|  |
| --- |
| **FATOS Software Development Kit**  **V3.0.0** |

Standard Article/Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Standard article/revision date** | **Details** | **Written by** | **Approved by** |
| 3.0.0 | 2019. 12. 02 | V3.0.0 First Draft released | K.I. Woo | H.J. Park |

Contents

1. GET FATOS SDK Code 9

2. Setting up a development project 9

3. Getting and applying API key 12

4. FATOS SDK for Android feature configuration 13

5. uNDERSTANDING CODE 13

5.1 Draw a map 13

5.1.1 CREATE Map Fragment 13

5.1.2 Draw a map on user activity 14

5.1.3 Move the map 16

5.1.4 Changing map mode 17

5.1.5 Move to the current location 18

5.1.6 Map Object Interface 18

void onSetMapStatus(int nStatus); // 0 : Driving, 1: Route summary 19

5.2 Search 19

5.2.1 POI(Point Of Interest) SEARCH 19

5.2.2 Search results 20

5.3 Explore the path 23

5.3.1 Explore the path based on the current location 23

5.3.2 Explore the origin settings path 25

5.4 About the path 26

5.4.1 Path summary 26

5.4.2 Route details 27

5.5 Directions 31

5.5.1 Start directions 31

5.5.2 About the path 32

6. API list 33

6.1 Initialization process 33

6.1.1 public int initFatosNaviEngine() 33

6.1.2 public int FM\_Init(Context context, String strApiKey) 34

6.1.3 public int FM\_Destory() 34

6.2 Map 34

6.2.1 MapStatusListener 34

6.2.2 public void setMainMapOption(boolean bDrawStartImg,boolean bDrawGoalImg,boolean bDrawPinImg,boolean bDrawEVWare) 35

6.2.3 public void onMapReady() 35

6.2.4 public void FM\_MoveCurPosition(FMBaseActivity.OnFatosMapListener onFatosMapListener) 36

6.2.5 public void FM\_SetMapAirMode(int nMapType) 36

6.2.6 public void FM\_EnableSmartZoomInOut(Boolean bEnable) 36

6.2.7 public void FM\_EnableWalkingMode(Boolean bEnable) 36

6.2.8 public int FM\_GetAddress(Handler \_resultHandler, double xlon, double ylat) 37

6.2.9 public int FM\_GetPosFromScreen(int x, int y, double[] xlonlat) 37

6.2.10 public ArrayList<GPSPOS> FM\_GetLastGPSTrace() 37

6.2.11 public void FM\_DrawGrid(double[] x, double[] y) 38

6.2.12 public void FM\_SetMapCenter(float hCenter, float vCenter) 38

6.2.13 public void FM\_AutoSetMapCenter(boolean bMapMove) 38

6.2.14 public void FM\_SetMapPosition(FMBaseActivity.OnFatosMapListener onFatosMapListener, double xlon, double ylat) 38

6.2.15 public void FM\_GetTbtImageRes(int nTbtCode) 39

6.2.16 public void FM\_SetNaviVolume(int volume) 39

6.2.17 public void FM\_IsPlaying() 39

6.2.18 public void FM\_GetMapVersion(NaviVersion versionInfo) 39

6.2.19 public class NaviVersion 40

6.2.20 public class DBVersion 40

6.2.21 public void FM\_RequestUpdate(int nDownloadGubun, long freeSpace, final MapUpdateListenerCallback mapUpdateListener) 41

6.2.22 public interface MapUpdateListenerCallback 41

6.2.23 public void FM\_MapUpdateCancel() 41

6.2.24 41

6.3 rOUTE SEARCH 42

6.3.1 public void FM\_StartRGService(FMBaseActivity.OnFatosMapListener onFatosMapListener) 42

6.3.2 public ArrayList<RouteSummaryData> FM\_RouteSummary(FMBaseActivity.OnFatosMapListener onFatosMapListener) 42

6.3.3 public void FM\_SelectRoute(int nType) 42

6.3.4 public class RouteSummaryData 42

6.3.5 public ArrayList<RouteSummaryDataDetail> FM\_RouteSummaryDetail(int nType, int nFromIdx) 43

6.3.6 public class RouteSummaryDataDetail 44

6.3.7 public void FM\_CancelRoute() 44

6.3.8 public void FM\_SetSimulationSpeed(int nSpeed) 45

6.3.9 public List<RoutePosition>FM\_GetViaPOIList() 45

6.3.10 public void FM\_RouteVol2\_Via(Handler \_resultHandler, List<RoutePosition> m\_PoiList) 45

6.3.11 public class RoutePosition 46

6.3.12 public Boolean FM\_GoNextGoal() 47

6.3.13 public void FM\_StartSimulation () 47

6.3.14 public void FM\_StopSimulation () 47

6.3.15 public void FM\_GetMapCenterPos(double[] lonlat) 47

6.3.16 public void FM\_GetSubMapCenterPos(double[] lonlat) 47

6.3.17 48

6.4 POI SEARCH 48

6.4.1 public int FM\_SearchPOI(Handler \_resultHandler,String strSearchName, boolean bSaveRecent) 48

6.4.2 public int FM\_RecommendWord(Handler \_resultHandler,String strSearchName) 48

6.4.3 public ArrayList<POIItem> FM\_GetSearchResult(int nSortOption) 48

6.4.4 public ArrayList<String> FM\_GetRecommentWordResult() 49

6.4.5 49

6.5 49

# GET FATOS SDK CODE

Please duplicate or download THE FATOS SDK for Android from GitHub.

# Setting up a development project

Download and [install](https://developer.android.com/studio/install.html?pkg=studio&hl=ko)AndroidStudio.  
(<https://developer.android.com/studio/index.html?hl=ko)>

The FATOS SDK supports Android 5.0 (API Level 20 and above).

It is distributed in form of FATOS SDK for Androids AAR(Android Archive Package) File. Therefore, you should link the AAR file to the project.

Link aar file at libs folder into app/build.gradle   
(Based on GitHub's [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest))

|  |
| --- |
| Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest)  repositories {  flatDir {  dirs 'libs'  }  }  dependencies {  implementation fileTree(dir: 'libs', include: ['\*.jar'])  implementation(name:'fire\_sdk\_vX.X.X', ext:'aar') |

The FATOS SDK for Androidmust set up three essential permissions:

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviSplashActivity.java)  @Override  protected void onCreate(Bundle savedInstanceState) {  super.onCreate(savedInstanceState);  if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M) {  int permissionResultWRITE\_EXTERNAL\_STORAGE = checkSelfPermission(Manifest.permission.WRITE\_EXTERNAL\_STORAGE);  int permissionResultACCESS\_FINE\_LOCATION = checkSelfPermission(Manifest.permission.ACCESS\_FINE\_LOCATION);  int permissionResultACCESS\_READ\_PHONE= checkSelfPermission(Manifest.permission.READ\_PHONE\_STATE);  if (permissionResultWRITE\_EXTERNAL\_STORAGE == PackageManager.PERMISSION\_DENIED  || permissionResultACCESS\_FINE\_LOCATION == PackageManager.PERMISSION\_DENIED  || permissionResultACCESS\_READ\_PHONE == PackageManager.PERMISSION\_DENIED) {  requestPermissions(new String[]{Manifest.permission.WRITE\_EXTERNAL\_STORAGE, Manifest.permission.ACCESS\_FINE\_LOCATION, Manifest.permission.READ\_PHONE\_STATE}, 1000);  }  else {  m\_Handler.postDelayed(startMainActivity, 10);  getWindow().setFlags(WindowManager.LayoutParams.FLAG\_FULLSCREEN,  WindowManager.LayoutParams.FLAG\_FULLSCREEN);  }  }  else {  m\_Handler.postDelayed(startMainActivity, 10);  getWindow().setFlags(WindowManager.LayoutParams.FLAG\_FULLSCREEN,  WindowManager.LayoutParams.FLAG\_FULLSCREEN);  }  } |

