# NAVER MAP OPEN API

20070298 김재호
20070285 김윤호
20070362 김현곤
20071223 조규철

# 목차

- 소개
- 준비 사항
- 지도생성
- 현재 위치 표시
- 오버레이 객체 표시

NAVER

# 소개

- 웹사이트와 아이폰, 안드로이드 스마트폰에 지도를 표시할 수 있음
- 지도 상의 원하는 위치에 데이터를 함께 표시할 수 있음

NAVER

# 개발 환경

| 지도 API | 설명 |
| --- | --- |
| JavaScript 2.0 | JavaScript 1.0에서 좌표 계산 등의 성능과<br>기능을 개선한 버전 제공 |
| JavaScript 1.0 | 초기 JavaScript 네이버 지도 API |
| Static Map 1.0 | JavaScript 없이 웹 페이지에<br>네이버 지도를 보여줄 때 사용 |
| Flash | 플래쉬, 플렉스에서 액션스크립트 3.0으로<br>개발한 서비스 에서 지도 제공 |
| iOS | iOS 3.1이상에서 네이버 지도 API를 사용 가능 |
| Android | Android 2.0이상버전에서 네이버 지도 API를 사용 가능 |

NAVER

# 준비 사항

- 안드로이드 네이버 지도 라이브러리용 API 키 발급
- 안드로이드용 네이버 지도 라이브러리 (nmaps.jar) 추가

NAVER

# API KEY 발급

NAVER

# 발급 화면

## 키 등록/관리

이 곳은, 오픈 API 를 이용해 새로운 사이트 및 프로그램을 개발하시기 위한 키(Key)를 발급받고 확인하시는 곳입니다.
검색, 지도 API의 키는 일정 쿼리 내에서 바로 키 발급이 가능합니다.

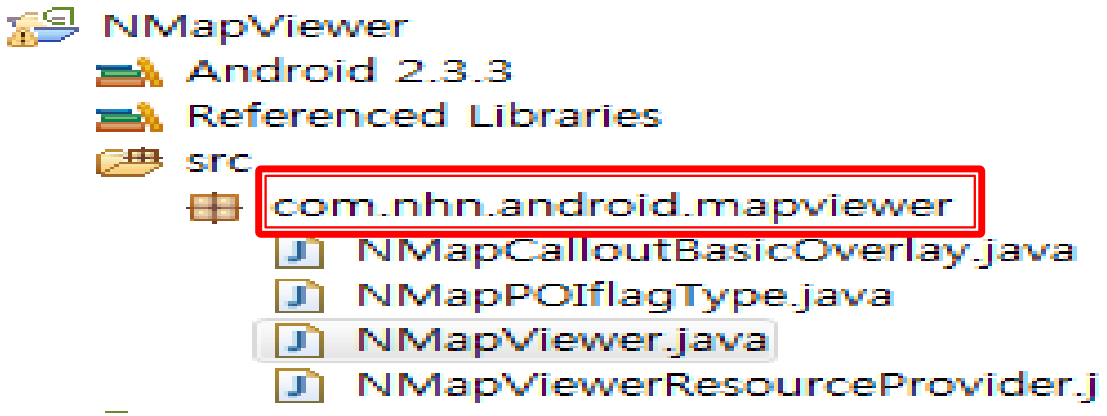
| API 명 | 발급정보 | 발급일 | 오늘 사용쿼리 | 키 추가/삭제 |
|---|---|---|---|---|
| 검색API | [키 추가]를 통해 새로운 키 발급이 가능합니다. | | | 키 추가 |
| 지도API | 환경 안드로이드<br>Package Name com.nhn.android.mapviewer/<br>발급키 9fbcb3b9cf20b0873aa80479e20df39a | 2012.04.05 | 3 / 5000 | 삭제 |
| | [키 추가]를 통해 새로운 키 발급이 가능합니다. | | | 키 추가 |
| 단축 URL API | [키 추가]를 통해 새로운 키 발급이 가능합니다. | | | 키 추가 |

