
        android:textSize="14sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>

```



view_infowindow.xml (정보창)

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:background="@drawable/infowindow_bg"  
    android:padding="8dp"  
    android:orientation="vertical"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content">  
    <TextView  
        android:id="@+id/title"  
        android:text="정보창 내용"  
        android:textColor="@android:color/black"  
        android:textSize="14sp" />  
</LinearLayout>
```

marker_bg.xml (drawable)

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"  
    android:shape="rectangle">  
    <solid android:color="#2196F3" />  
    <corners android:radius="8dp" />  
    <padding  
        android:left="8dp"  
        android:top="4dp"  
        android:right="8dp"  
        android:bottom="4dp" />  
</shape>
```

infowindow_bg.xml (drawable)

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"  
    android:shape="rectangle">  
    <solid android:color="#FFFFFF" />  
    <stroke android:color="#444444" android:width="1dp" />  
    <corners android:radius="8dp" />  
</shape>
```

실행 결과

- 처음 서울시청 중심으로 지도 표시
 - 내 위치 아이콘 활성화
 - 지도 클릭 시 클릭한 좌표에 마커 추가 + 경로(폴리라인) 업데이트
 - 마커 클릭 시 커스텀 정보창 표시
 - 줌, 틸트, 카메라 애니메이션 모두 사용 가능
-

이 샘플은 **Naver** 지도 주요 기능 대부분을 통합한 형태입니다.

원하신다면 다음 단계로 👉 “실시간 이동 시 경로 자동 업데이트 (**GPS 기반**)” 기능까지 확장한 버전도 바로 드릴 수 있습니다.

그 버전도 만들어드릴까요?



FusedLocationSource를 사용해서 권한 요청을 해야하나?



좋은 질문이에요 👍

결론부터 말하면, 네, FusedLocationSource 를 사용할 때는 위치 권한 요청이 반드시 필요합니다.

🔍 이유

FusedLocationSource 는 내 위치(현재 좌표) 를 얻기 위해

FusedLocationProviderClient (Google Play 서비스의 위치 제공자)를 내부적으로 사용합니다.

따라서 다음 권한이 반드시 필요합니다:

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

⚙️ 구현 순서

1 권한 요청 코드 작성

```
private val LOCATION_PERMISSION_REQUEST_CODE = 1000

private fun checkLocationPermission() {
    val permissions = arrayOf(
        Manifest.permission.ACCESS_FINE_LOCATION,
```



```
        Manifest.permission.ACCESS_COARSE_LOCATION
    )
    if (permissions.all {
        ContextCompat.checkSelfPermission(this, it) ==
PackageManager.PERMISSION_GRANTED
    }) {
        startMap()
    } else {
        ActivityCompat.requestPermissions(this, permissions,
LOCATION_PERMISSION_REQUEST_CODE)
    }
}
```

2 권한 결과 처리

```
override fun onRequestPermissionsResult(
    requestCode: Int,
    permissions: Array<out String>,
    grantResults: IntArray
) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults)
    if (requestCode == LOCATION_PERMISSION_REQUEST_CODE) {
        if (grantResults.all { it == PackageManager.PERMISSION_GRANTED }) {
            fusedLocationSource?.onRequestPermissionsResult(requestCode, permissions,
grantResults)
            naverMap.locationTrackingMode = LocationTrackingMode.Follow
        } else {
            Toast.makeText(this, "위치 권한이 필요합니다.", Toast.LENGTH_SHORT).show()
        }
    }
}
```

3 NaverMap 초기화 시 FusedLocationSource 연결

```
private lateinit var naverMap: NaverMap
private lateinit var fusedLocationSource: FusedLocationSource

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    val binding = ActivityMainBinding.inflate(layoutInflater)
    setContentView(binding.root)

    fusedLocationSource = FusedLocationSource(this, LOCATION_PERMISSION_REQUEST_CODE)
```



```
binding.mapView.getMapAsync { map ->
    naverMap = map
    naverMap.locationSource = fusedLocationSource
    naverMap.locationTrackingMode = LocationTrackingMode.Follow
}

checkLocationPermission()
}
```

🚀 결과

이렇게 하면:

- 권한이 승인되면 지도가 내 위치를 자동으로 추적
- 권한이 거부되면 “위치 권한이 필요합니다” 메시지 표시

원하신다면 이 코드를 포함해서

📍 내 위치 표시 + 지도 이동 + 마커 커스텀 + 정보창 + 폴리라인

까지 모두 구현된 전체 **XML + Kotlin** 예제 버전으로 완성해드릴 수도 있습니다.

그렇게 할까요?



CoroutineWorker로 내 위치 좌표 가져오는 class 생성



좋아요 👍

아래는 CoroutineWorker 를 사용해 주기적으로 내 위치 좌표를 가져오는 클래스 예시입니다.

이 코드는 FusedLocationProviderClient 를 사용하며,

WorkManager 안에서 비동기적으로 현재 위치를 받아와 서버로 전송하거나 DB에 저장할 때 사용할 수 있습니다.

✓ 예시: LocationWorker.kt

```
package com.example.app.worker

import android.Manifest
import android.annotation.SuppressLint
```



```

import android.content.Context
import android.content.pm.PackageManager
import android.location.Location
import androidx.core.content.ContextCompat
import androidx.work CoroutineWorker
import androidx.work.WorkerParameters
import com.google.android.gms.location.FusedLocationProviderClient
import com.google.android.gms.location.LocationServices
import kotlinx.coroutines.tasks.await
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.withContext
import android.util.Log

class LocationWorker(
    applicationContext: Context,
    params: WorkerParameters
) : CoroutineWorker(applicationContext, params) {

    private val fusedClient: FusedLocationProviderClient =
        LocationServices.getFusedLocationProviderClient(applicationContext)

    @SuppressLint("MissingPermission")
    override suspend fun doWork(): Result = withContext(Dispatchers.IO) {
        try {
            // 위치 권한 확인
            if (!hasLocationPermission()) {
                Log.e("LocationWorker", "위치 권한이 없습니다.")
                return@withContext Result.failure()
            }

            // 최신 위치 가져오기 (비동기 -> await)
            val location: Location? = fusedClient.lastLocation.await()

            if (location != null) {
                val latitude = location.latitude
                val longitude = location.longitude

                Log.i("LocationWorker", "현재 위치: $latitude, $longitude")

                // TODO: 이 부분에서 Firestore, Room, 서버 업로드 등 원하는 로직 수행
                // e.g., saveLocationToDatabase(latitude, longitude)

                Result.success()
            } else {
                Log.w("LocationWorker", "위치를 가져올 수 없습니다.")
                Result.retry()
            }
        }
    }
}

```



```

        }
    } catch (e: Exception) {
        Log.e("LocationWorker", "에러 발생: ${e.message}")
        Result.retry()
    }
}

private fun hasLocationPermission(): Boolean {
    val fine = ContextCompat.checkSelfPermission(
        applicationContext,
        Manifest.permission.ACCESS_FINE_LOCATION
    ) == PackageManager.PERMISSION_GRANTED
    val coarse = ContextCompat.checkSelfPermission(
        applicationContext,
        Manifest.permission.ACCESS_COARSE_LOCATION
    ) == PackageManager.PERMISSION_GRANTED
    return fine || coarse
}

```

✓ 권한 추가 (AndroidManifest.xml)

```

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

```

✓ 주기적 실행 예시 (예: 15분마다 실행)

```

import androidx.work.*

val locationWork = PeriodicWorkRequestBuilder<LocationWorker>(
    15, // 최소 15분 간격 (WorkManager 제약)
    java.util.concurrent.TimeUnit.MINUTES
).build()

WorkManager.getInstance(context)
    .enqueueUniquePeriodicWork(
        "LocationTracking",
        ExistingPeriodicWorkPolicy.UPDATE,
        locationWork
)

```



✓ 실행 시 주의점

- WorkManager는 앱이 백그라운드에 있을 때도 주기적으로 실행 가능
- 단, 정확한 주기는 시스템 상황에 따라 약간의 지연이 생길 수 있음
- 더 빠른 주기(예: 1분 이하)로 지속적인 위치 추적을 원하면 → `ForegroundService`를 사용하는 게 적합함

원하신다면 이 Worker가 **Room DB**에 위치를 저장하고,
나중에 Naver Map에 **폴리라인(이동 경로)**로 그리는 버전도 만들어드릴까요?



hilt 적용해서 알려줘



좋아요 👍

그럼 이번엔 `CoroutineWorker`에 **Hilt**를 적용해서
주기적으로 내 위치를 가져오는 **Worker** 예제를 보여드릴게요.