To use FATOS SDK for Android,Project's Application Classand MainActivitymust inherit the ANaviApplication Classand FMBaseActivity Classfrom the FATOS SDK, respectively. This allows for smooth directions.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviApplication.java, TNaviMainActivity.java)  @Override  public class TNaviApplication extends ANaviApplication {  private Context m\_Context;  @Override  public void onCreate() {    public class TNaviMainActivity extends FMBaseActivity { |

FaTOS SDK forAndroid must initiate the Navi Engine. Itis recommended that you do FMBaseActivity at the timeof onCreate() of The Inherited MainActivity.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviMainActivity.java)  //==============================================================================================  private void InitProcess() throws IOException {  m\_Context = this;  m\_gApp = (ANaviApplication) m\_Context.getApplicationContext();  m\_route = m\_gApp.getRouteApiInstance();  FMInterface.CreateInstance(m\_Context);  m\_FMInterface = FMInterface.GetInstance();  m\_iEngineInit = initFatosNaviEngine();  if(m\_iEngineInit != 1)  {  Intent intent = new Intent();  intent.setAction(TNaviActionCode.READY\_MAIN\_MAP); // Action name  sendBroadcast(intent);  return;  } |

# Getting and applying API key

API key request

Send email to [dev@fatoscorp.com](mailto:dev@fatoscorp.com) to get the API key

Applying API key

FMInterface.initKey(.) at the time of onCreate() with api key issued by Application Class It must be set through a function.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviApplication.java) public class TNaviApplication extends ANaviApplication {  private Context m\_Context;    @Override  public void onCreate() {  m\_Context = this;  FMInterface.initKey(m\_Context, "**FATOS SDK API key**");  super.onCreate();  ...  ... |

# FATOS SDK for Android feature configuration

**Map**

Provide Fatos Map Fragment

Provide callback for Map Event, Status information etc

Allows you to use Fatos Map flexibly.

Map Option features available.

Provide Map Control API for map mode changes, coordinate movements, Zoom In/Out, Map Picker, and more.

**Search**

FMS v1.01 provides T map search results

**Explore the path**

FMS v1.01 provides T map path navigation results

Provides path navigation data management

**Directions**

Provides a pathway horizontal and vertical Fragment and custom fragment interlocking functions.

**Provide Interface & Base Class**

Map, Route Guidance-related features and controls are provided through The Base Class toreduce redundancy and make navigation functionality easier to add.

# Understanding the code

# Draw a map

# Create map fragments

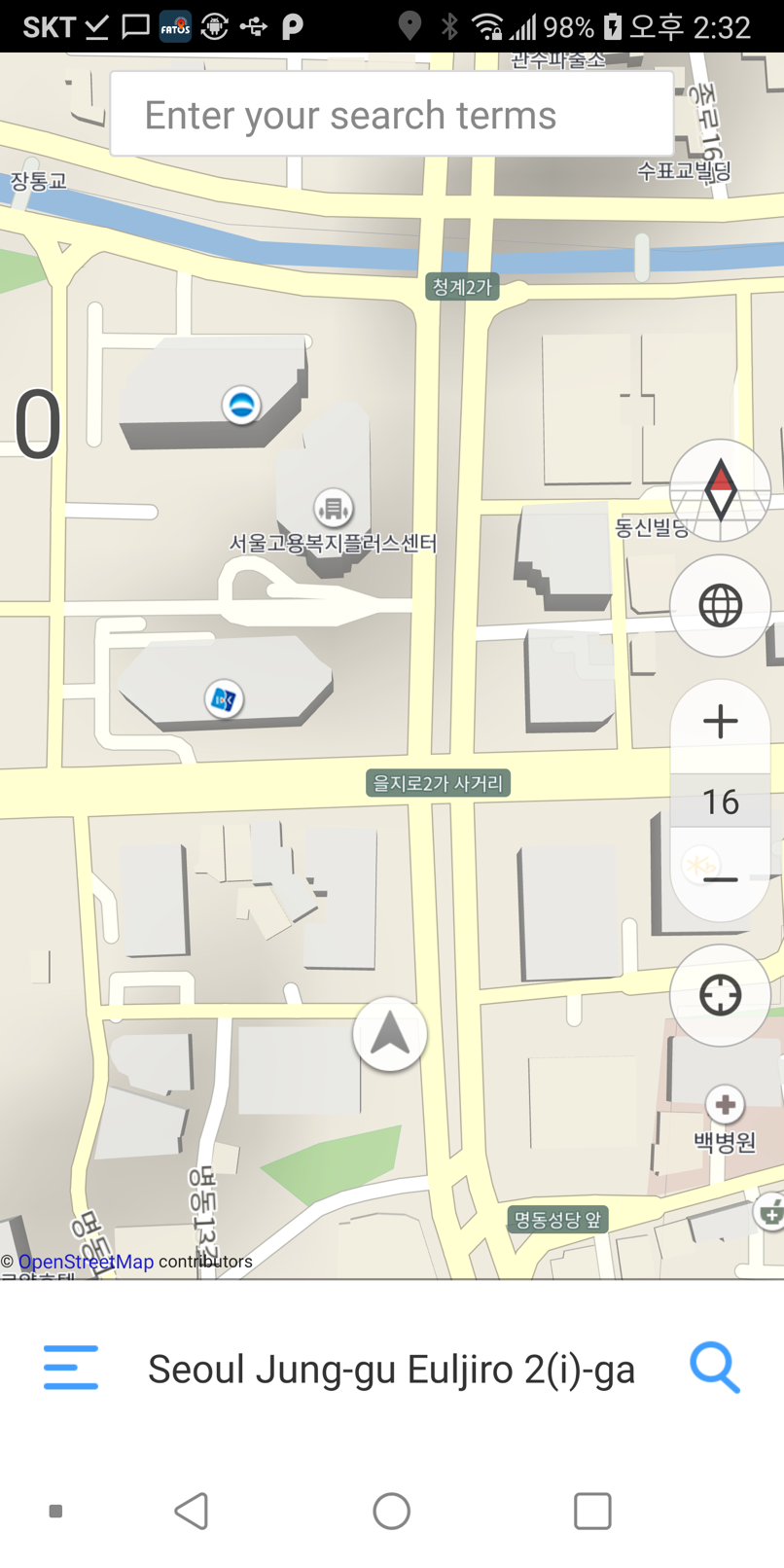
In order to display the FATOS map in the app, you need to create an inherited Fragment from AMapBaseFragment.   
Features such as Map Control and Map Setting are operated in the Fragment.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (FatosMapFragment.java)  public class FatosMapFragment extends AMapBaseFragment  implements TNaviMainActivity.OnFatosMapListener,  FatosMainMapView.OnFatosMapListener {  …  …  } |

# Draw a map on user activity

When you add the created Map Fragment to the activity you want to draw, the FATOS map is drawn.