NAVER

# 주의점

- 발급할 때의 Package 이름과 API를 사용할 프로젝트의 Package 이름, Manifest.xml의 Package이 동일해야 함

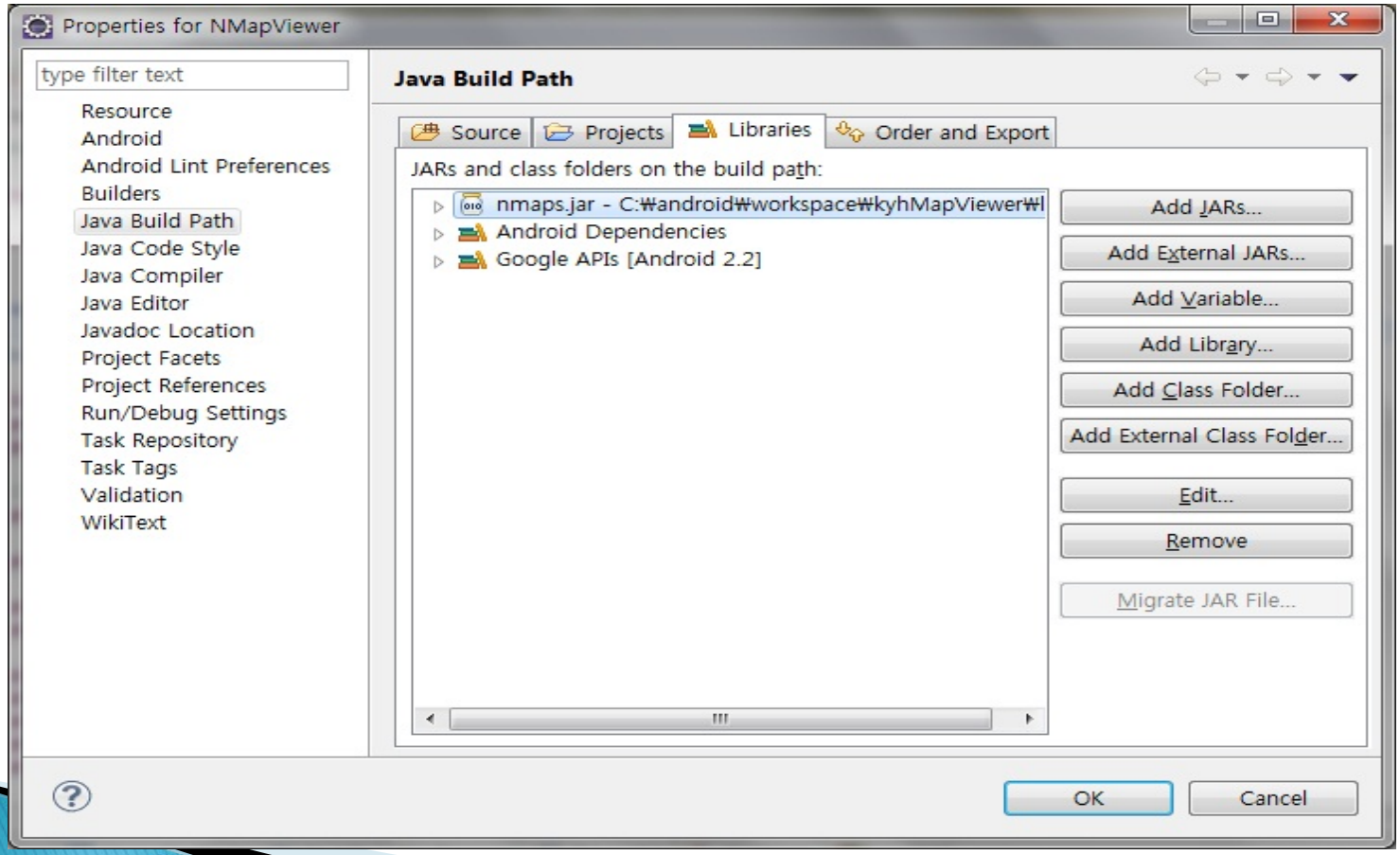
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.nhn.android.mapviewer"
    android:versionCode="1"
    android:versionName="1.2.9">
```



NAVER

# Library 추가

Project -> Properities -> Java Build Path
-> Add External JARs… -> nmaps.jar




NAVER

# Permission

지도 라이브러리에서 지도 파일 다운로드 및 서버 API 연동을 위해서 네트워크 연결이 필요

AndroidManifest 파일에 아래와 같이 네트워크 접근 권한을 설정

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.kyh.mapviewer"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="8" />
    <uses-permission android:name="android.permission.INTERNET"/>

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <activity
            android:name=".KyhMapViewerActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Manifest | Application | Permissions | Instrumentation | AndroidManifest.xml

NAVER

# 지도 생성

- 주요 클래스

| Class Name | Description |
| --- | --- |
| NMapActivity | 안드로이드 Activity 클래스를 상속받은 클래스로서 NMapView 객체를 표시<br>Activity 라이프 사이클에 따라서 내부적으로 지도 데이터를 관리 |
| NMapView | 안드로이드 ViewGroup 클래스를 상속받은 클래스로서 지도 데이터를 화면에 표시 |

NAVER

# 지도 생성

- NMapView 클래스를 포함하는 Activity는 반드시 NMapActivity클래스를 상속받아야 함
- 해당 Activity는 하나의 NMapView 클래스만 포함
- NMapActivity클래스를 상속받은 Activity의 onCreate() 메서드에서 NMapView객체를 생성
- NMapView 객체 생성후 발급 받은 API키를 등록

NAVER

# 지도 생성

