|  |
| --- |
| **FATOS Software Development Kit**  **V1.0.0** |

Revision History

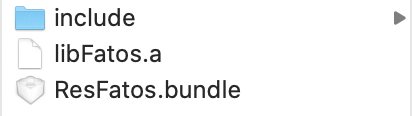
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ver** | **Revision date** | **Details** | **Written by** | **Approved by** |
| 1.0.0 | 2020. 02. 24 | V1.0.0 first draft released | Sim GyuBin | Yoo ChunSung |

# Get FATOS SDK code

Copy or download FATOS SDK for iOS from GitHub

# Setting up a development project

FATOS SDK for iOS is supported by Xcode 11 or higher, ios 12 or higher.



FATOS SDK for iOS needs to apply three(3) folders and libraries to the project : include (static library header), libFatos.a (static library), ResFatos.bundle (SDK Resources).

**How to add libFatos.a and basic framework**

Click to ‘project explorer’ -> click ‘TARGETS’ -> Build Phases -> add a FATOS SDK for iOS and framework needed for Link Binary With Librarises.

Add six(6) item

* libFatos.a (FATOS SDK for iOS),
* GLKit.framework,
* OpenGLES.framework,
* CoreLocation.framework,
* AVFoundation.framework,
* AudioToolbox.framework,

스크린샷이(가) 표시된 사진

자동 생성된 설명

**How to add ResFatos.bundle (SDK Resources)**

Click ‘project explorer’ -> click ‘TARGETS’ -> Build Phases -> add ResFatos.bundle to Copy Bundle Resoureces

스크린샷이(가) 표시된 사진

자동 생성된 설명

**How to link include (static library header)**

Click ‘project explorer’ -> click ‘TARGETS’ -> click ‘Build Settings’ -> Search Paths -> Set ‘include’ folder path to Header Search Paths

스크린샷이(가) 표시된 사진

자동 생성된 설명

# Getting and applying SDK key

**Request to SDK key**

Users the SDK key must be issued by email(dev@fatoscorp.com)

**Apply SDK key**

장치, 스크린샷이(가) 표시된 사진

자동 생성된 설명register SDK key issued to “Project > info.plist > sdk\_key”

# Feature configuration of FATOS SDK for iOS

**Map**

Supply FatosMapView

Allows FatosMap to be used flexibly.

Supply Map Option functions.

Supply Map control API like change map mode, move coordinate, Zoom in/out and map picker etc.

**Search**

Supply FATOS search results

**Route Planning**

Supply FATOS route planning results

Supply route planning data management functions

**Route Guide**

Supply custom API for route guide horizontal/length response

**Supply Interface & Base Class**

Map , Route Guidance-related functions and controls are provided through Base Class to reduce duplication and easily add navigation functions.

# Understanding code

# Create AppDelegate

# To inherit AppDelegate

To set basic SDK, user have to inherit FatosBaseAppDeleate.

Then, generate AppDelegate inherited from FatosBaseAppDeleate.

|  |
| --- |
| #import <FatosBaseAppDeleate.h>  @interface AppDelegate : FatosBaseAppDelegate    @end |

FatosBaseAppDeleate provides app background processing and gps-related processing.

# Creat ViewController

**5.2.1 NaviEngine Event**

|  |
| --- |
| Sample in [FATOS SDK Guide App for](https://github.com/devfatoscorp/FatosSDKTest) iOS (ViewController.m)  #import "ViewController.h"  #import <FatosBaseAppDelegate.h>  @interface ViewController ()  @end  - (void)viewDidLoad {    [[FatosBaseAppDelegate sharedAppDelegate] setRouteStartListener:self selector:@selector(onRouteStart:)];  [[FatosBaseAppDelegate sharedAppDelegate] setRouteResultListener:self selector:@selector(onRouteResult:)];  [[FatosBaseAppDelegate sharedAppDelegate] setRouteCancelListener:self selector:@selector(onRouteCancel)];  [[FatosBaseAppDelegate sharedAppDelegate] setRouteCompleteListener:self selector:@selector(onRouteComplete)];  }  - (void) onRouteStart:(NSMutableDictionary \*)jsonDic  {  int nType = [[jsonDic objectForKey:@"type"] intValue];    switch (nType) {  case 0:  [self showIndicator];  break;  case 1:  [self showIndicator];  break;  case 2:  break;  default:  break;  }  }  - (void) onRouteResult:(NSMutableDictionary \*)jsonDic  {  int nType = [[jsonDic objectForKey:@"type"] intValue];    [self hideIndicator];  }  - (void) onRouteCancel  {    }  - (void) onRouteComplete  {    }} |

four events can be registered and used in FatosBaseAppDelegate.