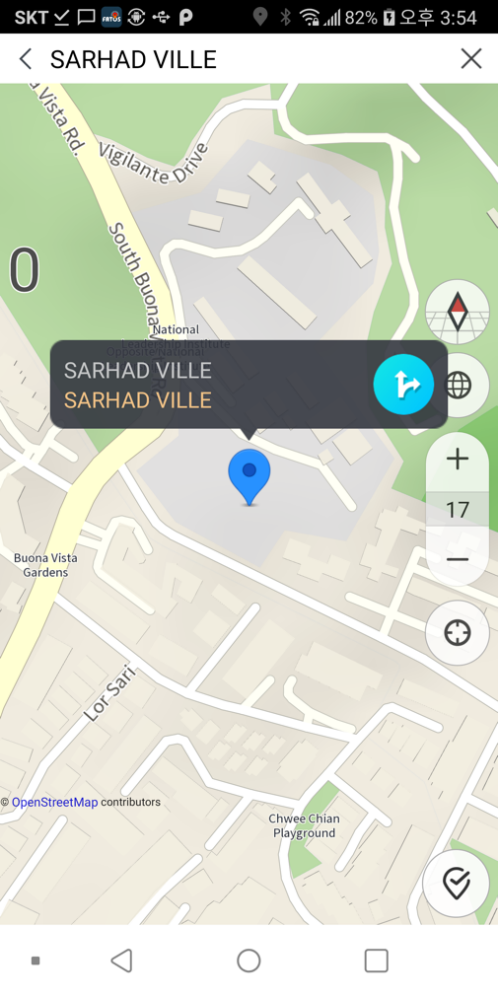
|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (activity\_tnavi\_main.xml)  <?xml version="1.0" encoding="utf-8"?>  <biz.fatossdk.exlib.fatoslayout.AMapSCRelativeLayout  xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:id="@+id/rl">  <biz.fatossdk.exlib.fatoslayout.AMapSCLinearLayout  android:orientation="vertical"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  >  <biz.fatossdk.exlib.fatoslayout.AMapSCRelativeLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <fragment  android:id="@+id/map\_fragment"  **android:name="kr.fatos.tnavi.tnavifragment.FatosMapFragment"**  android:layout\_width="match\_parent"  android:layout\_height="match\_parent" /> |



# Move the map

WgS84 coordinates can be passed to move the map to that location.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviMainActivity.java)  //Move coordinates in map view  public void MovePoint(double dX, double dY, String i\_strAddr)  {  m\_FMInterface.FM\_SetMapPosition(0, dX, dY, 16.f);  fragmentManager = getSupportFragmentManager();  searchShowMapFragment = (SearchShowMapFragment)fragmentManager.findFragmentById(R.id.container);  } |



# Changing map mode

You can change and set the map Object draw, such as air / satellite, building, road, etc. as well as the map direction mode such as 2D, 3D, North up.

m\_FMInterface.FM\_SetMapAirMode(FMInterface.MAPMODE\_AIR\_OFF\_BUILDING\_ON); // Air mode off building on

public static final int MAPMODE\_AIR\_ON\_BUILDING\_ON = 0; // air mode on, building on

public static final int MAPMODE\_AIR\_ON\_BUILDING\_OFF = 1; // air mode on, building off

public static final int MAPMODE\_AIR\_OFF\_BUILDING\_ON = 2; // air mode off, building on



# Move to the current position

Provides the ability to move to gps (or last receiving location) coordinates while moving the map.

|  |  |
| --- | --- |
| Description | go to the current position |
| Example | m\_FMInterface.FM\_MoveCurPosition(FMBaseActivity.OnFatosMapListener) |

# Map Object Interface

For map functions such as Map Flicking and Two Touch Zoom In/out to work, it must be a functional definition for the callback function.

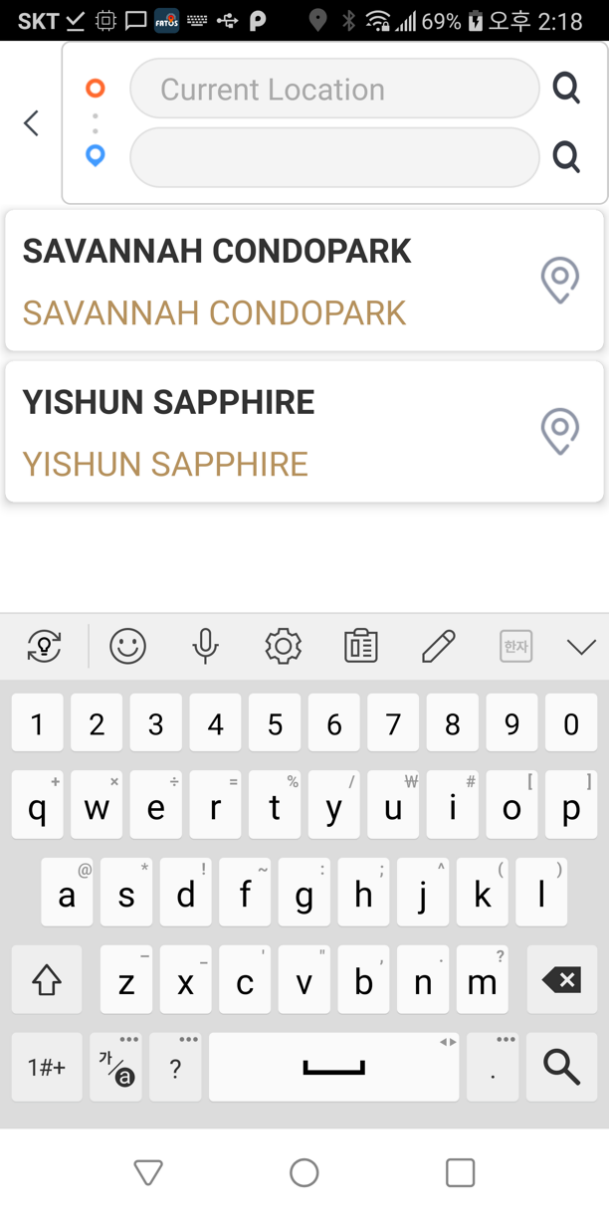
|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (FatosMapFragment.java)  public interface OnFatosMapListener {  void onCustomZoomInOutEnd(boolean bEnd);  void moveMapCurrentPos();  void mapMoveAniReset();  void mapMoveDirectCurPos();  void mapCenterUpdate();  void setMapMode();  void onMapAnimation(MapAnimation aniInfo);  void onCustomZoomInOut();  void onMapLevelInOut(float fLevel);  void onMapLevelInOut(float fLevel, float fTilt);  void onMapDrawPinImg(double x, double y, int nPinType); void onSetMapStatus(int nStatus); // 0 : driving , 1: route summary } |

# Search

# POI(Point of Interest) search

Search by all categories, depending on the API key property issued.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviMainActivity.java)  public void getSearchPoiItem(String strMsg)  {  ArrayList<String> arInfo = new ArrayList<>();  arInfo.add(strMsg);  int index = prefs.getInt(SettingsCode.getKeyIndex(), 0);  String[] countrycodes;  countrycodes = getResources().getStringArray(R.array.wecountry\_codes);  arInfo.add(countrycodes[0]);  arInfo.add(String.valueOf(AMapPositionManager.getCurrentLonX()));  arInfo.add(String.valueOf(AMapPositionManager.getCurrentLatY()));  m\_FMInterface.FM\_SearchPOIForTNavi(new HttpResultHandler(TNaviMainActivity.this), arInfo, false);  } |