이 버전은

- `HiltWorker` + `@AssistedInject` 방식으로 의존성 주입
- `FusedLocationProviderClient`를 Hilt로 주입받음
- `Room`이나 `Repository` 주입도 확장 가능

✓ 1 Gradle 설정

app/build.gradle

```
plugins {
    id("com.google.dagger.hilt.android")
    kotlin("kapt")
}

dependencies {
    implementation("com.google.dagger:hilt-android:2.52")
    kapt("com.google.dagger:hilt-compiler:2.52")

    // Hilt WorkManager 통합
    implementation("androidx.hilt:hilt-work:1.2.0")
    kapt("androidx.hilt:hilt-compiler:1.2.0")
}
```



```
// WorkManager + Coroutines
implementation("androidx.work:work-runtime-ktx:2.9.0")

// Location
implementation("com.google.android.gms:play-services-location:21.3.0")
}
```

✓ 2 Application 설정

```
@HiltAndroidApp
class MyApplication : Application()
```

✓ 3 FusedLocationProviderClient 제공 모듈

```
package com.example.app.di

import android.content.Context
import com.google.android.gms.location.FusedLocationProviderClient
import com.google.android.gms.location.LocationServices
import dagger.Module
import dagger.Provides
import dagger.hilt.InstallIn
import dagger.hilt.android.qualifiers.ApplicationContext
import dagger.hilt.components.SingletonComponent
import javax.inject.Singleton

@Module
@InstallIn(SingletonComponent::class)
object LocationModule {

    @Provides
    @Singleton
    fun provideFusedLocationProviderClient(
        @ApplicationContext context: Context
    ): FusedLocationProviderClient {
        return LocationServices.getFusedLocationProviderClient(context)
    }
}
```

✓ 4 Hilt 기반 CoroutineWorker

```
package com.example.app.worker

import android.Manifest
import android.annotation.SuppressLint
import android.content.pm.PackageManager
import android.location.Location
import android.util.Log
import androidx.core.content.ContextCompat
import androidx.hilt.work.HiltWorker
import androidx.work CoroutineWorker
import androidx.work.WorkerParameters
import com.google.android.gms.location.FusedLocationProviderClient
import dagger.assisted.Assisted
import dagger.assisted.AssistedInject
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.tasks.await
import kotlinx.coroutines.withContext

@HiltWorker
class LocationWorker @AssistedInject constructor(
    @Assisted private val applicationContext: android.content.Context,
    @Assisted private val params: WorkerParameters,
    private val fusedClient: FusedLocationProviderClient
) : CoroutineWorker(applicationContext, params) {

    @SuppressLint("MissingPermission")
    override suspend fun doWork(): Result = withContext(Dispatchers.IO) {
        try {
            if (!hasLocationPermission()) {
                Log.e("LocationWorker", "위치 권한이 없습니다.")
                return@withContext Result.failure()
            }

            val location: Location? = fusedClient.lastLocation.await()
            if (location != null) {
                Log.i("LocationWorker", "현재 위치: ${location.latitude}, ${location.longitude}")

                // TODO: Repository.saveLocation(location) 등으로 DB나 서버 업로드 가능
                Result.success()
            } else {
                Log.w("LocationWorker", "위치를 가져올 수 없음. 재시도.")
                Result.retry()
            }
        } catch (e: Exception) {
            Log.e("LocationWorker", "에러 발생: ${e.message}")
        }
    }
}
```

```
        Result.retry()
    }

}

private fun hasLocationPermission(): Boolean {
    val fine = ContextCompat.checkSelfPermission(
        applicationContext, Manifest.permission.ACCESS_FINE_LOCATION
    ) == PackageManager.PERMISSION_GRANTED
    val coarse = ContextCompat.checkSelfPermission(
        applicationContext, Manifest.permission.ACCESS_COARSE_LOCATION
    ) == PackageManager.PERMISSION_GRANTED
    return fine || coarse
}
```

✓ 5 Hilt + WorkManager 초기화

HiltWorkerFactory 를 연결해야 Hilt 기반 Worker가 정상 동작합니다.

```
package com.example.app

import android.app.Application
import androidx.hilt.work.HiltWorkerFactory
import androidx.work.Configuration
import dagger.hilt.android.HiltAndroidApp
import javax.inject.Inject

@HiltAndroidApp
class MyApplication : Application(), Configuration.Provider {

    @Inject
    lateinit var workerFactory: HiltWorkerFactory

    override fun getWorkManagerConfiguration() =
        Configuration.Builder()
            .setWorkerFactory(workerFactory)
            .build()
}
```

✓ 6 Worker 실행 코드

```
import androidx.work.*
import java.util.concurrent.TimeUnit
```



```

fun startLocationWorker(context: android.content.Context) {
    val request = PeriodicWorkRequestBuilder<LocationWorker>(
        15, TimeUnit.MINUTES
    ).build()

    WorkManager.getInstance(context)
        .enqueueUniquePeriodicWork(
            "LocationTracking",
            ExistingPeriodicWorkPolicy.UPDATE,
            request
        )
}

```

✓ 7 권한 설정 (AndroidManifest.xml)

```

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

```

이제 이 `LocationWorker`는 Hilt로 관리되며,
필요하면 `@Inject constructor()`를 통해 `Repository`나 `RoomDao`도 함께 주입할 수 있습니다.

원하신다면 이 Worker가 주기적으로 얻은 좌표를

`Room DB`에 저장하고 →

앱 실행 시 `Naver Map`에 **폴리라인(이동 경로)**으로 표시하는 구조까지 연결된 완전 예시도 만들어드릴까요?



worker에서 위치를 가져오는건 1회성이야



정확합니다 

`Coroutineworker` (또는 `Worker`) 는 **한 번 실행될 때 단 한 번의 위치를 가져오는 “1회성 작업”**이에요.
즉,

- `dowork()` → 실행될 때 현재 위치를 한 번만 가져옴
- 완료되면 종료 (`Result.success()` or `Result.retry()` or `Result.failure()`)
- 주기적 위치 추적을 원한다면, `PeriodicWorkRequest`로 주기적 실행 예약을 해야 합니다.

◆ 정리하자면

방식	실행 횟수	설명
OneTimeWorkRequest	1회성	앱에서 요청 시 한 번만 실행됨
PeriodicWorkRequest	주기적 (최소 15분 간격)	시스템이 15분 이상 간격으로 반복 실행 (정확한 시간 보장 X)
ForegroundService	지속 실행 가능	위치 추적, 경로 로깅 등 “실시간 감시”용에 적합

✓ 예시 비교

📍 1회성 요청 (예: 버튼 클릭 시 현재 위치 가져오기)

```
val work = OneTimeWorkRequestBuilder<LocationWorker>().build()
WorkManager.getInstance(context).enqueue(work)
```

⌚ 주기적 요청 (예: 15분마다 실행)

```
val work = PeriodicWorkRequestBuilder<LocationWorker>(
    15, TimeUnit.MINUTES
).build()
```

```
WorkManager.getInstance(context).enqueueUniquePeriodicWork(
    "LocationTracking",
    ExistingPeriodicWorkPolicy.UPDATE,
    work
)
```

🚀 실시간 지속 추적 (예: 1초~10초마다 위치 감시)

➡️ `ForegroundService` 또는 `Foreground + FusedLocationCallback` 사용해야 함.

이 경우 `WorkManager`로는 불가능합니다 (최소 주기가 15분이기 때문).

◆ 결론

- `CoroutineWorker` → 1회성 위치 가져오기용 (서버 업로드 등)
- `PeriodicWorkRequest` → 주기적 백그라운드 위치 기록
- `ForegroundService` → 실시간 위치 추적 / 지도에 경로 표시용



원하시는 기능이 “지도 위에 내가 이동한 경로(폴리라인)를 계속 그리기”라면,

ForegroundService 기반으로 구현하는 게 맞습니다.