🡪 (setRouteStartListener, setRouteResultListener, setRouteCancelListener, setRouteCompleteListener)

**onRouteStart**

Call when route planning starts.

[Type] 0: initial search, 1:Re-search, 2:Periodic re-search

**onRouteResult**

Call when route planning ends.

[Type] 0: initial search, 1:Re-search, 2:Periodic re-search

**onRouteCancel**

Call when route planning cancels.

**onRouteComplete**

Call when route planning completes.

# Drawing a map & Handling events

|  |
| --- |
| Sample in [FATOS SDK Guide App for](https://github.com/devfatoscorp/FatosSDKTest) iOS (ViewController.m)  #import <FatosMapView.h>  @interface ViewController () <FatosMapViewDelegate>  @end  - (void)viewDidLoad {  [super viewDidLoad];  FatosMapView \*mapView = [[FatosMapView alloc] initWithFrame:[[UIScreen mainScreen] bounds]];  mapView.delegate = self;  self.view = mapView;  }  #pragma mark - FatosMapViewDelegate  - (void) MapLevelUpdateListener:(int)nLevel  {    }  - (void) PosWorldLocationUpdateListener:(NSString \*)strLocation  {    }  - (void) TouchMoveModeListener:(int)nMode  {    }  - (void) MapLongTouchListener:(int)x y:(int)y  {  }  - (void) UpdatePickerInfo:(NSString \*)strID nLong:(int)nLong nLat:(int)nLat  {    }  - (void) MapReadyListener  {    } |

ViewController root view 로 FatosMapView 등록

Register FatosMapViewDelegate in ViewController class.

**MapLevelUpdateListener**

When the map level changes, it receives the changed level value.

**TouchMoveModeListener**

When the map location changes, it receives the changed location value.

**TouchMoveModeListener**

When map view touch, it receives the touch event status value.

nMode 1 : touch began, 2 : touch end

**MapLongTouchListener**

When map view long touch, it receives the touch event status value.

**UpdatePickerInfo**

When user symbol registers, it receives symbol touch coordinate value.

**MapReadyListener**

Called when map view creation is complete

# API list

# FatosNaviBridge(Route, Search API)

|  |
| --- |
| @interface FatosNaviBridge : NSObject  + (void) ReRoute;  + (void) Route:(NSString \*)startLat startLon:(NSString \*)startLon  goalLat:(NSString \*)goalLat goalLon:(NSString \*)goalLon;  + (void) CancelRoute;  + (void) DriveControl:(int)value;  + (void) DriveSpeed:(int)value;  + (void) DriveClose;  + (void) Search:(id)target selector:(SEL)selector searchText:(NSString \*)searchText;  + (void) StartRouteGuidance:(int)index;  + (void) StartSimulation:(int)index;  + (BOOL) IsRoute;  + (NSString \*) GetRouteSummaryJson;  + (NSString \*) GetGeoCodeString:(double)lon lat:(double)lat;  @end |

**ReRoute**

Route re-search API

**Route**

Route planning API

Route planning based on start point and destination coordinates.

If user call start point - startLat :@”0”, startLon :@”0”, route search based on current location.

**CancelRoute**

API to cancel route planning

**DriveControl**

API to operate simulated driving

value 1 : pause 2: resume

**DriveSpeed**

Simulated driving speed API

Value unit : km (kilometer)

**Search**

search API

Set targets and selectors to receive search keywords and search results as parameters

Search results are received in the form of json string

Callback function example

Ex) - (void) searchResult:(NSString \*)strResult

Search results json example

Ex)

|  |
| --- |
| {  pgno : “page number”,  cnt : “number of searches”  items : [  {  "id" : "POI ID",  "addr1" : "POI Display name",  "addr2" : "New address(the road name address)",  "phone" : Phone number(separators comma(,))",  "cate" : "Classification code",  "posx" : "POI X coordinates",  "posy" : "POI Y coordinates",  "entx" : “POI point of entry X coordinates",  "enty" : "POI point of entry Y coordinates",  "dist" : "distance"  }  ]  } |

**StartRouteGuidance**

API to start route guidance

[Value] 0 : recommendation, 8 : general, 32 : free

**StartSimulation**

API to start simulated driving

[Value] 0 : recommendation, 8 : general, 32 : free

**IsRoute**

API to check during route guidance

**GetRouteSummaryJson**

API to get route summary information in json format

Values are only available for successful route planning

|  |
| --- |
| {  contexts : [  {  "Type" : "(int)Route planning option",  "Length" : "(String)Total distance",  "Time" : "(int)Total time",  "Fee" : "(int) Charge information,  "AvgSpeed" :"(int) section average speed",  "TurnCongestion" : "(int) section congestion"  }  ]  } |

**GetGeoCodeString**

API that receives location information based on coordinates.