# Search results

The results of the POI retrieved through the FM\_SearchPOIForTNavi are passed through the Pass handler.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviMainActivity.java)  private void handleMessage(Message msg) {  String result = msg.getData().getString(AMapGoogleSearchUtil.RESULT);  ArrayList<String> searchList = new ArrayList<String>();  if (result.equals(FMError.FME\_SUCCESS\_SEARCH\_SUCCESS)) {  searchResultPOI = m\_FMInterface.FM\_GetSearchResult(FMSortOption.FM\_SORT\_BY\_DIST);  if(searchResultPOI != null)  {  SearchFragment sf = (SearchFragment) getSupportFragmentManager().findFragmentById(R.id.container);  sf.MergePOIItem(searchResultPOI, result);  } |

The information of the POI being retrieved. Depending on the POI, there may be items that exist and items that do not exist. However, coordinates and names are mandatory.

|  |
| --- |
| public class POIItem {  // poi data items  public String id = ""; // poi id  public String name = ""; // poi name  public String telNo; // poi telephone  public String frontLat = null; // Entrance point coordinates  public String frontLon = null; // Entrance point coordinates  public String noorLat; // Center coordinates  public String noorLon; // Center coordinates  …  …  public int nFromCurPosDist; // Distance from the current location to that location(meter Unit)    public String secondBuildNo;  public String radius;  public String bizName;  public String upperBizName;  public String middleBizName;  public String lowerBizName;  public String detailBizName;  public String rpFlag = "";  public String parkFlag;  public String detailInfoFlag;  public String desc; |

# Explore the path

# Explore the path based on the current location

The FM\_RouteVol2\_Via api provides path navigation based on the current location.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviMainActivity.java)  //Navigate to your destination  public void routeTovia(double x, double y, String flag, @Nullable NPoiItem i\_PoiItem){  show\_ProgressDialog(R.string.string\_higetting\_search\_direction , false);  if(i\_PoiItem != null) {  m\_RecentPOIItem = i\_PoiItem;  if (i\_PoiItem.getAddressFull() == null){  i\_PoiItem.setAdminLevel1LocalName("");  }  if(i\_PoiItem.getEnglishName() == null){  i\_PoiItem.setEnglishName("");  }  if(i\_PoiItem.getAddressFull().equals("") || i\_PoiItem.getEnglishName().equals("")){  String address\_wgs = String.format("%.5f",i\_PoiItem.getLocationPointX()) + ", " +  String.format("%.5f",i\_PoiItem.getLocationPointY());  m\_RecentPOIItem.setAdminLevel1EnglishName("No Address Name");  m\_RecentPOIItem.setEnglishName(address\_wgs);  }  }  m\_FMInterface.FM\_GetViaPOIList().clear();  List<RoutePosition> positionList = m\_FMInterface.FM\_GetViaPOIList();  RoutePosition positionList0 = new RoutePosition();  RoutePosition positionList1 = new RoutePosition();  switch(flag){  case TNaviActionCode.JUST\_GOAL :  positionList.clear();  positionList0.x = this.m\_dStartCoord[0];  positionList0.y = this.m\_dStartCoord[1];  positionList0.name = this.strAddr[0];  positionList.add(positionList0);  positionList1.x = x;  positionList1.y = y;  positionList1.name = this.strAddr[1];  if(this.strAddr[1].equals("")){  positionList1.name = "Goal";  }  positionList1.bPassingPoint = false;  positionList.add(positionList1);  saved\_data.clearData();  saved\_data.viaX = positionList0.x;  saved\_data.viaY = positionList0.y;  saved\_data.goalX = x;  saved\_data.goalY = y;  break;  ...  ...  }  m\_FMInterface.FM\_RouteVol2\_Via(new HttpResultHandler(TNaviMainActivity.this), positionList);  } |

|  |
| --- |
| public void FM\_RouteVol3\_Via(Handler \_resultHandler, List<RoutePosition> m\_PoiList, int routeoption)  Parameter & Return \*  @param  resultHandler : route result handler  PoiList : start/goal setting  routeoption :  public static final int ROUTE\_OPTION1\_Fastest= 1;  // Fastest    public static final int ROUTE\_OPTION5\_Shortest = 16;  // Shortest    public static final int ROUTE\_OPTION6\_ Shortest = 32;  // Fastest noToll    public static final int ROUTE\_OPTION7\_Shortest\_noToll = 256;  // Shortest noToll  설명 Ex)  m\_FMInterface.FM\_RouteVol3\_Via(new  HttpResultHandler(TNaviMainActivity.this),  positionList, FMInterface. ROUTE\_OPTION5\_Shortest +FMInterface. ROUTE\_OPTION6\_ Shortest +FMInterface. ROUTE\_OPTION7\_Shortest\_noToll); |

# Explore the origin settings path

You can request to navigate the route from that origin to your destination by setting the coordinates of your origin, not the GPS current location.

|  |
| --- |
| public void setStartCoord(double i\_dX, double i\_dY)  {  if(i\_dX <= 0 || i\_dY <= 0)  {  this.m\_dStartCoord[0] = m\_FMInterface.FM\_GetDriveInfo().getM\_fCurLonX();  this.m\_dStartCoord[1] = m\_FMInterface.FM\_GetDriveInfo().getM\_fCurLatY();  }  else {  this.m\_dStartCoord[0] = i\_dX;  this.m\_dStartCoord[1] = i\_dY;  }  }  …  …  positionList.clear();  positionList0.x = this.m\_dStartCoord[0];  positionList0.y = this.m\_dStartCoord[1];  positionList0.name = this.strAddr[0];  positionList.add(positionList0);  m\_FMInterface.FM\_RouteVol2\_Via(new HttpResultHandler(TNaviMainActivity.this), positionList); |