그럼 제가 `Hilt + ForegroundService + FusedLocationProviderClient` 구조로
실시간 위치를 가져오고, 경로를 Naver Map에 표시하는 예시도 만들어드릴까요?



```
* Exception is:  
org.gradle.api.tasks.TaskExecutionException: Execution failed  
for task ':app:mergeDebugNativeLibs'.  
        at  
org.gradle.api.internal.tasks.execution.CatchExceptionTaskExec  
uter.execute(CatchExceptionTaskExecuter.java:38)  
        at  
org.gradle.api.internal.tasks.execution.EventFiringTaskExecuter  
$1.executeTask(EventFiringTaskExecuter.java:77)  
        at  
org.gradle.api.internal.tasks.execution.EventFiringTaskExecuter  
$1.call(EventFiringTaskExecuter.java:55)  
        at  
org.gradle.api.internal.tasks.execution.EventFiringTaskExecuter  
$1.call(EventFiringTaskExecuter.java:52)  
        at  
org.gradle.internal.operations.DefaultBuildOperationRunner$Cal  
lableBuildOperationWorker.execute(DefaultBuildOperationRunn  
er.java:210)  
        at  
org.gradle.internal.operations.DefaultBuildOperationRunner$Cal  
lableBuildOperationWorker.execute(DefaultBuildOperationRunn  
er.java:205)  
        at  
org.gradle.internal.operations.DefaultBuildOperationRunner$2.e  
xecute(DefaultBuildOperationRunner.java:67)  
        at  
org.gradle.internal.operations.DefaultBuildOperationRunner$2.e  
xecute(DefaultBuildOperationRunner.java:60)  
        at  
org.gradle.internal.operations.DefaultBuildOperationRunner.exe  
cute(DefaultBuildOperationRunner.java:167)  
        at  
org.gradle.internal.operations.DefaultBuildOperationRunner.exe  
cute(DefaultBuildOperationRunner.java:60)  
        at  
org.gradle.internal.operations.DefaultBuildOperationRunner.call
```



```
(DefaultBuildOperationRunner.java:54)
    at
org.gradle.api.internal.tasks.execution.EventFiringTaskExecuter.
execute(EventFiringTaskExecuter.java:52)
    at
org.gradle.execution.plan.LocalTaskNodeExecutor.execute(Local
TaskNodeExecutor.java:42)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$Inv
okeNodeExecutorsAction.execute(DefaultTaskExecutionGraph.ja
va:331)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$Inv
okeNodeExecutorsAction.execute(DefaultTaskExecutionGraph.ja
va:318)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$Bui
ldOperationAwareExecutionAction.lambda$execute$0(DefaultTa
skExecutionGraph.java:314)
    at
org.gradle.internal.operations.CurrentBuildOperationRef.with(Cu
rrentBuildOperationRef.java:85)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$Bui
ldOperationAwareExecutionAction.execute(DefaultTaskExecutio
nGraph.java:314)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$Bui
ldOperationAwareExecutionAction.execute(DefaultTaskExecutio
nGraph.java:303)
    at
org.gradle.execution.plan.DefaultPlanExecutor$ExecutorWorker.
execute(DefaultPlanExecutor.java:459)
    at
org.gradle.execution.plan.DefaultPlanExecutor$ExecutorWorker.
run(DefaultPlanExecutor.java:376)
    at
org.gradle.execution.plan.DefaultPlanExecutor.process(DefaultP
lanExecutor.java:111)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph.exe
cuteWithServices(DefaultTaskExecutionGraph.java:138)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph.exe
cute(DefaultTaskExecutionGraph.java:123)
    at
```

```
org.gradle.execution.SelectedTaskExecutionAction.execute(Sele
ctedTaskExecutionAction.java:35)
    at
org.gradle.execution.DryRunBuildExecutionAction.execute(DryR
unBuildExecutionAction.java:51)
    at
org.gradle.execution.BuildOperationFiringBuildWorkerExecutor$ExecuteTasks.call(BuildOperationFiringBuildWorkerExecutor.java:54)
    at
org.gradle.execution.BuildOperationFiringBuildWorkerExecutor$ExecuteTasks.call(BuildOperationFiringBuildWorkerExecutor.java:43)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner$CallableBuildOperationWorker.execute(DefaultBuildOperationRunner.java:210)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner$CallableBuildOperationWorker.execute(DefaultBuildOperationRunner.java:205)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner$2.e
xecute(DefaultBuildOperationRunner.java:67)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner$2.e
xecute(DefaultBuildOperationRunner.java:60)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner.exe
cute(DefaultBuildOperationRunner.java:167)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner.exe
cute(DefaultBuildOperationRunner.java:60)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner.call
(DefaultBuildOperationRunner.java:54)
    at
org.gradle.execution.BuildOperationFiringBuildWorkerExecutor.e
xecute(BuildOperationFiringBuildWorkerExecutor.java:40)
    at
org.gradle.internal.build.DefaultBuildLifecycleController.lambda
$executeTasks$10(DefaultBuildLifecycleController.java:313)
    at
org.gradle.internal.model.StateTransitionController.doTransition(
StateTransitionController.java:266)
    at
```

```
org.gradle.internal.model.StateTransitionController.lambda$tryT
ransition$8(StateTransitionController.java:177)
    at
org.gradle.internal.work.DefaultSynchronizer.withLock(DefaultS
ynchronizer.java:46)
    at
org.gradle.internal.model.StateTransitionController.tryTransition
(StateTransitionController.java:177)
    at
org.gradle.internal.build.DefaultBuildLifecycleController.execute
Tasks(DefaultBuildLifecycleController.java:304)
    at
org.gradle.internal.build.DefaultBuildWorkGraphController$Defa
ultBuildWorkGraph.runWork(DefaultBuildWorkGraphController.ja
va:220)
    at
org.gradle.internal.work.DefaultWorkerLeaseService.withLocks(
DefaultWorkerLeaseService.java:263)
    at
org.gradle.internal.work.DefaultWorkerLeaseService.runAsWork
erThread(DefaultWorkerLeaseService.java:127)
    at
org.gradle.composite.internal.DefaultBuildController.doRun(Def
aultBuildController.java:181)
    at
org.gradle.composite.internal.DefaultBuildController.access$00
0(DefaultBuildController.java:50)
    at
org.gradle.composite.internal.DefaultBuildController$BuildOpRu
nnable.lambda$run$0(DefaultBuildController.java:198)
    at
org.gradle.internal.operations.CurrentBuildOperationRef.with(Cu
rrentBuildOperationRef.java:85)
    at
org.gradle.composite.internal.DefaultBuildController$BuildOpRu
nnable.run(DefaultBuildController.java:198)
    at
java.base/java.util.concurrent.Executors$RunnableAdapter.call(
Unknown Source)
    at java.base/java.util.concurrent.FutureTask.run(Unknown
Source)
    at
org.gradle.internal.concurrent.ExecutorPolicy$CatchAndRecordF
ailures.onExecute(ExecutorPolicy.java:64)
    at
org.gradle.internal.concurrent.AbstractManagedExecutor$1.run(
```

```
AbstractManagedExecutor.java:48)
    at
java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(U
nknown Source)
    at
java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(
Unknown Source)
        at java.base/java.lang.Thread.run(Unknown Source)
Caused by:
org.gradle.api.internal.artifacts.ivyservice.TypedResolveExcepti
on: Could not resolve all files for configuration
':app:debugRuntimeClasspath'.
    at
org.gradle.api.internal.artifacts.ResolveExceptionMapper.mapFa
ilure(ResolveExceptionMapper.java:70)
    at
org.gradle.api.internal.artifacts.ResolveExceptionMapper.mapFa
ilures(ResolveExceptionMapper.java:62)
    at
org.gradle.api.internal.artifacts.configurations.DefaultConfigura
tion$DefaultResolutionHost.consolidateFailures(DefaultConfigur
ation.java:1928)
    at
org.gradle.api.internal.artifacts.configurations.ResolutionHost.re
throwFailuresAndReportProblems(ResolutionHost.java:75)
    at
org.gradle.api.internal.artifacts.configurations.ResolutionBacked
FileCollection.maybeThrowResolutionFailures(ResolutionBacked
FileCollection.java:86)
    at
org.gradle.api.internal.artifacts.configurations.ResolutionBacked
FileCollection.visitContents(ResolutionBackedFileCollection.java:
76)
    at
org.gradle.api.internal.file.AbstractFileCollection.getFiles(Abstra
ctFileCollection.java:123)
    at
org.gradle.api.internal.file.AbstractFileCollection.iterator(Abstra
ctFileCollection.java:164)
    at
org.gradle.api.internal.file.FilteredFileCollection.iterator(Filtered
FileCollection.java:80)
    at
com.google.common.collect.ImmutableSet.copyOf(ImmutableS
et.java:264)
    at
```

```
org.gradle.api.internal.file.collections.DefaultConfigurableFileCollection$2.visitCollection(DefaultConfigurableFileCollection.java:411)
    at
org.gradle.api.internal.file.AbstractFileCollection.visitContents(AbstractFileCollection.java:365)
    at
org.gradle.api.internal.file.AbstractFileCollection.visitStructure(AbstractFileCollection.java:360)
    at
org.gradle.api.internal.file.collections.DefaultConfigurableFileCollection.lambda$calculateFinalizedValue$1(DefaultConfigurableFileCollection.java:408)
    at
org.gradle.api.internal.file.collections.UnpackingVisitor.add(UnpackingVisitor.java:67)
    at
org.gradle.api.internal.file.collections.DefaultConfigurableFileCollection$UnresolvedItemsCollector.visitContents(DefaultConfigurableFileCollection.java:616)
    at
org.gradle.api.internal.file.collections.DefaultConfigurableFileCollection.calculateFinalizedValue(DefaultConfigurableFileCollection.java:408)
    at
org.gradle.api.internal.file.collections.DefaultConfigurableFileCollection.finalizeNow(DefaultConfigurableFileCollection.java:124)
    at
org.gradle.api.internal.file.collections.DefaultConfigurableFileCollection.lambda$visitChildren$2(DefaultConfigurableFileCollection.java:432)
    at
org.gradle.api.internal.provider.ValueState.finalizeOnReadIfNecessary(ValueState.java:140)
    at
org.gradle.api.internal.file.collections.DefaultConfigurableFileCollection.visitChildren(DefaultConfigurableFileCollection.java:432)
    at
org.gradle.api.internal.file.CompositeFileCollection.visitContents(CompositeFileCollection.java:113)
    at
org.gradle.api.internal.file.AbstractFileCollection.visitStructure(AbstractFileCollection.java:360)
    at
org.gradle.api.internal.file.CompositeFileCollection.lambda$visitContents$0(CompositeFileCollection.java:113)
```

```
        at
org.gradle.api.internal.file.collections.UnpackingVisitor.add(Unp
ackingVisitor.java:67)
        at
org.gradle.api.internal.file.collections.UnpackingVisitor.add(Unp
ackingVisitor.java:92)
        at
org.gradle.api.internal.file.DefaultFileCollectionFactory$Resolvin
gFileCollection.visitChildren(DefaultFileCollectionFactory.java:30
6)
        at
org.gradle.api.internal.file.CompositeFileCollection.visitContents
(CompositeFileCollection.java:113)
        at
org.gradle.api.internal.file.AbstractFileCollection.visitStructure(A
bstractFileCollection.java:360)
        at
org.gradle.api.internal.file.CompositeFileCollection.lambda$visit
Contents$0(CompositeFileCollection.java:113)
        at
org.gradle.api.internal.tasks.PropertyFileCollection.visitChildren(
PropertyFileCollection.java:48)
        at
org.gradle.api.internal.file.CompositeFileCollection.visitContents
(CompositeFileCollection.java:113)
        at