# FatosMapBridge(Map API)

|  |
| --- |
| @interface FatosMapViewBridge : NSObject  + (void) setViewMode:(nonnull NSNumber \*)mode;  + (void) setLayer:(NSDictionary \*) baseLayerType bVisible:(NSDictionary \*)bVisible;  + (void) MapLevelIn:(nonnull NSNumber \*)type;  + (void) MapLevelOut:(nonnull NSNumber \*)type;  + (void) setMapLevel:(float)fLevel type:(nonnull NSNumber \*)type;  + (void) MapAuto;  + (void) MapMove:(float)fLonX fLatY:(float)fLatY;  + (void) MapSelectRouteLine:(int)nIndex;  + (void) SummaryMapSetting:(NSDictionary \*)lineColor  xScale:(float)xScale yScale:(float)yScale  hCenter:(float)hCenter vCenter:(float)vCenter blnViewMode:(BOOL)blnViewMode;  + (void) DefaultMapSetting;  + (void) SelectRouteLine:(int)index;  + (void) ApplySelectRouteLine:(int)index;;  + (void) SetMapCenter:(float)hCenter vCenter:(float)vCenter;  + (void) SetPosWGS84:(double)xlon ylat:(double)ylat;  + (NSMutableDictionary \*) GetPosWorldFromScreen:(float)fCenterX fCenterY:(float)fCenterY;  + (NSMutableDictionary \*) ConvWorldtoWGS84:(int)x y:(int)y;  + (NSMutableDictionary \*) GetMapCenter;  + (NSMutableDictionary \*) GetPosWorldtoWGS84FromScreen:(float)fCenterX fCenterY:(float)fCenterY; ;  @end |

**setViewMode**

API to change map view mode

0 : MAP\_VIEW\_MODE\_BIRD, 1 : MAP\_VIEW\_MODE\_NORTHUP, 2: MAP\_VIEW\_MODE\_HEADING

**setLayer**

API to change map view layer

Example of parameters

|  |
| --- |
| baseLayerType = {  @”0”: @”5” (BASEMAP\_LAYER\_AEROPHOTO),  @”1”: @”8” (BASEMAP\_LAYER\_BUILDING),  @”2”: @”9” (BASEMAP\_LAYER\_POI),  @”3”: @”6” (BASEMAP\_LAYER\_ROAD),  @”4”: @”4” (BASEMAP\_LAYER\_SATELLITE)  };  bVisible = {  @”0”: @"false",  @”1”: @"true",  @”2”: @"true",  @”3”: @"true",  @”4:” @"false",  }; |

**MapLevelIn**

API to reduce map view level

[Type] 0 : ani 1 : direct

**MapLevelOut**

API to increase map view level

[Type] 0 : ani 1 : direct

**setMapLevel**

API to change map view level

Level : Level value to be changed

[Type] 0 : ani 1 : direct

**MapAuto**

API to move current location

**MapMove**

API to move map location

**SummaryMapSetting**

API to change to route summary screen.

Set the map center value, scale value, and route line values as parameters.

Example of lineColor parameter

Separate Rgba values with commas(,)

|  |
| --- |
| lineColor = {  @”0”: @''255,108,108,255,  @”1”: @''21,181,36,255,  @”2”: '@”2,228,193,255,  }; |

**DefaultMapSetting**

API to set default map view screen

**SelectRouteLine**

API to select route

**ApplySelectRouteLine**

API to apply route selected

**SetMapCenter**

API to change map center

**SetPosWGS84**

API to move map based on coordinates (No ani effect)

**GetPosWorldFromScreen**

API to get coordinate value of screen center

Example of return value

|  |
| --- |
| {  x : (int)  y: (int)  } |

**ConvWorldtoWGS84**

API to convert World coordinate to WGS84.