# About the path

# Path summary

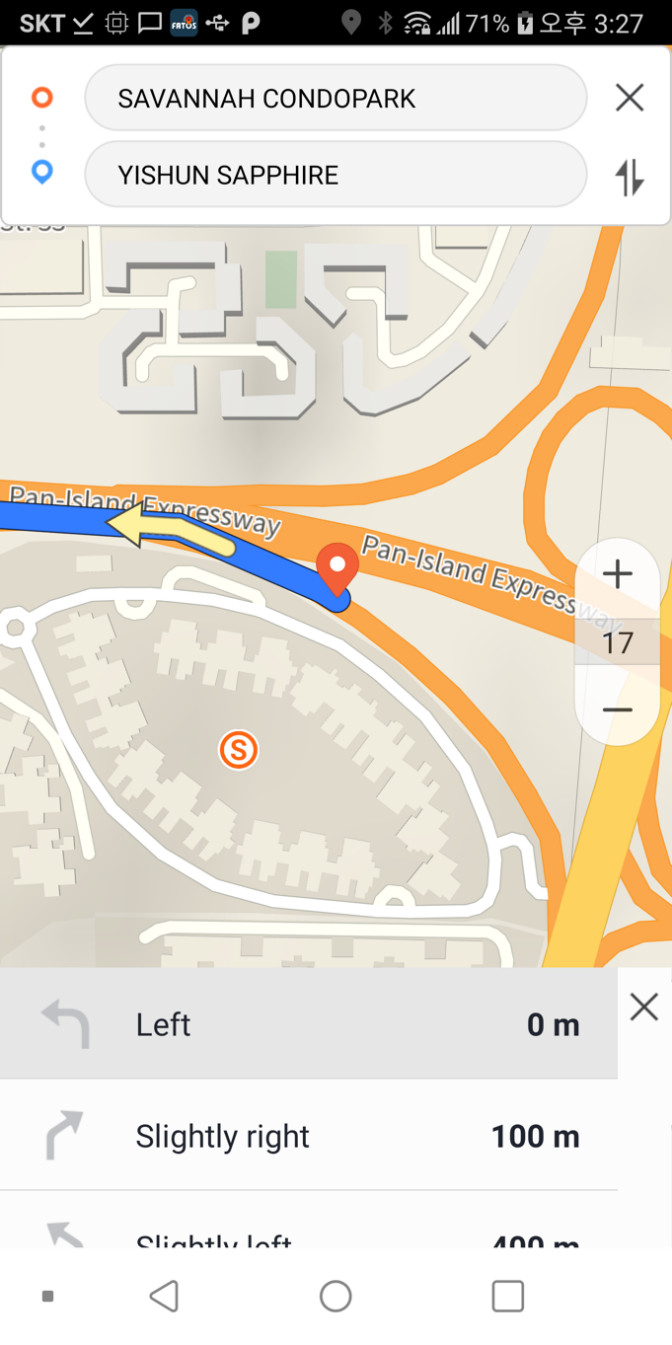
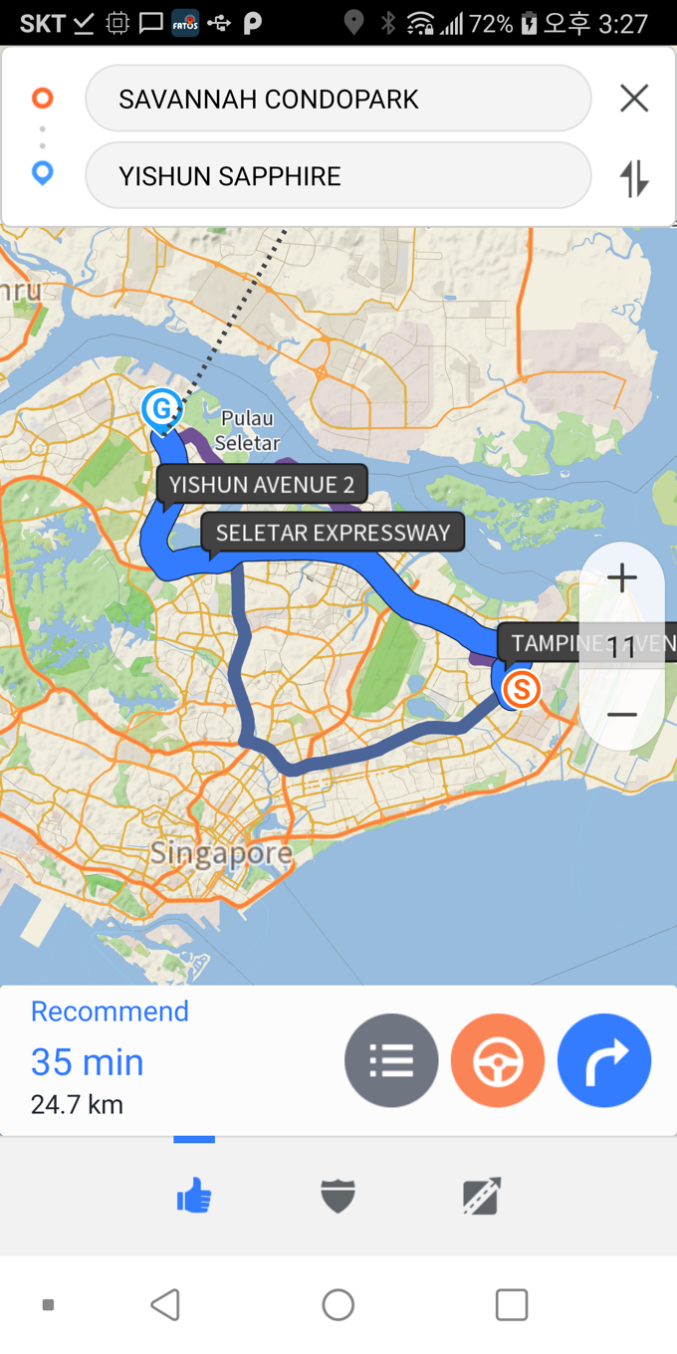
The Handler that is passed to the FM\_RouteVol2\_Via tells you the results of the path navigation and provides a summary of the path when it succeeds.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (TNaviMainActivity.java) //Run when the path is completed,  private void routeSuccess(){  ArrayList<RouteCardData> summaryCardData = new ArrayList<RouteCardData>();  ArrayList<RouteSummaryData> summaryData = m\_FMInterface.FM\_RouteSummary(FMBaseActivity.onFatosMapListener);  for(int i=0; i<summaryData.size(); i++ ){  RouteSummaryData summarydata = summaryData.get(i);  //String strTypeName, int nType, int nLength, int nTime, int nFee, int nAvgSpeed, int nTurnCongestion, int nOptionColor  RouteCardData data = new RouteCardData(summarydata.strTypeName,summarydata.nType,summarydata.nLength,summarydata.nTime,summarydata.nFee,  summarydata.nAvgSpeed,  summarydata.nTurnCongestion,m\_Context);  summaryCardData.add(data);  }  saved\_summaryCardData = new ArrayList<RouteCardData>();  saved\_summaryCardData = summaryCardData;  if(!bLastRouteFlag) {  setAPP\_MODE(TNaviActionCode.APP\_MODE\_SHOWING\_SUMMARY);  fragmentManager = getSupportFragmentManager();  Bundle args = new Bundle();  args.putParcelableArrayList(TNaviActionCode.LONGTOUCH\_ROUTESUMMARY, summaryCardData);  SummaryFragment summaryFragment = new SummaryFragment();  summaryFragment.setArguments(args);  GoLib.getInstance().goFragment(getSupportFragmentManager(), R.id.container,summaryFragment,tag\_summary\_fragment,args);  }  } |

# Route details

FATOS SDK Sample provides a summary-related screen through AMapSDKCommonRGInfoFragment.java.

|  |
| --- |
| #Sample in [FATOS SDK Guide App for Android](https://github.com/devfatoscorp/FatosSDKTest) (activity\_tnavi\_main.xml)  <biz.fatossdk.exlib.fatoslayout.AMapSCLinearLayout  android:orientation="vertical"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  >  <biz.fatossdk.exlib.fatoslayout.AMapSCRelativeLayout  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <fragment  android:id="@+id/map\_fragment"  android:name="kr.fatos.tnavi.tnavifragment.FatosMapFragment"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent" />  <fragment  android:id="@+id/drive\_ui\_fragment"  android:name="**biz.fatossdk.fminterface.drivefragment.AMapSDKCommonRGInfoFragment**"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent" />  <ImageView  android:layout\_width="28dp"  android:layout\_height="28dp"  android:id="@+id/imageView\_MapCenterPoint"  android:src="@drawable/img\_mapcenter"  android:layout\_centerInParent="true"  android:visibility="gone"/>  </biz.fatossdk.exlib.fatoslayout.AMapSCRelativeLayout>  </biz.fatossdk.exlib.fatoslayout.AMapSCLinearLayout>  ….  ….  private int m\_nUIMode = ANaviApplication.UI\_DRIVE\_MODE;  public void setUIMode(int nUIMode) {  m\_nUIMode = nUIMode;  switch (m\_nUIMode)  {  case ANaviApplication.UI\_DRIVE\_MODE:  {  m\_CurSpeed.setVisibility(View.VISIBLE);  m\_AerialLayout.setVisibility(View.VISIBLE);  m\_CommpassLayout.setVisibility(View.VISIBLE);  m\_layoutCurPos.setVisibility(View.VISIBLE);  }break;  case ANaviApplication.UI\_SUMMARY\_MODE:  {  m\_CurSpeed.setVisibility(View.INVISIBLE);  m\_AerialLayout.setVisibility(View.GONE);  m\_CommpassLayout.setVisibility(View.GONE);  m\_layoutCurPos.setVisibility(View.GONE);  }break;  case ANaviApplication.UI\_SIMUL\_MODE:  {  m\_CurSpeed.setVisibility(View.VISIBLE);  m\_AerialLayout.setVisibility(View.VISIBLE);  m\_CommpassLayout.setVisibility(View.VISIBLE);  m\_layoutCurPos.setVisibility(View.VISIBLE);  }break;  case ANaviApplication.UI\_NORMAL\_DRIVE\_MODE:  {  m\_CurSpeed.setVisibility(View.VISIBLE);  m\_AerialLayout.setVisibility(View.VISIBLE);  m\_CommpassLayout.setVisibility(View.VISIBLE);  m\_layoutCurPos.setVisibility(View.VISIBLE);  }break;  }  } |