org.gradle.api.internal.file.AbstractFileCollection.visitStructure(A
bstractFileCollection.java:360)
        at
org.gradle.internal.fingerprint.impl.DefaultFileCollectionSnapsho
tter.snapshot(DefaultFileCollectionSnapshotter.java:47)
        at
org.gradle.internal.execution.impl.DefaultInputFingerprinter$Inp
utCollectingVisitor.visitInputFileProperty(DefaultInputFingerprint
er.java:133)
        at
org.gradle.internal.execution.steps.AbstractSkipEmptyWorkStep
$1.visitInputFileProperty(AbstractSkipEmptyWorkStep.java:104)
        at
org.gradle.api.internal.tasks.execution.TaskExecution.visitRegul
arInputs(TaskExecution.java:324)
        at
org.gradle.internal.execution.steps.AbstractSkipEmptyWorkStep
.lambda$fingerprintPrimaryInputs$2(AbstractSkipEmptyWorkSt
ep.java:100)
        at
```

```
org.gradle.internal.execution.impl.DefaultInputModuleProperties(DefaultInputModuleProperties.java:63)
    at
org.gradle.internal.execution.steps.AbstractSkipEmptyWorkStep
    .fingerprintPrimaryInputs(AbstractSkipEmptyWorkStep.java:95)
    at
org.gradle.internal.execution.steps.AbstractSkipEmptyWorkStep
    .execute(AbstractSkipEmptyWorkStep.java:52)
    at
org.gradle.internal.execution.steps.AbstractSkipEmptyWorkStep
    .execute(AbstractSkipEmptyWorkStep.java:36)
    at
org.gradle.internal.execution.steps.legacy.MarkSnapshottingInputsStartedStep.execute(MarkSnapshottingInputsStartedStep.java:38)
    at
org.gradle.internal.execution.steps.LoadPreviousExecutionStateStep.execute(LoadPreviousExecutionStateStep.java:36)
    at
org.gradle.internal.execution.steps.LoadPreviousExecutionStateStep.execute(LoadPreviousExecutionStateStep.java:23)
    at
org.gradle.internal.execution.steps.HandleStaleOutputsStep.execute(HandleStaleOutputsStep.java:75)
    at
org.gradle.internal.execution.steps.HandleStaleOutputsStep.execute(HandleStaleOutputsStep.java:41)
    at
org.gradle.internal.execution.steps.AssignMutableWorkspaceStep$lambda$execute$0(AssignMutableWorkspaceStep.java:35)
    at
org.gradle.api.internal.tasks.execution.TaskExecution$4.withWorkspace(TaskExecution.java:289)
    at
org.gradle.internal.execution.steps.AssignMutableWorkspaceStep.execute(AssignMutableWorkspaceStep.java:31)
    at
org.gradle.internal.execution.steps.AssignMutableWorkspaceStep.execute(AssignMutableWorkspaceStep.java:22)
    at
org.gradle.internal.execution.steps.ChoosePipelineStep.execute(ChoosePipelineStep.java:40)
    at
org.gradle.internal.execution.steps.ChoosePipelineStep.execute(ChoosePipelineStep.java:23)
    at
```

```
org.gradle.internal.execution.steps.ExecuteWorkBuildOperation
FiringStep.lambda$execute$2(ExecuteWorkBuildOperationFiring
Step.java:67)
    at java.base/java.util.Optional.orElseGet(Unknown Source)
    at
org.gradle.internal.execution.steps.ExecuteWorkBuildOperation
FiringStep.execute(ExecuteWorkBuildOperationFiringStep.java:6
7)
    at
org.gradle.internal.execution.steps.ExecuteWorkBuildOperation
FiringStep.execute(ExecuteWorkBuildOperationFiringStep.java:3
9)
    at
org.gradle.internal.execution.steps.IdentityCacheStep.execute(I
dentityCacheStep.java:46)
    at
org.gradle.internal.execution.steps.IdentityCacheStep.execute(I
dentityCacheStep.java:34)
    at
org.gradle.internal.execution.steps.IdentifyStep.execute(Identif
yStep.java:48)
    at
org.gradle.internal.execution.steps.IdentifyStep.execute(Identif
yStep.java:35)
    at
org.gradle.internal.execution.impl.DefaultExecutionEngine$1.ex
ecute(DefaultExecutionEngine.java:61)
    at
org.gradle.api.internal.tasks.execution.ExecuteActionsTaskExec
uter.executelfValid(ExecuteActionsTaskExecuter.java:127)
    at
org.gradle.api.internal.tasks.execution.ExecuteActionsTaskExec
uter.execute(ExecuteActionsTaskExecuter.java:116)
    at
org.gradle.api.internal.tasks.execution.ProblemsTaskPathTrackin
gTaskExecuter.execute(ProblemsTaskPathTrackingTaskExecuter.j
ava:40)
    at
org.gradle.api.internal.tasks.execution.FinalizePropertiesTaskEx
ecuter.execute(FinalizePropertiesTaskExecuter.java:46)
    at
org.gradle.api.internal.tasks.execution.ResolveTaskExecutionMo
deExecuter.execute(ResolveTaskExecutionModeExecuter.java:51
)
    at
org.gradle.api.internal.tasks.execution.SkipTaskWithNoActionsE
```

```
xecuter.execute(SkipTaskWithNoActionsExecuter.java:57)
    at
org.gradle.api.internal.tasks.execution.SkipOnlyIfTaskExecuter.e
xecute(SkipOnlyIfTaskExecuter.java:74)
    at
org.gradle.api.internal.tasks.execution.CatchExceptionTaskExec
uter.execute(CatchExceptionTaskExecuter.java:36)
    at
org.gradle.api.internal.tasks.execution.EventFiringTaskExecuter
$1.executeTask(EventFiringTaskExecuter.java:77)
    at
org.gradle.api.internal.tasks.execution.EventFiringTaskExecuter
$1.call(EventFiringTaskExecuter.java:55)
    at
org.gradle.api.internal.tasks.execution.EventFiringTaskExecuter
$1.call(EventFiringTaskExecuter.java:52)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner$Cal
lableBuildOperationWorker.execute(DefaultBuildOperationRunn
er.java:210)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner$Cal
lableBuildOperationWorker.execute(DefaultBuildOperationRunn
er.java:205)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner$2.e
xecute(DefaultBuildOperationRunner.java:67)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner$2.e
xecute(DefaultBuildOperationRunner.java:60)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner.exe
cute(DefaultBuildOperationRunner.java:167)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner.exe
cute(DefaultBuildOperationRunner.java:60)
    at
org.gradle.internal.operations.DefaultBuildOperationRunner.call
(DefaultBuildOperationRunner.java:54)
    at
org.gradle.api.internal.tasks.execution.EventFiringTaskExecuter.
execute(EventFiringTaskExecuter.java:52)
    at
org.gradle.execution.plan.LocalTaskNodeExecutor.execute(Local
TaskNodeExecutor.java:42)
    at
```

```
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$InvokeNodeExecutorsAction.execute(DefaultTaskExecutionGraph.java:331)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$InvokeNodeExecutorsAction.execute(DefaultTaskExecutionGraph.java:318)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$BuildOperationAwareExecutionAction.lambda$execute$0(DefaultTaskExecutionGraph.java:314)
    at
org.gradle.internal.operations.CurrentBuildOperationRef.with(CurrentBuildOperationRef.java:85)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$BuildOperationAwareExecutionAction.execute(DefaultTaskExecutionGraph.java:314)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph$BuildOperationAwareExecutionAction.execute(DefaultTaskExecutionGraph.java:303)
    at
org.gradle.execution.plan.DefaultPlanExecutor$ExecutorWorker.execute(DefaultPlanExecutor.java:459)
    at
org.gradle.execution.plan.DefaultPlanExecutor$ExecutorWorker.run(DefaultPlanExecutor.java:376)
    at
org.gradle.execution.plan.DefaultPlanExecutor.process(DefaultPlanExecutor.java:111)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph.executeWithServices(DefaultTaskExecutionGraph.java:138)
    at
org.gradle.execution.taskgraph.DefaultTaskExecutionGraph.execute(DefaultTaskExecutionGraph.java:123)
    at
org.gradle.execution.SelectedTaskExecutionAction.execute(SelectedTaskExecutionAction.java:35)
    at
org.gradle.execution.DryRunBuildExecutionAction.execute(DryRunBuildExecutionAction.java:51)
    at
org.gradle.execution.BuildOperationFiringBuildWorkerExecutor$ExecuteTasks.call(BuildOperationFiringBuildWorkerExecutor.java
```

```
:54)
    at
    org.gradle.execution.BuildOperationFiringBuildWorkerExecutor$ExecuteTasks.call(BuildOperationFiringBuildWorkerExecutor.java:43)
    at
    org.gradle.internal.operations.DefaultBuildOperationRunner$CallableBuildOperationWorker.execute(DefaultBuildOperationRunner.java:210)
    at
    org.gradle.internal.operations.DefaultBuildOperationRunner$CallableBuildOperationWorker.execute(DefaultBuildOperationRunner.java:205)
    at
    org.gradle.internal.operations.DefaultBuildOperationRunner$2.execute(DefaultBuildOperationRunner.java:67)
    at
    org.gradle.internal.operations.DefaultBuildOperationRunner$2.execute(DefaultBuildOperationRunner.java:60)
    at
    org.gradle.internal.operations.DefaultBuildOperationRunner.execute(DefaultBuildOperationRunner.java:167)
    at
    org.gradle.internal.operations.DefaultBuildOperationRunner.execute(DefaultBuildOperationRunner.java:60)
    at
    org.gradle.internal.operations.DefaultBuildOperationRunner.call(DefaultBuildOperationRunner.java:54)
    at
    org.gradle.execution.BuildOperationFiringBuildWorkerExecutor.execute(BuildOperationFiringBuildWorkerExecutor.java:40)
    at
    org.gradle.internal.build.DefaultBuildLifecycleController.lambda$executeTasks$10(DefaultBuildLifecycleController.java:313)
    at
    org.gradle.internal.model.StateTransitionController.doTransition(StateTransitionController.java:266)
    at
    org.gradle.internal.model.StateTransitionController.lambda$tryTransition$8(StateTransitionController.java:177)
    at
    org.gradle.internal.work.DefaultSynchronizer.withLock(DefaultSynchronizer.java:46)
    at
    org.gradle.internal.model.StateTransitionController.tryTransition(StateTransitionController.java:177)
```

```
        at
org.gradle.internal.build.DefaultBuildLifecycleController.execute
Tasks(DefaultBuildLifecycleController.java:304)
        at
org.gradle.internal.build.DefaultBuildWorkGraphController$Defa
ultBuildWorkGraph.runWork(DefaultBuildWorkGraphController.ja
va:220)
        at
org.gradle.internal.work.DefaultWorkerLeaseService.withLocks(
DefaultWorkerLeaseService.java:263)
        at
org.gradle.internal.work.DefaultWorkerLeaseService.runAsWork
erThread(DefaultWorkerLeaseService.java:127)
        at
org.gradle.composite.internal.DefaultBuildController.doRun(Def
aultBuildController.java:181)
        at
org.gradle.composite.internal.DefaultBuildController.access$00
0(DefaultBuildController.java:50)
        at
org.gradle.composite.internal.DefaultBuildController$BuildOpRu
nnable.lambda$run$0(DefaultBuildController.java:198)
        at
org.gradle.internal.operations.CurrentBuildOperationRef.with(Cu
rrentBuildOperationRef.java:85)
        at
org.gradle.composite.internal.DefaultBuildController$BuildOpRu
nnable.run(DefaultBuildController.java:198)
        at
java.base/java.util.concurrent.Executors$RunnableAdapter.call(
Unknown Source)
        at java.base/java.util.concurrent.FutureTask.run(Unknown
Source)
        at
org.gradle.internal.concurrent.ExecutorPolicy$CatchAndRecordF
ailures.onExecute(ExecutorPolicy.java:64)
        at
org.gradle.internal.concurrent.AbstractManagedExecutor$1.run(
AbstractManagedExecutor.java:48)
        at
java.base/java.util.concurrent.ThreadPoolExecutor.runWorker(U
nknown Source)
        at
java.base/java.util.concurrent.ThreadPoolExecutor$Worker.run(
Unknown Source)
        at java.base/java.lang.Thread.run(Unknown Source)
```