Example of return value

|  |
| --- |
| {  xlon : (float)  ylon: (float)  } |

**GetMapCenter**

API to get map center value

Example of return value

|  |
| --- |
| {  hCenter : (float)  vCenter: (float)  } |

**GetPosWorldtoWGS84FromScreen**

API to get coordinate values of screen center (WGS84 coordinate)

# FatosEnvBridge(Route, Map, Environment API)

|  |
| --- |
| @interface FatosEnvBridge : NSObject  + (void) SetLanguage:(nonnull NSNumber \*)value;  + (void) SetPathLineColor:(nonnull NSNumber \*)value;  + (void) SetNavigationOptions:(NSArray \*)array;  + (void) SetMapColor:(nonnull NSNumber \*)value;  + (void) SetSmartDrivingMode:(nonnull NSNumber \*)value;  + (void) SetCamreaOptions:(NSArray \*)array;  + (void) SetOperationState:(NSArray \*)array;  + (void) SetFacility:(NSArray \*)array;  + (void) SetGuidevoice:(nonnull NSNumber \*)value;  + (void) SetRediscover:(nonnull NSNumber \*)value;  + (void) SetWayPoint:(nonnull NSNumber \*)value;  + (void) SetHiPass:(BOOL)value;  + (void) SetCarType:(nonnull NSNumber \*)value;  + (void) SetFuel:(nonnull NSNumber \*)value;  + (void) SetSeatPosition:(nonnull NSNumber \*)value;  + (void) SetCarvata:(nonnull NSNumber \*)value;  + (NSString \*) GetPathLineColor;  + (NSString \*) GetLanguage;  + (NSString \*) GetNavigationOptions;  + (NSString \*) GetMapColor;  + (NSString \*) GetSmartDrivingMode;  + (NSString \*) GetCamreaOptions;  + (NSString \*) GetOperationState;  + (NSString \*) GetFacility;  + (NSString \*) GetGuidevoice;  + (NSString \*) GetRediscover;  + (NSString \*) GetWayPoint;  + (NSString \*) GetHiPass;  + (NSString \*) GetCarType;  + (NSString \*) GetFuel;  + (NSString \*) GetSeatPosition;  + (NSString \*) GetCarvata;  @end |

**SetLanguage**

API to set language

0 : KOREA, 1 : ENGLISH

**SetPathLineColor**

API to set route line color

0 : Red, 1 : Blue, 2 : Green, 3 : Purple

**SetNavigationOptions**

API to change navigation options

Example of bool array parameter

|  |
| --- |
| [  true, (Recommended)  true, (Expressway priority)  true, (Avoid toll roads)  false, (Avoid toll roads)  false, (Shortest route)  ], |

**SetMapColor**

API to change map color

0 : day, 1: night, 2: auto

**SetSmartDrivingMode**

API to set smart driving mode

0 :off, 1 : on

**SetCamreaOptions**

API to set camera caution.

Example of bool array parameter

|  |
| --- |
| [  true, (Fixed camera)  true, (Mobile camera)  true, (Signal control section)  false, (Intervention control)  false, (Parking control),  false, (Bus lane)  ], |

**SetOperationState**

API to set navigation options

Example of bool array parameter

|  |
| --- |
| [  true, (Sharp curve)  true, (Child protection zone)  true, (Accident hazard)  ], |

**SetFacility**

API to set facility caution.

Example of bool array parameter

|  |
| --- |
| [  true, (Traffic information collection)  ], |

**SetGuidevoice**

API to set up voice options guide

0 : TTS, 1 : gerneral

**SetRediscover**

API to set periodic re-search

0 : 5min, 1 : 10min

**SetWayPoint**

API to set up waypoint guide

0 : Use, 1 : Not Use

**SetHiPass**

API to set up hi-pass guide

**SetCarType**

API to set vehicle type

0 : compact car, 1 : passenger car, 2 : SUV, 3 : MPV, 4 : Truck, 5 : Special freight vehicle

**SetFuel**

API to set fuel type

0 : Gasoline, 2 : diesel, 3 : LPG

**SetSeatPosition**

API to set driving position type

0 : Left, 1 : Right

**SetCarvata**

API to set carvata type

**GetPathLineColor**

API to get route line color information

0 : Red, 1 : Blue, 2 : Green, 3 : Purple

**GetLanguage**

API to get language setting information

0 : KOREA, 1 : ENGLISH

**GetNavigationOptions**

API to get guide setting information

Return Json string of set information

**GetMapColor**

API to get map setting information

**GetSmartDrivingMode**

API to get smart driving mode information

**GetCamreaOptions**

to get camera caution setting information

Return Json string of set information

**GetOperationState**

API to get guide setting information

Return Json string of set information

**GetFacility**

API to get facility caution setting information

Return Json string of set information

**GetGuidevoice**

API to get voice guide setting information

**GetRediscover**

API to get periodic re-search setting information

**GetWayPoint**

API to get waypoint setting information

**GetHiPass**

API to get hi-pass setting information

**GetCarType**

API to get vehicle type setting information

**GetFuel**

API to get fuel type setting information

**GetSeatPosition**

API to get driver position setting information

**GetCarvata**

API to get carvata setting information