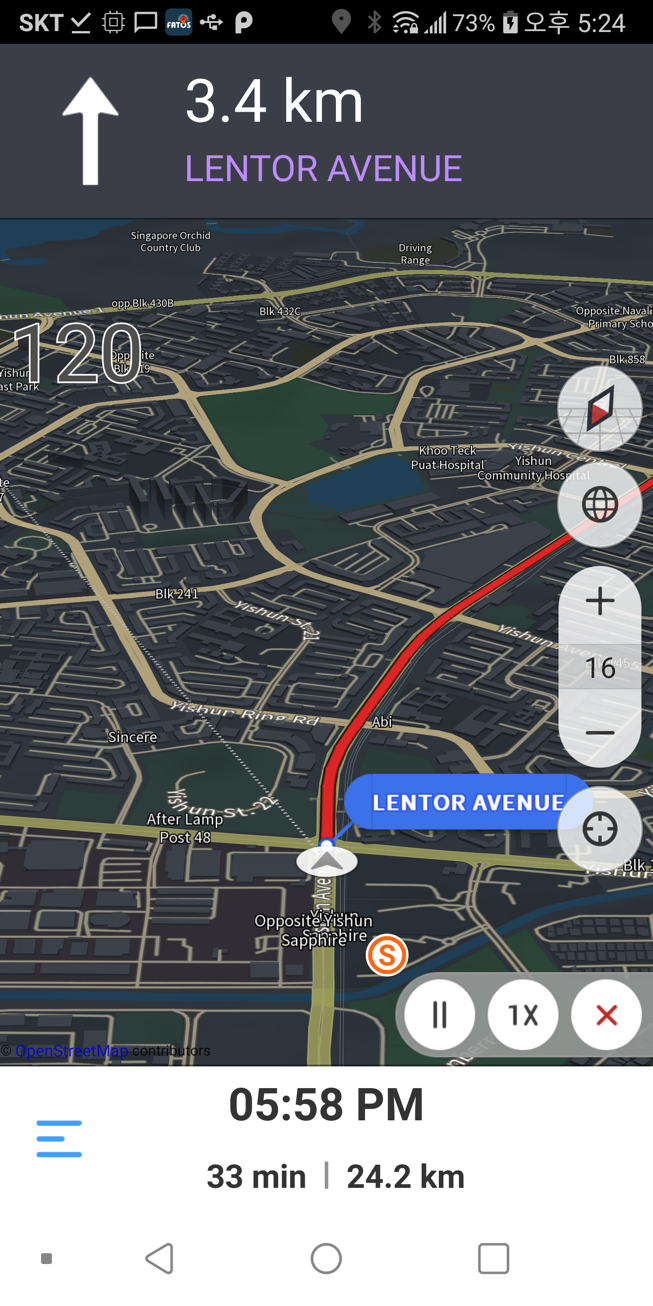


# Directions

# Starting routing

With the path navigation complete, you can receive route guidance through FM\_StartRGService.

|  |
| --- |
| m\_FMInterface.FM\_StartRGService(FMBaseActivity.onFatosMapListener) |



# About the path

You can get driving information through the FM\_GetDriveInfo() while driving.

|  |
| --- |
| **public class FMDriveInfo**  private int m\_iGpsSpeed;   * gps speed   private boolean m\_bIsRoute;   * true: Driving on the road false : normal driving   private String m\_strCurPosName;   * Driving administrative names   private double m\_fCurLonX = -180;   * Current location coordinates(Longitude, x)   private double m\_fCurLatY = -180;   * Current location coordinates(Latitude, y)   private int m\_nCurAngle = 0;   * Current location coordinates(Angle)   private int m\_nGpsStatus = 0;   * gps State value   0 : bad  1 : simulation  2 : good  3: test  Only the coordinates of “good” state are valid values.  private boolean m\_bCannotEnterRoad = false;   * inaccessible roads   private int m\_nCannotEnterRoadCode = 0;   * inaccessible roads   private int m\_nGateNearbyDist = 0;   * Distance around gate   private int m\_nTotalRemainderDist= 0;   * The distance remaining to your destination   private int m\_nServiceLinkRemainderTime= 0;   * Time remaining to your destination   public int nViaTotalCount = 0;   * Total number of way points   public int[] listViaRemainderDist;   * The remaining distance to the waypoints (Unit : meter)   public int[] listViaRemainderTime;   * The remaining time to the waypoints (Unit : Second)   private int m\_nTripTime = 0;   * Total driving time   private int m\_nTripDist= 0;  Total mileage |

# API list

# Initializing process

# public int initFatosNaviEngine()

|  |  |
| --- | --- |
| Description | On an app that uses the FATOS SDK, you should call FMBaseActivity'sinitFatosNaviEngine for the first time. |
| Parameter & Return | 1 : Success  INIT\_ERROR\_JNI = -1; // -1: JNI Initialization failure  INIT\_ERROR\_SDL = -2; // -2 : SDL Initialization failure  INIT\_ERROR\_RP = -3; // -3 : Network initialization failure  INIT\_ERROR\_FORCEUPDATE = -4; // -4 : Forced updates  INIT\_ERROR\_DAL = -5; // -5 : DAL Other errors  INIT\_ERROR\_CONFIG = -6; // -6 : Failed to load the config  INIT\_ERROR\_LOCAL\_SEARCH\_DB = -7;//-7 : No local (search) DB  INIT\_ERROR\_LOCAL\_RP\_DB = -8; //-8 : No road (search) DB |

# 

# public int FM\_Init(Context context, String strApiKey)

|  |  |
| --- | --- |
| Description | Appsthat use FMSmust be called at the time of initialization. |
| Parameter & Return | Context : deliver Application Context.  Return : FMError.Success -> success  FMError.Error -> failure |

# public int FM\_Destory()

|  |  |
| --- | --- |
| Description | A function that must be called at the end of the app. |
| Parameter & Return | Return : FMError.Success -> success  FMError.Error -> failure |

# map

# MapStatusListener

|  |  |
| --- | --- |
| Description | End of path / map status information interface |
| Parameter & Return | public void onRouteFinish()  public void onMapMove(boolean bMapMove) |

# public void setMainMapOption(boolean bDrawStartImg,boolean bDrawGoalImg,boolean bDrawPinImg,boolean bDrawEVWare)

|  |  |
| --- | --- |
| Description | Set the Map Option state   * Set whether you want to draw a specific object on a map |
| Parameter & Return | Function NativeNavi.MAP\_OBJ\_FONTTBT,   * Whether or not to set Font TBT Draw   NativeNavi.MAP\_OBJ\_FLAG\_START,   * Whether or not to draw the starting flag   NativeNavi.MAP\_OBJ\_FLAG\_GOAL,   * Whether or not to draw the destination flag   NativeNavi.MAP\_OBJ\_FLAG\_PIN   * Whether or not to draw poi Pin image   NativeNavi.MAP\_OBJ\_EVWHERE,   * Whether or not to draw EV information on the map   NativeNavi.MAP\_OBJ\_ADVENCED\_TBT,   * Whether or not to draw Advance tbt   NativeNavi.MAP\_OBJ\_SCALEBAR   * Whether or not to draw the Scale bar |