Caused by:
org.gradle.internal.resolve.ModuleVersionNotFoundException:
Could not find com.google.android.gms:play-services-location-ktx:21.3.0.
Searched in the following locations:

위에 이슈 발생했어



좋아요 👍 로그의 핵심 부분은 바로 이겁니다 👇

Caused by: org.gradle.internal.resolve.ModuleVersionNotFoundException:
Could not find com.google.android.gms:play-services-location-ktx:21.3.0

즉,

`com.google.android.gms:play-services-location-ktx:21.3.0`
을 **Gradle**에서 찾지 못해서 빌드 실패한 것이에요.

⭐ 원인 정리

- 존재하지 않는 버전을 사용 중 (21.3.0은 실제 존재하지 않음)
- 또는 `google()` 저장소가 `repositories` 블록에 빠짐

✓ 해결 방법

1 build.gradle(:app) 수정

`play-services-location`은 **KTX** 버전이 따로 존재하지 않습니다.

➡ 아래처럼 수정하세요:

```
dependencies {  
    implementation("com.google.android.gms:play-services-location:21.3.0")  
}
```

`play-services-location-ktx` ✗
`play-services-location` ✓

입니다.

2 build.gradle(:project) 저장소 확인

`google()` 저장소가 반드시 있어야 합니다 👇



```
pluginManagement {  
    repositories {  
        google()  
        mavenCentral()  
        gradlePluginPortal()  
    }  
}  
dependencyResolutionManagement {  
    repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)  
    repositories {  
        google()  
        mavenCentral()  
    }  
}
```

3 Gradle 캐시 정리 (문제가 지속될 경우)

터미널에서 실행 ↗

```
./gradlew clean build --refresh-dependencies
```

✓ 최종 의존성 예시

```
dependencies {  
    implementation("com.naver.maps:map-sdk:3.17.0")  
    implementation("com.google.android.gms:play-services-location:21.3.0")  
    implementation("androidx.hilt:hilt-work:1.2.0")  
    kapt("androidx.hilt:hilt-compiler:1.2.0")  
    implementation("androidx.work:work-runtime-ktx:2.9.0")  
}
```

원하신다면 위 의존성을 기반으로

Hilt + CoroutineWorker + FusedLocationProviderClient

로 1회성 내 위치 가져오기 Worker 클래스 전체 코드 (권한포함) 예제까지 완성해드릴까요?



Powered by [ChatGPT Exporter](#)

```
Could not instantiate  
com.khs.preinterview.worker.LocationWorker (Ask Gemini)  
  
java.lang.NoSuchMethodException:  
com.khs.preinterview.worker.LocationWorker.<init> [class  
android.content.Context, class  
androidx.work.WorkerParameters]  
  
at java.lang.Class.getConstructor0(Class.java:3385)  
  
at java.lang.Class.getDeclaredConstructor(Class.java:3067)  
  
at  
androidx.work.WorkerFactory.createWorkerWithDefaultFallback$  
fallbackToReflection(WorkerFactory.kt:87)  
  
at  
androidx.work.WorkerFactory.createWorkerWithDefaultFallback(  
WorkerFactory.kt:96)  
  
at  
androidx.work.impl.WorkerWrapper.runWorker(WorkerWrapper.k  
t:234)  
  
at  
androidx.work.impl.WorkerWrapper.access$runWorker(WorkerW  
rapper.kt:67)
```



좋습니다 👍

이 오류의 핵심 원인은 👇

```
NoSuchMethodException: com.khs.preinterview.worker.LocationWorker.<init> [Context,  
WorkerParameters]
```

즉,

WorkManager 가 Worker 인스턴스를 만들 때

필수 생성자 LocationWorker(Context, WorkerParameters) 를 찾을 수 없다는 뜻이에요.

⭐ 원인 정리

Coroutineworker 또는 Worker 를 상속할 때

반드시 다음과 같은 생성자가 있어야 합니다 👇

```
class LocationWorker(
    applicationContext: Context,
    workerParams: WorkerParameters
) : CoroutineWorker(applicationContext, workerParams)
```

그런데 Hilt를 적용하면서

`@Inject constructor(...)` 를 잘못 지정했거나,
생성자 시그니처가 깨쳤을 가능성이 높습니다.

✓ 해결 방법 (Hilt + WorkManager 통합 시)

1 Worker 정의

```
@HiltWorker
class LocationWorker @AssistedInject constructor(
    @Assisted applicationContext: Context,
    @Assisted workerParams: WorkerParameters,
    private val fusedLocationClient: FusedLocationProviderClient
) : CoroutineWorker(applicationContext, workerParams) {

    override suspend fun doWork(): Result {
        return try {
            val location = fusedLocationClient
                .lastLocation
                .await() // kotlinx-coroutines-play-services 확장 필요

            if (location != null) {
                Log.d("LocationWorker", "위도: ${location.latitude}, 경도: ${location.longitude}")
                Result.success()
            } else {
                Result.retry()
            }
        } catch (e: Exception) {
            e.printStackTrace()
            Result.failure()
        }
    }
}
```

💡 핵심: `@HiltWorker` + `@AssistedInject` + `@Assisted` 세트로 사용해야 함.

2 Hilt WorkerFactory 등록 (Application)

```
@HiltAndroidApp
class MyApplication : Application(), Configuration.Provider {

    @Inject lateinit var workerFactory: HiltWorkerFactory

    override fun getWorkManagerConfiguration() =
        Configuration.Builder()
            .setWorkerFactory(workerFactory)
            .build()

}
```

3 Hilt Module (의존성 제공)

```
@Module
@InstallIn(SingletonComponent::class)
object LocationModule {

    @Provides
    fun provideFusedLocationProviderClient(
        @ApplicationContext context: Context
    ): FusedLocationProviderClient {
        return LocationServices.getFusedLocationProviderClient(context)
    }
}
```

4 Gradle 의존성

```
dependencies {
    implementation("androidx.hilt:hilt-work:1.2.0")
    kapt("androidx.hilt:hilt-compiler:1.2.0")
    implementation("com.google.dagger:hilt-android:2.52")
    kapt("com.google.dagger:hilt-compiler:2.52")

    implementation("com.google.android.gms:play-services-location:21.3.0")
    implementation("androidx.work:work-runtime-ktx:2.9.0")
    implementation("org.jetbrains.kotlinx:kotlinx-coroutines-play-services:1.7.3")
}
```

5 Worker 실행 코드

```
val request = OneTimeWorkRequestBuilder<LocationWorker>().build()
WorkManager.getInstance(context).enqueue(request)
```

✓ 요약

항목	필수
Worker 생성자 시그니처	(Context, WorkerParameters)
Hilt DI 적용 시	@HiltWorker, @AssistedInject, @Assisted
Application	implements Configuration.Provider 로 HiltWorkerFactory 등록
Gradle	hilt-work + kotlinx-coroutines-play-services 필요

원하신다면 위 코드를 기반으로

👉 위치 권한 요청 → Hilt + WorkManager로 1회성 위치 저장

까지 완전 동작하는 예시 프로젝트 구조 (viewModel, worker, DI)도 만들어드릴까요?



```
package com.khs.preinterview

import androidx.lifecycle.ViewModel
import androidx.work.OneTimeWorkRequest
import androidx.work.WorkManager
import com.khs.preinterview.worker.LocationWorker
import dagger.hilt.android.lifecycle.HiltViewModel
import jakarta.inject.Inject
import kotlinx.coroutines.flow.Flow
import java.util.concurrent.TimeUnit

@HiltViewModel
class MainViewModel @Inject constructor(
    private val locationDao: LocationDao,
    private val workManager: WorkManager
) : ViewModel() {

    val locationRecords: Flow<List<LocationModel>> =
        locationDao.getAllRecords()

    fun enqueueLocationWork() {
```

```

        val locationWorkRequest: OneTimeWorkRequest =
    OneTimeWorkRequest.Builder(LocationWorker::class.java)
        .setInitialDelay(2, TimeUnit.MINUTES)
        .setBackoffCriteria(
            androidx.work.BackoffPolicy.LINEAR,
            10,
            TimeUnit.SECONDS
        )
        .build()

    workManager.enqueue(locationWorkRequest)
}
}

```

위 코드에서 db에서 받아온 좌표를 가지고 마커를 쓰고싶어. 추가될때
마다 자동으로 마커가 늘어나야하고, 마커에 text를, 정보창에도 text를
쓸거야



좋아요 👍 정확히 이해했습니다.