```
public class NMapViewer extends NMapActivity {

    private static final String API_KEY = "9fbcb3b9cf20b0873aa80479e20df39";
    private NMapView mMapView;

    public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);

            // create map view
            mMapView = new NMapView(this);

            // set a registered API key for Open MapViewer Library
            mMapView.setApiKey(API_KEY);
            setContentView(mMapContainerView);

            // initialize map view
            mMapView.setClickable(true);
            …
    }
…
}
```

NAVER

# 지도 초기화 콜백 인터페이스

- 지도 초기화가 완료되면 콜백 인터페이스 호출
- 초기화 성공 시
  - 지도 보기 모드 및 중심 좌표 등을 설정
- 초기화 실패 시
  - 적절한 예외 처리를 수행함

NAVER

# 지도 초기화 콜백 인터페이스

```
public void onMapInitHandler
(NMapView mapView, NMapError errorInfo) {
   if (errorInfo == null) { // success
         mMapController.setMapCenter(new
   NGeoPoint(126.978371, 37.5666091), 11);
   }
   else { // fail
         Log.e(LOG_TAG, "onMapInitHandler: error=" +
   errInfo.toString());
   }
}
```

NAVER

# 지도 생성

오전 9:12
SK본사
동아
미디어센터
화면세점
한국금융사
박물관
서울
파이낸스센터
서울특별시
의회
서울신문사
농협중앙회
서울시청
서울광장
호텔
프레지던트
대한문
플라자호텔
웨스틴
조선호텔
NAVER

NAVER

# 현재 위치 표시

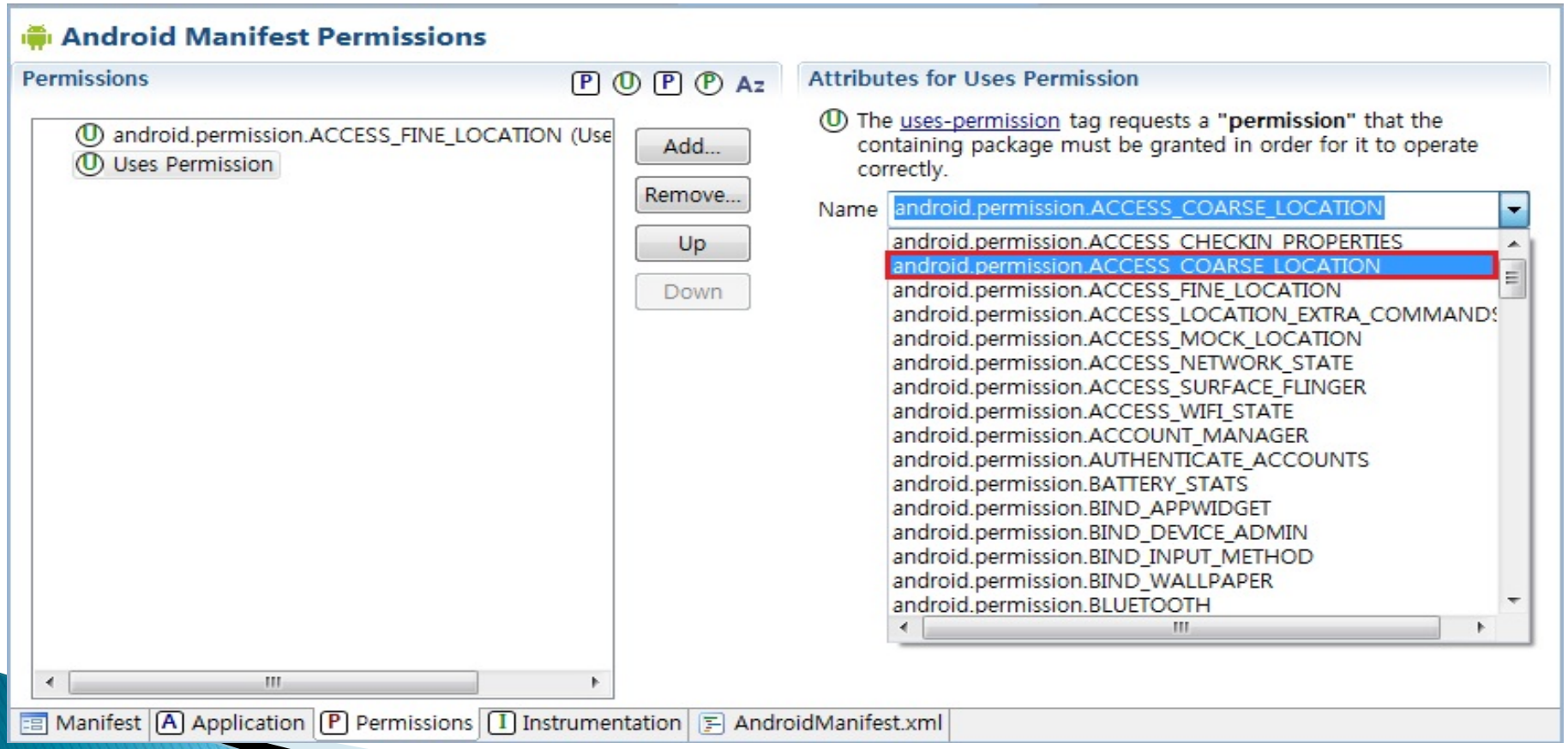
- 주요 클래스

| Class Name | Description |
| --- | --- |
| NMapMyLocationOverlay | 지도 위에 현재 위치를 표시하는 오버레이 클래스<br>현재 위치 관리자와 나침반 관리자로 구성 |

# 현재 위치 표시

현재 위치 표시 기능을 사용하기 위해서 접근 권한을 설정

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




NAVER

# 현재 위치 표시

현재 위치 표시와 나침반에 의한 지도 회전 기능을 구현하기 위해 NMapMyLocationOverlay객체를 생성

```
// 현재위치 manager
mMapLocationManager = new NMapLocationManager(this);
mMapLocationManager.setOnLocationChangeListener(onMyLocationChangeListener);

// 나침반 manager
mMapCompassManager = new NMapCompassManager(this);