# public void onMapReady()

|  |  |
| --- | --- |
| Description | Callback message tells the user when the FATOS map has completed the initialization process and is ready to use the feature. |
| Example | @Override  public void onMapReady() {  ((FMBaseActivity)getActivity()).onMapReady();  } |

# public void FM\_MoveCurPosition(FMBaseActivity.OnFatosMapListener onFatosMapListener)

|  |  |
| --- | --- |
| Description | go to the current position |
| Example | m\_FMInterface.FM\_MoveCurPosition(FMBaseActivity.OnFatosMapListener) |

# public void FM\_SetMapAirMode(int nMapType)

|  |  |
| --- | --- |
| Description | Set air mode |
| Parameter & Return | nMapType :  FMInterface.MAPMODE\_AIR\_ON\_BUILDING\_ON; // air mode on building on  FMInterface.MAPMODE\_AIR\_ON\_BUILDING\_OFF; // air mode on building off  FMInterface.MAPMODE\_AIR\_OFF\_BUILDING\_ON; // air mode off building on |

# public void FM\_EnableSmartZoomInOut(Boolean bEnable)

|  |  |
| --- | --- |
| Description | Set whether to use automatic scale mode while driving |
| Parameter & Return | bEnable  true : Auto-driving mode on  false : Auto-driving mode off |

# public void FM\_EnableWalkingMode(Boolean bEnable)

|  |  |
| --- | --- |
| Description | Whether or not to use walking mode |
| Parameter & Return | bEnable  true : Walking mode on  false : Walking mode off |

# public int FM\_GetAddress(Handler \_resultHandler, double xlon, double ylat)

|  |  |
| --- | --- |
| Description | When you get the administrative name based on the base coordinates. |
| Parameter & Return | #Param \_resultHandler : handler to get search results or status  xlon, ylat : Coordinates to get an address |

# public int FM\_GetPosFromScreen(int x, int y, double[] xlonlat)

|  |  |
| --- | --- |
| Description | The screen returns the touch pointof wgs84 coordinates. |
| Parameter & Return | #Param x,y : Map touched screen position  xlonlat : wgs84 coordinates for touched position |

# public ArrayList<GPSPOS> FM\_GetLastGPSTrace()

|  |  |
| --- | --- |
| Description | We bring GPS trajectory information from the beginning of the route guide |
| Parameter & Return | .#Return  ArrayList<GPSPOS> : GPS Trajectory information List |

# public void FM\_DrawGrid(double[] x, double[] y)

|  |  |
| --- | --- |
| Description | Draw a grid on the map. |
| Parameter & Return | #Param  x,y : Grid coordinates list |

# public void FM\_SetMapCenter(float hCenter, float vCenter)

|  |  |
| --- | --- |
| Description | Bottom left 0.f, 0.f Top right : 1.f, 1.f Set the center of the reference.  ex) 0.5f, 0.5f the map screen center position   * Among the parameters, hCenter and vCenter means the position when looking at the bottom left corner of the terminal screen at 0,0 and the top right 1,1. Therefore, 0.5,0.5 becomes the center of the screen. |
| Parameter & Return | #Param  hCenter : The horizontal position of the map area  vCenter : Vertical position of map area |

# public void FM\_AutoSetMapCenter(boolean bMapMove)

|  |  |
| --- | --- |
| Description | Depending on the bMapMove value, the center of the map is divided into driving mode and map movement mode. |
| Parameter & Return | #Param  bMapMove : Whether the current map is in a moving state. |

# public void FM\_SetMapPosition(FMBaseActivity.OnFatosMapListener onFatosMapListener, double xlon, double ylat)

|  |  |
| --- | --- |
| Description | the center of the map is divided into driving mode and map movement mode. |
| Parameter & Return | .#Param  onFatosMapListener : Listener associated with Mapview  xlon, ylat : WGS84 coordinates of the location you want to move |

# public void FM\_GetTbtImageRes(int nTbtCode)

|  |  |
| --- | --- |
| Description | Get TBTCode to return its image resource ID |
| Parameter & Return | #Param  nTbtCode : Tbtcode in summary information |

# public void FM\_SetNaviVolume(int volume)

|  |  |
| --- | --- |
| Description | Adjust the volume of the basic voice. |
| Parameter & Return | #Param  volume : Volume size (0 to 100) – 0 is seen as mute. |

# public void FM\_IsPlaying()

|  |  |
| --- | --- |
| Description | Check to see if the default voice is currently occurring |
| Parameter & Return | . #return  true : When the primary voice is occurring  false : The default voice is not occurring. |

# public void FM\_GetMapVersion(NaviVersion versionInfo)

|  |  |
| --- | --- |
| Description | Get the Current background, search, path search, others’ Map name / version (SDI, EV.), update availability, status of updates, etc |
| Parameter & Return | #Param  NaviVersion : See structure information |

# public class NaviVersion

|  |
| --- |
| Thestructure that contains information from Map Data |
| int nMapDate;   * Background DB build date   DBVersion[] listMap;   * Background DB information list   int nSearchDate;   * search DB build date   DBVersion[] listSearch;   * search DB information list   int nNetworkDate;   * path navigation DB build date   DBVersion[] listNetwork;   * path navigation DB information list   DBVersion[] listEtc;   * other DB information list (SDI, EV...) |

# public class DBVersion

|  |
| --- |
| A structure containing each DB information for Background, search, route navigation, others (SDI, EV.) |
| String strName;   * DB name   int nCurDate;   * Current DB build date   int nUpdateDate;   * updatable DB build date   double fVersion;   * version   boolean bUpdate;   * Whether the update is available   int nUpdateStatus;   * 0 : not downloaded, 1 : Downloading, 2 : Downloaded |

# public void FM\_RequestUpdate(int nDownloadGubun, long freeSpace, final MapUpdateListenerCallback mapUpdateListener)

|  |  |
| --- | --- |
| Description | DBVersion-based information is requested map download for each category based on the updateable (bUpdate) value. |
| Parameter & Return | #Param  nDownloadGubun : Background, search, and path navigation, Other(SDI,EV) are classified  # Gubun values definition  public static final int MAP\_DOWNLOAD\_BASEMAP = 1;  public static final int MAP\_DOWNLOAD\_SEARCH = 2;  public static final int MAP\_DOWNLOAD\_RP = 3;  public static final int MAP\_DOWNLOAD\_ANDO = 4;  freeSpace : Remaining capacity of storagewith current Map DB  mapUpdateListener : Listener for getting update status information |

# public interface MapUpdateListenerCallback

|  |
| --- |
| A structure containing each DB information for Background, search, route navigation, other (SDI, EV.) |
| void onData(mapDownloadResult resultMapDownload);  void onError(int nErrorCode); |

# public void FM\_MapUpdateCancel()

|  |  |
| --- | --- |
| Description | Call when you cancel during a map update. |

# 

# Explore the path

# public void FM\_StartRGService(FMBaseActivity.OnFatosMapListener onFatosMapListener)

|  |  |
| --- | --- |
| Description | In the route summary, you should select the route and call it when you go to the driving screen. |
| Parameter & Return | onFatosMapListener : Listener associated with Map view |