원하시는 건 다음 흐름이에요 ↴

- 1 Room DB(`locationDao`)에서 `Flow<List<LocationModel>>`로 좌표 목록을 관찰
- 2 새로운 위치가 추가될 때마다 자동으로 마커 추가
- 3 마커에 텍스트 표시,
- 4 마커 클릭 시 정보창에도 텍스트 표시

아래는 XML 기반 NaverMap + ViewModel + LiveData/Flow 연동 전체 구조 예시입니다.

✓ 1 데이터 모델 (LocationModel)

```

@Entity(tableName = "location")
data class LocationModel(
    @PrimaryKey(autoGenerate = true) val id: Int = 0,
    val latitude: Double,
    val longitude: Double,
    val title: String,
    val snippet: String // 마커 클릭 시 정보창에 표시할 텍스트
)

```

✓ 2 ViewModel (현재 코드 기반 개선)

`locationRecords` Flow를 Naver Map 쪽에서 수집하면 됩니다 ↴



```
@HiltViewModel
class MainViewModel @Inject constructor(
    private val locationDao: LocationDao,
    private val workManager: WorkManager
) : ViewModel() {

    val locationRecords: Flow<List<LocationModel>> = locationDao.getAllRecords()

    fun enqueueLocationWork() {
        val locationWorkRequest = OneTimeWorkRequest.Builder(LocationWorker::class.java)
            .setInitialDelay(2, TimeUnit.MINUTES)
            .setBackoffCriteria(BackoffPolicy.LINEAR, 10, TimeUnit.SECONDS)
            .build()
        workManager.enqueue(locationWorkRequest)
    }
}
```

✓ 3 XML Layout (Naver Map View 포함)

```
<layout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <data>
        <variable
            name="viewModel"
            type="com.khs.preinterview.MainViewModel" />
    </data>

    <FrameLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent">

        <com.naver.maps.map.MapView
            android:id="@+id/naverMapView"
            android:layout_width="match_parent"
            android:layout_height="match_parent" />
    </FrameLayout>
</layout>
```

✓ 4 Fragment / Activity 코드

Room DB에서 값이 추가될 때마다 자동으로 마커 갱신됩니다.



```
@AndroidEntryPoint
class MainFragment : Fragment() {

    private lateinit var binding: FragmentMainBinding
    private val viewModel: MainViewModel by viewModels()
    private lateinit var mapView: MapView
    private var naverMap: NaverMap? = null
    private val markers = mutableListOf<Marker>()

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View {
        binding = FragmentMainBinding.inflate(inflater, container, false)
        mapView = binding.naverMapView
        mapView.onCreate(savedInstanceState)

        mapView.getMapAsync { map ->
            naverMap = map
            observeLocations()
        }
        return binding.root
    }

    private fun observeLocations() {
        lifecycleScope.launch {
            viewModel.locationRecords.collect { list ->
                updateMarkers(list)
            }
        }
    }

    private fun updateMarkers(locations: List<LocationModel>) {
        val map = naverMap ?: return

        // 기존 마커 제거
        markers.forEach { it.map = null }
        markers.clear()

        // 새 마커 추가
        for (loc in locations) {
            val marker = Marker().apply {
                position = LatLng(loc.latitude, loc.longitude)
                captionText = loc.title           // 마커에 표시할 텍스트
                width = Marker.SIZE_AUTO
            }
            map.addMarker(marker)
            markers.add(marker)
        }
    }
}
```

```

        height = Marker.SIZE_AUTO
        map = naverMap

        // 정보창 설정
        infowindow = InfoWindow().apply {
            adapter = object : InfoWindow.DefaultTextAdapter(requireContext()) {
                override fun getText(infoWindow: InfoWindow): CharSequence {
                    return loc.snippet
                }
            }
        }

        setOnClickListener {
            if (infowindow?.map == null) infowindow?.open(this)
            else infowindow?.close()
            true
        }
    }
    markers.add(marker)
}

// 마지막 마커로 카메라 이동
if (locations.isNotEmpty()) {
    val last = locations.last()
    val cameraUpdate = CameraUpdate.scrollTo(LatLng(last.latitude,
last.longitude))
    map.moveCamera(cameraUpdate)
}
}

override fun onStart() { super.onStart(); mapView.onStart() }
override fun onResume() { super.onResume(); mapView.onResume() }
override fun onPause() { super.onPause(); mapView.onPause() }
override fun onStop() { super.onStop(); mapView.onStop() }
override fun onDestroyView() { super.onDestroyView(); mapView.onDestroy() }
}

```

✓ 5 작동 흐름

단계

설명

CoroutineWorker

주기적으로 LocationDao.insert() 실행

ViewModel.locationRecords

Room Flow로 DB 데이터 emit



단계	설명
Fragment.observeLocations()	Flow collect 하여 마커 갱신
updateMarkers()	마커 + 정보창 자동 추가 / 갱신

✓ 6 마커 커스텀 (텍스트 스타일 포함)

마커 텍스트 색상, 테두리 등 설정 가능 👇

```
marker.captionText = loc.title
marker.captionColor = Color.BLUE
marker.captionHaloColor = Color.WHITE
marker.captionTextSize = 14f
marker.iconTintColor = Color.RED
```

◆ 정리

✓ 자동 마커 갱신 구조

- Room Flow → collect → NaverMap 갱신

✓ 텍스트 마커 + 정보창

- captionText + InfoWindow.DefaultTextAdapter

✓ 확장 가능

- 폴리라인 (경로 표시), 카메라 이동 애니메이션 등 쉽게 추가 가능

원하신다면 이 코드에 이어서

👉 “DB에 저장된 좌표를 선으로 연결하는 **폴리라인(경로 표시)** 기능”까지 바로 추가해드릴까요?



market에 textView를 넣은 커스텀파일을 사용하고싶어



좋아요 👏

지금 말씀하신 건 **NaverMap의 Marker**를 완전히 커스텀 **View(TextView 등으로 구성)** 해서 표시하고 싶다는 거죠?

즉, `captionText` 같은 단순 텍스트가 아니라,

TextView 포함된 커스텀 레이아웃 XML 을 마커로 쓰고 싶은 상황이에요 ✓



✓ Naver Map에서 커스텀 마커 만들기 (XML + TextView)

NaverMap은 BitmapDescriptor 기반으로 커스텀 마커를 지원합니다.

즉, 커스텀 View → Bitmap 변환 → Marker.icon에 적용 방식으로 구현합니다.

◆ 1 마커용 커스텀 레이아웃 XML

res/layout/custom_marker.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:orientation="vertical"  
    android:padding="4dp"  
    android:background="@drawable/bg_marker"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:gravity="center">  
  
    <TextView  
        android:id="@+id/markerText"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:text="Marker"  
        android:textColor="@android:color/white"  
        android:textStyle="bold"  
        android:textSize="12sp"  
        android:gravity="center" />  
/</LinearLayout>
```

◆ 2 마커 백그라운드 (둥근 배경 예시)

res/drawable/bg_marker.xml

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"  
    android:shape="rectangle">  
    <solid android:color="#E53935"/>  
    <corners android:radius="12dp"/>  
    <padding  
        android:left="6dp"  
        android:top="4dp"  
        android:right="6dp"  
        android:bottom="4dp"/>  
/</shape>
```



◆ 3 View → Bitmap 변환 함수

Extension 으로 만들어두면 재사용 쉬움 ↗

```
fun View.toBitmap(): Bitmap {
    measure(
        View.MeasureSpec.makeMeasureSpec(0, View.MeasureSpec.UNSPECIFIED),
        View.MeasureSpec.makeMeasureSpec(0, View.MeasureSpec.UNSPECIFIED)
    )
    layout(0, 0, measuredWidth, measuredHeight)
    val bitmap = Bitmap.createBitmap(measuredWidth, measuredHeight,
        Bitmap.Config.ARGB_8888)
    val canvas = Canvas(bitmap)
    draw(canvas)
    return bitmap
}
```

◆ 4 커스텀 마커 생성 코드

Fragment 내부에서 updateMarkers() 내의 Marker() 생성 부분을 아래처럼 수정 ↗

```
private fun createCustomMarker(context: Context, text: String): BitmapDescriptor {
    val markerView = LayoutInflater.from(context).inflate(R.layout.custom_marker, null)
    val textView = markerView.findViewById<TextView>(R.id.markerText)
    textView.text = text
    val bitmap = markerView.toBitmap()
    return BitmapDescriptorFactory.fromBitmap(bitmap)
}
```

◆ 5 마커 생성 시 적용

```
val marker = Marker().apply {
    position = LatLng(loc.latitude, loc.longitude)
    icon = createCustomMarker(requireContext(), loc.title) // 커스텀 뷰 적용
    width = Marker.SIZE_AUTO
    height = Marker.SIZE_AUTO
    map = naverMap

    // 클릭 시 정보창 열기
    infoWindow = InfoWindow().apply {
        adapter = object : InfoWindow.DefaultTextAdapter(requireContext()) {
            override fun getText(infoWindow: InfoWindow): CharSequence {
                return loc.snippet
            }
    }}
```



```
        }

    setOnClickListener {
        if (infoWindow?.map == null) infoWindow?.open(this)
        else infoWindow?.close()
        true
    }
}
```

✓ 결과

👉 완벽히 커스터マイ징된 마커가 생성됩니다.