// create my location overlay
mMyLocationOverlay =
mOverlayManager.createMyLocationOverlay(mMapLocationManager,
mMapCompassManager);
```

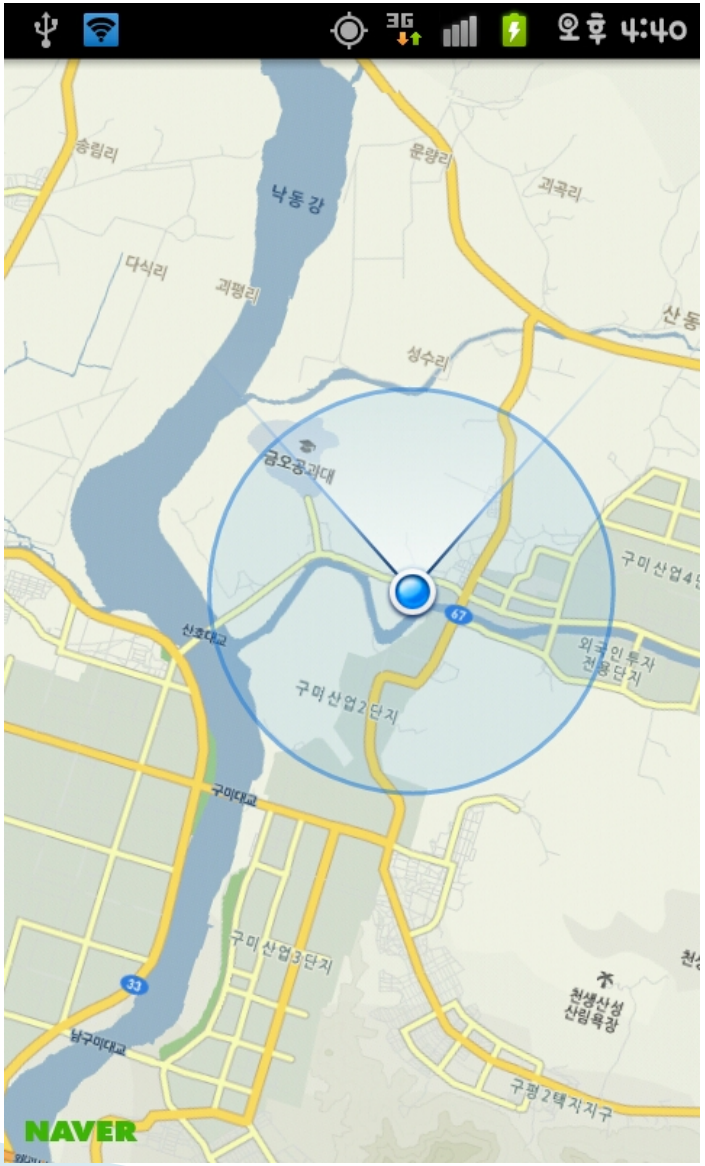
NAVER

# 현재 위치 표시

**onLocationChangeListener**
단말기의 현재 위치 상태 변경 시 호출되는 콜백 인터페이스를 정의

| 구문 | 설명 |
| --- | --- |
| boolean onLocationChanged(NMapLocationManager locationManager, NGeoPoint myLocation) | 현재 위치 변경 시 호출<br>myLocation객체에 변경된 좌표가 전달<br>현재 위치를 계속 탐색하려면 true를 반환 |
| void onLocationUpdateTimeout(NMapLocationManager locationManager) | 정해진 시간 내에 현재 위치 탐색 실패 시 호출 |

# 현재 위치 표시




NAVER

# 오버레이 아이템 표시

- 주요 클래스

| Class Name | Description |
| --- | --- |
| NMapOverlayManager | 지도 위에 표시되는 오버레이 객체를 관리 |
| NMapResourceProvider | 지도 위의 오버레이 객체 드로잉에 필요한 리소스 데이터를 제공하기 위한 추상 클래스 |
| NMapPOIdataOverlay | 여러 개의 오버레이 아이템을 포함할 수 있는 오버레이 클래스<br>그룹 오버레이 아이템을 효과적으로 처리할 수 있는 기능을 제공 |
| NMapCalloutOverlay | 지도 위의 오버레이 아이템 선택 시 표시되는 말풍선 오버레이의 추상 클래스 |

NAVER

# 오버레이 아이템 표시

- 지도 위에 표시되는 오버레이 객체들을 관리하는 NMapOverlayManager 객체 생성
- 객체 생성 시에는 화면에 표시될 오버레이 객체에 대한 이미지 리소스를 제공하기 위해 NMapResourceProvider 클래스를 상속받은 객체를 전달
- NMapResourceProvider 클래스는 지도 위의 오버레이 객체 드로잉에 필요한 리소스 데이터를 제공하기 위한 추상 클래스

NAVER

# 오버레이 아이템 표시

```
// create resource provider
mMapViewerResourceProvider =
 new NMapViewerResourceProvider(this);

// create overlay manager
mOverlayManager =
 new NMapOverlayManager(this, mMapView,
  mMapViewerResourceProvider);
```

NAVER

# 오버레이 아이템 표시

- 여러 개의 오버레이 아이템을 하나의 오버레이 객체에서 관리하기 위한 오버레이 클래스 NMapPOIdataOverlay의 객체 생성

```
NMapPOIdata poiData = new NMapPOIdata(2,
    mMapViewerResourceProvider);

poiData.beginPOIdata(2);
poiData.addPOIitem(127.0630205, 37.5091300, "Pizza 777-
    111", markerId, 0);
poiData.addPOIitem(127.061, 37.51, "Pizza 123-456",
    markerId, 0);
poiData.endPOIdata();