# public ArrayList<RouteSummaryData> FM\_RouteSummary(FMBaseActivity.OnFatosMapListener onFatosMapListener)

|  |  |
| --- | --- |
| Description | After the successful path navigation, call to obtain information about each path option. |
| Parameter & Return | Parm: onFatosMapListener : Listener associated with Map view  Return : Information list for each path option |

# public void FM\_SelectRoute(int nType)

|  |  |
| --- | --- |
| Parameter & Return | # Parm  nType : type information for RouteSummaryData |
| Description | In the path summary screen, you will be called when you select the path option. |

# public class RouteSummaryData

|  |  |
| --- | --- |
| Description | POI search results structure  public String strTypeName;   * Route navigation options title   public int nType;   * Route navigation options (Minimum, Shortest, Recommended etc)   public int nLength;   * Total length of path(Unit : meter)   public int nTime;   * Total time spent on routes(unit : second)   public int nFee;   * Pricing information(Won)   public int nAvgSpeed;   * Average speed of intervals   public int nTurnCongestion;   * Congestion in the area   public int nOptionColor;   * Route options Color |
| Parameter & Return | Path summary |

# public ArrayList<RouteSummaryDataDetail> FM\_RouteSummaryDetail(int nType, int nFromIdx)

|  |  |
| --- | --- |
| Description | Transfer the detailed path information for each Type. |
| Parameter & Return | # Parm  nType : type information for RouteSummaryData  nFromIdx : current RG Index  ( nFromIdxshould be obtained from the public void updateServiceLinkIdx (int nIdx, boolean bReset)as asample. ) |

# public class RouteSummaryDataDetail

|  |  |
| --- | --- |
| Description | public int nTbtCode;   * tbt code (1~53) * Included in the SDK. Use R.drawable.list\_01\_straight to R.drawable.list\_53\_via3   public int nLength;   * The turn length(unit : meter)   public int nTime;   * The turn pass time(unit : second)   public double dLonX;  public double dLatY;   * Turn Point Coordinates   public String strDongName;   * Turn point administration name   public int nTurnCongestion;   * Congestion in the area   public int nAvgSpeed;   * Average speed of intervals   public String strTurnInfoTextName;   * Turn Text Info |
| Parameter & Return | Path summary details |

# public void FM\_CancelRoute()

|  |  |
| --- | --- |
| Description | If there is a path, cancel it. (Do not draw route lines, exits, destinations flags, etc as the route is cancelled.)  After cancellation, onRouteFinish is called Callback. |

# public void FM\_SetSimulationSpeed(int nSpeed)

|  |  |
| --- | --- |
| Description | Set the speed during the simulation drive. |
| Parameter & Return | nSpeed : Simulated speed |

# public List<RoutePosition>FM\_GetViaPOIList()

|  |  |
| --- | --- |
| Description | POI information list for stop over navigation must be set at least two at least when requesting a path navigation. |
| Parameter & Return | Return : returned list to include coordinate information |

# public void FM\_RouteVol2\_Via(Handler \_resultHandler, List<RoutePosition> m\_PoiList)

|  |  |
| --- | --- |
| Description | navigate the waypoints over the coordinates set in the M\_PoiList ( 0 idx : starting point, last idx : destination, other way points)  Among the error codes of \_resultHandler, FMError.FME\_ERROR\_VIACOUNT\_ROUTE occurs when the coordinates list is less than two.  Ex)  RoutePosition positionList0 = new RoutePosition();  positionList0.x = 0;  positionList0.y = 0;  positionList0.name = "starting point";  positionList.add(positionList0);  RoutePosition positionList1 = new RoutePosition();  positionList1.x = 127.46534606;  positionList1.y = 37.28005176;  positionList1.name = "icheon city court";  positionList1.addr = "Anheung-dong, Icheon City, Gyeonggi";  positionList.add(positionList1);  RoutePosition positionList2 = new RoutePosition();  positionList2.x = 127.43412687;  positionList2.y = 37.27288535;  positionList2.name = "Icheon City Hall";  positionList2.addr = " Zhongli-dong, Icheon City, Gyeonggi ";  positionList.add(positionList2);  RoutePosition positionList3 = new RoutePosition();  positionList3.x = 127.46534606;  positionList3.y = 37.28005176;  positionList3.name = "Icheon General Bus Terminal";  positionList3.addr = "Anheung-dong, Icheon City, Gyeonggi";  m\_FMInterface.FM\_RouteVol2\_Via(new  HttpResultHandler(FatosMapSDKSampleLandFragment.this),positionList |
| Parameter & Return | resultHandler : handler to get search results or status  m\_PoiList : stating point, way point, destination poi information List |

# public class RoutePosition

|  |  |
| --- | --- |
| Description | Structure containing coordinates of way point poi  Information other than the information below is not used on the SDK.  public int nPosKind; // 0 starting point , 1 destination, 2 Waiting for input  public double x = 0;  public double y = 0;   * Poi wgs84 Coordinates   public String name; // Nameex) “Gangnam Station(2 Line)”  public int nPoiID; // POI ID (If there is no, 0)  public String poiFlag; // POI RPFlag (If there is no, 0)  public int nFloor = 1; // Number of floors(1=1st floor, -2=Underground level 2, ...)  public boolean bDir = true; // Directional check  public String addr; // Address information(If there is no, null)  public Boolean bPassingPoint = false; // Setting the perception of a stopover that only takes a path |

# public Boolean FM\_GoNextGoal()

|  |  |
| --- | --- |
| Description | While using way point including route setting, it is used to pass the way point just in front and then take the route from the next way point. |
| Parameter & Return | # Parm  return : Normally set to the next stop.: true  I can't set it as the next stop.: false  (If you're only at your destination) |

# public void FM\_StartSimulation ()

|  |  |
| --- | --- |
| Description | After navigating the path, you will be driving a simulation when you call. |

# public void FM\_StopSimulation ()

|  |  |
| --- | --- |
| Description | I call it when I end the simulation mode and go back to driving mode |

# public void FM\_GetMapCenterPos(double[] lonlat)

|  |  |
| --- | --- |
| Description | Get the WGS84 coordinates of the map center. |
| Parameter & Return | Double array to receive map-centric coordinates |

# public void FM\_GetSubMapCenterPos(double[] lonlat)

|  |  |
| --- | --- |
| Description | Get submap WGS84 coordinates. |
| Parameter & Return | Double array to receive map-centric coordinates |

# 

# POI search

# public int FM\_SearchPOI(Handler \_resultHandler,String strSearchName, boolean bSaveRecent)

|  |  |
| --- | --- |
| Description | API when you want to get search results by a specific name |
| Parameter & Return | \_resultHandler : handler to get search results or status  strSearchName : POI search name  bSaveRecent : Whether you want to save your latest search results list |

# public int FM\_RecommendWord(Handler \_resultHandler,String strSearchName)

|  |  |
| --- | --- |
| Description | Give a list of recommendations according to the search characters that include the first sound |
| Parameter & Return | strSearchName : Recommended search letters(include the first sound) |

# public ArrayList<POIItem> FM\_GetSearchResult(int nSortOption)

|  |  |
| --- | --- |
| Description | Use when you get integrated search results. It is sorted according to nSortOption |
| Parameter & Return | nSortOption  0 : Search for accuracy  1 : Search by distance |

# public ArrayList<String> FM\_GetRecommentWordResult()

|  |  |
| --- | --- |
| Description | Get the search results for the recommendations. |

# 

# 