- 마커 위에 TextView 표시 가능
- Text 색상, 폰트, 배경 전부 커스터マイ즈 가능
- 동적으로 텍스트 변경 가능
- Room DB Flow로 간신 시 자동 반영됨

⚙ 확장 가능

이 방식으로 ImageView, TextView, ProgressBar, ConstraintLayout 등

모든 Android View 조합을 마커로 표현 가능합니다.

원하신다면,

👉 “이 커스텀 마커를 터치했을 때 아래쪽에 BottomSheetDialog로 상세 정보 표시”
기능까지 이어서 추가해드릴까요?



```
package com.khs.preinterview.presentation

import android.Manifest
import android.content.Context
import android.content.pm.PackageManager
import android.os.Build
import android.os.Bundle
import android.view.LayoutInflater
import android.widget.Toast
import androidx.activity.viewModels
```



```
import androidx.appcompat.app.AppCompatActivity
import androidx.core.app.ActivityCompat
import androidx.core.content.ContextCompat
import androidx.databinding.DataBindingUtil
import androidx.lifecycle.MutableLiveData
import androidx.lifecycle.lifecycleScope
import
com.google.android.gms.location.FusedLocationProviderClient
import com.google.android.gms.location.LocationServices
import com.khs.preinterview.data.local.model.LocationModel
import com.khs.preinterview.R
import com.khs.preinterview.TimeUtil
import com.khs.preinterview.databinding.ActivityMainBinding
import com.khs.preinterview.databinding.CustomMarkerBinding
import com.khs.preinterview.toBitmap
import com.naver.maps.geometry.LatLng
import com.naver.maps.map.LocationTrackingMode
import com.naver.maps.map.MapView
import com.naver.maps.map.NaverMap
import com.naver.maps.map.OnMapReadyCallback
import com.naver.maps.map.overlay.InfoWindow
import com.naver.maps.map.overlay.Marker
import com.naver.maps.map.overlay.Overlay
import com.naver.maps.map.overlay.OverlayImage
import com.naver.maps.map.util.FusedLocationSource
import dagger.hilt.android.AndroidEntryPoint
import kotlinx.coroutines.flow.collectLatest
import kotlinx.coroutines.launch

@AndroidEntryPoint
class MainActivity : AppCompatActivity(), OnMapReadyCallback
{

    private lateinit var binding: ActivityMainBinding

    private lateinit var mapView: MapView
    private lateinit var naverMap: NaverMap
    private val markerList = mutableListOf<Marker>()
    private var marketHistory = emptyList<LocationModel>()
    private val currentWindow = InfoWindow()

    private lateinit var fusedLocationClient:
FusedLocationProviderClient
    private lateinit var locationSource: FusedLocationSource

    val isBasic = MutableLiveData(true)
```

```
private val viewModel: MainViewModel by viewModels()

companion object {
    private const val LOCATION_PERMISSION_REQUEST_CODE
        = 1000
    private const val
    BACKGROUND_LOCATION_PERMISSION_REQUEST_CODE = 1001
}
```



```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = DataBindingUtil.setContentView(this,
R.layout.activity_main)
    binding.activity = this

    binding.lifecycleOwner = this
    mapView = binding.mapView

    mapView.onCreate(savedInstanceState)
    mapView.getMapAsync(this)
    requestLocationPermissions()
    locationSource = FusedLocationSource(
        this,
        LOCATION_PERMISSION_REQUEST_CODE
    )
    fusedLocationClient =
    LocationServices.getFusedLocationProviderClient(this)

    binding.tvSave.setOnClickListener {
        if
        (checkSelfPermission(Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {
            viewModel.enqueueLocationWork()
            Toast.makeText(this, "현 위치 저장 작업 요청됨.",
            Toast.LENGTH_SHORT).show()
        } else {
            requestLocationPermissions()
            Toast.makeText(this, "위치 권한이 필요합니다.",
            Toast.LENGTH_SHORT).show()
        }
    }
}

override fun onMapReady(map: NaverMap) {
```

```

        naverMap = map
        naverMap.locationSource = locationSource
        naverMap.locationTrackingMode =
        LocationTrackingMode.Face //NONE,NOFOLLOW,FOLLOW,FACE

        // 지도 설정
        naverMap.uiSettings.isZoomControlEnabled = true
        naverMap.uiSettings.isCompassEnabled = true
        naverMap.uiSettings.isScaleBarEnabled = true
        naverMap.uiSettings.isLocationButtonEnabled = false

        binding.locationBtn.map = naverMap

        naverMap.setOnMapClickListener { point, latLng ->
            currentWindow.close()
        }

        observeLocationData()
    }

override fun onRequestPermissionsResult(
    requestCode: Int,
    permissions: Array<String>,
    grantResults: IntArray
) {
    if (requestCode ==
LOCATION_PERMISSION_REQUEST_CODE) {
        if (grantResults.isNotEmpty() && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
            requestBackgroundLocationPermission()
        }

        if
(locationSource.onRequestPermissionsResult(requestCode,
permissions, grantResults)) {
            if (!locationSource.isActivated) {
                naverMap.locationTrackingMode =
                LocationTrackingMode.None
            }
        }
    }
    super.onRequestPermissionsResult(requestCode,
permissions, grantResults)
}

```

```
    private fun checkLocationPermission() =  
        ContextCompat.checkSelfPermission(  
            this,  
            Manifest.permission.ACCESS_FINE_LOCATION  
) == PackageManager.PERMISSION_GRANTED  
  
    private fun requestLocationPermissions() {  
        if (!checkLocationPermission()) {  
            ActivityCompat.requestPermissions(  
                this,  
  
                arrayOf(Manifest.permission.ACCESS_FINE_LOCATION),  
                LOCATION_PERMISSION_REQUEST_CODE  
            )  
        } else {  
            requestBackgroundLocationPermission()  
        }  
    }  
  
    private fun requestBackgroundLocationPermission() {  
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.Q) {  
            if (ContextCompat.checkSelfPermission(  
                this,  
  
                Manifest.permission.ACCESS_BACKGROUND_LOCATION  
) != PackageManager.PERMISSION_GRANTED  
            ) {  
                ActivityCompat.requestPermissions(  
                    this,  
  
                    arrayOf(Manifest.permission.ACCESS_BACKGROUND_LOCATION)  
,  
  
                    BACKGROUND_LOCATION_PERMISSION_REQUEST_CODE  
                )  
            }  
        }  
    }  
  
    private fun observeLocationData() {  
        lifecycleScope.launch {  
            viewModel.locationRecords.collectLatest { records ->  
                updateMapMarkers(records)  
            }  
        }  
    }  
}
```

```

        private fun updateMapMarkers(records:
List<LocationModel>) {
    if (!::naverMap.isInitialized) return

    val newRecords = records.filter { it !in marketHistory }
    if (newRecords.isEmpty()) {
        return
    }

    newRecords.forEach { record ->
        val marker = Marker().apply {
            position = LatLng(record.latitude, record.longitude)
            icon = createCustomMarker(this@MainActivity,
record.id.toString())
            width = 60
            height = 60
            map = naverMap

            onClickListener = Overlay.OnClickListener {
                currentWindow.adapter =
                    object :
InfoWindow.DefaultTextAdapter(this@MainActivity) {
                        override fun getText(infoWindow2:
InfoWindow): CharSequence {
                            return
TimeUtil.formatFromMillis(record.timestamp)
                        }
                    }
                if (currentWindow.marker == this) {
                    currentWindow.close()
                } else {
                    currentWindow.open(this)
                }
                true
            }
        }
        markerList.add(marker)
    }
    marketHistory = records
}

private fun createCustomMarker(context: Context, text:
String): OverlayImage {
    val binding =

```

```
        DataBindingUtil.inflate<CustomMarkerBinding>(
            LayoutInflater.from(context),
            R.layout.custom_marker,
            null,
            false
        )

        binding.markerText = text
        binding.executePendingBindings()

        val markerView = binding.root
        val bitmap = markerView.toBitmap()
        return OverlayImage.fromBitmap(bitmap)
    }

    fun toggleMapType() {
        isBasic.value = !(isBasic.value ?: false)
        naverMap.mapType =
            if (isBasic.value ?: true) NaverMap.MapType.Basic else
        NaverMap.MapType.Satellite
    }

    override fun onSaveInstanceState(outState: Bundle) {
        super.onSaveInstanceState(outState)
        mapView.onSaveInstanceState(outState)
    }

    override fun onStart() {
        super.onStart(); binding.mapView.onStart()
    }

    override fun onResume() {
        super.onResume(); binding.mapView.onResume()
    }

    override fun onPause() {
        super.onPause(); binding.mapView.onPause()
    }

    override fun onStop() {
        super.onStop(); binding.mapView.onStop()
    }

    override fun onDestroy() {
        super.onDestroy(); binding.mapView.onDestroy()
    }
```

```
        override fun onLowMemory() {
            super.onLowMemory(); binding.mapView.onLowMemory()
        }
    }
```

이거 compose 로도 변환이 가능할까?



좋아요 👍 아주 잘 짜여진 코드입니다.

결론부터 말하면 —

가능합니다! 완벽하게 **Compose**로 변환할 수 있습니다.

다만 주의할 점은 Naver Map SDK가 Jetpack Compose를 직접 지원하지 않습니다.