// create POI data overlay
NMapPOIdataOverlay poiDataOverlay =
    mOverlayManager.createPOIdataOverlay(poiData, null);
```

NAVER

# 오버레이 아이템 표시

NAVER

# 오버레이 아이템 표시(썸네일)

NMapViewer class

```
//maker클릭시 호출되는 메소드
private final NMapOverlayManager.OnCalloutOverlayListener
   onCalloutOverlayListener = new
   NMapOverlayManager.OnCalloutOverlayListener()
{
   public NMapCalloutOverlay
         onCreateCalloutOverlay(NMapOverlay itemOverlay,
         NMapOverlayItem overlayItem,
         Rect itemBounds)
    {
         …
         return new NMapCalloutBasicOverlay(NMapViewer.this,
         itemOverlay, overlayItem, itemBounds);
   }
}
```

NAVER

# 오버레이 아이템 표시(썸네일)

NMapCalloutBasicOverlay class

```
protected void drawCallout(Canvas canvas, NMapView
   mapView, boolean shadow, long when) {
   …
   Paint pnt = new Paint();
   BitmapDrawable b1 =
   (BitmapDrawable)mContext.getResources().getDrawa
   ble(R.drawable.moon);
   Bitmap bit1 = b1.getBitmap();
   canvas.drawBitmap(bit1, mTempRectF.left,
   mTempRectF.top-60, pnt);
}
```

NAVER

# 오버레이 아이템 표시(썸네일)

NAVER

# 오버레이 아이템 변경(썸네일)

NMapViewer class

```
private void testPOIdataOverlay() {
    int markerId = NMapPOIflagType.PIN;

    NMapPOIdata poiData = new NMapPOIdata(2, mMapViewerResourceProvider);
    poiData.beginPOIdata(2);
    poiData.addPOIitem(127.0630205, 37.5091300, "Pizza 777-111", markerId, 0);
    poiData.endPOIdata();

    NMapPOIdataOverlay poiDataOverlay =
    mOverlayManager.createPOIdataOverlay(poiData, null);

    poiDataOverlay.setOnStateChangeListener(onPOIdataStateChangeListener);

    poiDataOverlay.selectPOIitem(0, true);
}
```

NAVER

# 오버레이 아이템 변경(썸네일)

NMapPOIflagType class

```
public class NMapPOIflagType {
    public static final int UNKNOWN = 0x0000;

    // Single POI icons
    Private static final int SINGLE_POI_BASE = 0x0100;

    // Spot, Pin icons
    public static final int SPOT = SINGLE_POI_BASE + 1;
    public static final int PIN = SPOT + 1;
    ....
}
```

NAVER

# 오버레이 아이템 변경(썸네일)

NMapViewerResourceProvider class

```
public class NMapViewerResourceProvider extends NMapResourceProvider {

    private class ResourceIdsOnMap {
        int markerId;
        int resourceId;
        int resourceIdFocused;

        ResourceIdsOnMap(int markerId, int resourceId, int
        resourceIdFocused) {
            this.markerId = markerId;
            this.resourceId = resourceId;
            this.resourceIdFocused = resourceIdFocused;
        }
    }
}
```

NAVER

# 오버레이 아이템 변경(썸네일)

NMapViewerResourceProvider class

```
private final ResourceIdsOnMap mResourceIdsForMarkerOnMap[] = {
// Spot, Pin icons
    new ResourceIdsOnMap(NMapPOIflagType.PIN, R.drawable.iu,
            R.drawable.iu),
    new ResourceIdsOnMap(NMapPOIflagType.SPOT, R.drawable.iu,
            R.drawable.iu),

    // Direction POI icons: From, To
    new ResourceIdsOnMap(NMapPOIflagType.FROM,
            R.drawable.ic_map_start, R.drawable.ic_map_start_over),
    new ResourceIdsOnMap(NMapPOIflagType.TO,
            R.drawable.ic_map_arrive, R.drawable.ic_map_arrive_over),
};
```

NAVER

# 오버레이 아이템 변경(썸네일)

**Focusing 전**

**Focusing 후**

NAVER

# Q&A

NAVER