그래서 `AndroidView` 를 사용해 `MapView` 를 Compose 트리 안에 포함시키는 방식으로 변환해야 합니다.

Compose 변환 개념 요약

기존 XML 코드	Compose 변환
<code>activity_main.xml</code>	<code>@Composable fun MainScreen()</code>
<code>MapView</code>	<code>AndroidView(factory = { MapView(it) })</code>
<code>DataBinding</code>	<code>ViewModel + collectAsState()</code>
<code>LiveData / Flow</code>	<code>collectAsState()</code>
<code>Button , TextView</code>	Compose의 <code>Button , Text</code>
<code>toBitmap()</code> 커스텀 마커 동일하게 사용 가능 (View → Bitmap 변환)	

전체 Compose 변환 예시

```
@AndroidEntryPoint
class MainActivity : ComponentActivity() {

    private val viewModel: MainViewModel by viewModels()
    private lateinit var fusedLocationClient: FusedLocationProviderClient
    private lateinit var locationSource: FusedLocationSource

    companion object {
        private const val LOCATION_PERMISSION_REQUEST_CODE = 1000
    }
}
```



```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    fusedLocationClient = LocationServices.getFusedLocationProviderClient(this)
    locationSource = FusedLocationSource(this, LOCATION_PERMISSION_REQUEST_CODE)

    setContent {
        NaverMapComposeScreen(
            viewModel = viewModel,
            locationSource = locationSource,
            context = this
        )
    }
}
```



NaverMapComposeScreen.kt

```
@Composable
fun NaverMapComposeScreen(
    viewModel: MainViewModel,
    locationSource: FusedLocationSource,
    context: Context
) {
    val locationRecords by viewModel.locationRecords.collectAsState(initial =
emptyList())
    val isBasic = remember { mutableStateOf(true) }

    Column(modifier = Modifier.fillMaxSize()) {
        AndroidView(
            factory = { ctx ->
                val mapView = MapView(ctx)
                mapView.getMapAsync { naverMap ->
                    setupNaverMap(
                        naverMap = naverMap,
                        records = locationRecords,
                        context = ctx
                    )
                    naverMap.locationSource = locationSource
                    naverMap.locationTrackingMode = LocationTrackingMode.Face
                }
                mapView
            },
            modifier = Modifier.weight(1f)
        )
    }
}
```



Powered by [ChatGPT Exporter](#)

```

        )

Row(
    modifier = Modifier
        .fillMaxWidth()
        .padding(8.dp),
    horizontalArrangement = Arrangement.SpaceEvenly
) {
    Button(onClick = {
        viewModel.enqueueLocationWork()
        Toast.makeText(context, "현 위치 저장 요청됨", Toast.LENGTH_SHORT).show()
    }) {
        Text("현 위치 저장")
    }

    Button(onClick = {
        isBasic.value = !isBasic.value
    }) {
        Text("지도 모드 전환")
    }
}
}

```

🧭 지도 설정 + 마커 추가 로직

```

fun setupNaverMap(
    naverMap: NaverMap,
    records: List<LocationModel>,
    context: Context
) {
    naverMap.uiSettings.apply {
        isZoomControlEnabled = true
        isCompassEnabled = true
        isScaleBarEnabled = true
    }

    val markerList = mutableListOf<Marker>()
    val infoWindow = InfoWindow()

    records.forEach { record ->
        val marker = Marker().apply {
            position = LatLng(record.latitude, record.longitude)
            icon = createCustomMarker(context, record.id.toString())
        }
    }
}

```

```

        width = 60
        height = 60
        map = naverMap

        onClickListener = Overlay.OnClickListener {
            infoWindow.adapter = object : InfoWindow.DefaultTextAdapter(context) {
                override fun getText(infoWindow2: InfoWindow): CharSequence {
                    return TimeUtil.formatFromMillis(record.timestamp)
                }
            }
            if (infoWindow.marker == this) infoWindow.close()
            else infoWindow.open(this)
            true
        }
    }
    markerList.add(marker)
}

if (records.isNotEmpty()) {
    val last = records.last()
    val update = CameraUpdate.scrollTo(LatLng(last.latitude, last.longitude))
    naverMap.moveCamera(update)
}
}

```

🎨 커스텀 마커 (Compose에서도 동일하게 사용 가능)

```

fun createCustomMarker(context: Context, text: String): OverlayImage {
    val binding = DataBindingUtil.inflate<CustomMarkerBinding>(
        LayoutInflater.from(context),
        R.layout.custom_marker,
        null,
        false
    )

    binding.markerText = text
    binding.executePendingBindings()

    val markerView = binding.root
    val bitmap = markerView.toBitmap()
    return OverlayImage.fromBitmap(bitmap)
}

```



✓ 권장 구조

- `MainActivity` → Compose 진입점
- `NaverMapComposeScreen()` → UI 정의
- `setupNaverMap()` → 지도 초기화 + 마커 추가
- `ViewModel` → Room Flow 관찰 (자동 마커 갱신)
- `Worker` → 주기적 위치 저장

⚙️ Compose로 전환 시 주의점

항목	주의사항
<code>AndroidView</code>	<code>MapView</code> 는 Compose 생명주기와 분리되어 있으므로 직접 관리할 필요 없음
<code>collectAsState()</code>	Flow를 UI에서 안전하게 수집
<code>remember</code>	상태 보존 필요 시 사용
<code>Context</code>	Compose 내부에서는 <code>LocalContext.current</code> 로 접근 가능

⭐ 정리

- ✓ Compose에서도 완전히 동일한 기능 구현 가능
- ✓ 커스텀 마커 / 정보창 / 위치 추적 / Worker 연동 전부 동일
- ✓ 단, 지도 부분만 `AndroidView(MapView)` 로 감싸서 사용해야 함

원하신다면, 위 Compose 코드에

👉 Room Flow 가 갱신될 때 자동으로 `MapView` 내부 마커 업데이트가 반영되도록 `LaunchedEffect` 까지 적용한 “완전 실사용 버전”으로 이어서 정리해드릴까요?



LaunchedEffect로 `isBasic`은 처리했어, 마커 업데이트 알려줘



좋아요 👍 이미 `isBasic` 상태 처리는 `LaunchedEffect`로 하셨다면,

이제 마커 업데이트 도 Compose의 상태 기반으로 자동 반영되게 만들 수 있습니다.

아래는 `naverMap` Compose 환경에서 `locationRecords` 가 변경될 때마다 마커를 갱신하는 Compose 스타일 예시입니다 📸



✓ 전제

- `viewModel.locationRecords: Flow<List<LocationModel>>`
- `LocationModel → (latitude: Double, longitude: Double, id: Int, timestamp: Long)`
- XML 커스텀 마커처럼 **텍스트 포함 마커**를 표시하고 싶음

Compose 코드 예시

```
@Composable
fun NaverMapScreen(
    viewModel: MainViewModel = hiltViewModel()
) {
    val context = LocalContext.current
    val locationRecords by viewModel.locationRecords.collectAsState(initial =
emptyList())

    // 네이버 지도 컨트롤러 상태
    val mapView = remember { MapView(context) }
    val mapReady = remember { mutableStateOf<NaverMap?>(null) }

    AndroidView(
        factory = {
            mapView.apply {
                getMapAsync { naverMap ->
                    mapReady.value = naverMap
                    naverMap.locationTrackingMode = LocationTrackingMode.Face
                    naverMap.uiSettings.apply {
                        isZoomControlEnabled = true
                        isCompassEnabled = true
                        isScaleBarEnabled = true
                    }
                }
            }
        },
        modifier = Modifier.fillMaxSize()
    )

    //💡 마커 업데이트는 locationRecords가 바뀔 때마다 실행됨
    LaunchedEffect(locationRecords, mapReady.value) {
        val naverMap = mapReady.value ?: return@LaunchedEffect

        // 이전 마커 제거
        naverMap.overlays.filterIsInstance<Marker>().forEach { it.map = null }
    }
}
```



```
// 새 마커 추가
locationRecords.forEach { record ->
    val marker = Marker().apply {
        position = LatLng(record.latitude, record.longitude)
        icon = createCustomMarker(context, record.id.toString())
        width = 60
        height = 60
        map = naverMap
    }
    onClickListener = Overlay.OnClickListener {
        val infoWindow = InfoWindow().apply {
            adapter = object : InfoWindow.DefaultTextAdapter(context) {
                override fun getText(infoWindow: InfoWindow): CharSequence {
                    return TimeUtil.formatFromMillis(record.timestamp)
                }
            }
        }
        infoWindow.open(this)
        true
    }
}
}
```

커스텀 마커 함수 (XML 재활용)

기존 Activity 코드 그대로 사용 가능합니다

```
fun createCustomMarker(context: Context, text: String): OverlayImage {
    val binding = DataBindingUtil.inflate<CustomMarkerBinding>(
        LayoutInflater.from(context),
        R.layout.custom_marker,
        null,
        false
    )
    binding.markerText = text
    binding.executePendingBindings()

    val markerView = binding.root
    val bitmap = markerView.toBitmap()
    return OverlayImage.fromBitmap(bitmap)
}
```

💬 작동 방식 요약

단계 설명

- 1 `viewModel.locationRecords` Flow → Compose에서 collect
- 2 데이터 변경 시 `LaunchedEffect` 트리거
- 3 기존 마커 모두 제거 후 새 마커 추가
- 4 각 마커 클릭 시 `InfoWindow` 표시
- 5 XML 기반 커스텀 마커(`CustomMarkerBinding`) 그대로 재사용

원하신다면 `Polyline`(지나간 길 표시)도 Compose 기반으로 같이 추가해드릴 수 있습니다.

→ 같이 넣을까요?

