

# 位置情報収集基盤位置測位ライブラ リiOSインターフェース仕様書

株式会社ゼンリン

Version 3.0.0, 2023.11.21

# 目次

1. 改定履歴	1
2. はじめに	2
2.1. 動作仕様について	2
3. 概要	3
3.1. 機能概要	3
3.2. 動作環境	3
4. ライブラリの利用方法	4
4.1. ライブラリの提供の仕方	4
4.2. ライブラリの基本的な使用方法	4
4.3. ライブラリ使用時に設定が必要な項目	4
4.4. パラメータと測位方式等の関係	7
4.5. LocationInfo	8
5. ライブラリ導入方法	10
5.1. ライブラリをプロジェクトに追加	10
5.2. 関連ライブラリの導入	10
5.3. プロジェクトの設定を変更	10
5.4. Swiftを用いている場合	10
6. プロパティ	11
6.1. delegate	11
7. API	12
7.1. defaultManager	12
7.2. measuringStart	12
7.3. measuringStop	21
7.4. libraryVersion	22
7.5. currentLocation	23
7.6. librarySettings	26
8. プロトコル	27
8.1. LocationManagerDelegate	27
9. シーケンス	30

# 1. 改定履歴

#	日付	版数	変更内容
1	2022/01/25	1.0.0	初版作成
2	2022/09/09	1.1.0	7.2. measuringStart 7.2.3. パラメータ ・ 認証サーバ環境をパラメータに追加 7.5. currentLocation 7.5.2. プロトタイプ宣言 ・ パラメータ「認証サーバ環境」の有無でAPIを分けた 7.5.3. パラメータ ・ 認証サーバ環境をパラメータに追加
3	2022/09/29	2.0.0	対応するライブラリバージョンを「2.0.0」に変更 対応OSを「iOS15 / iOS16」に変更
4	2023/05/26	2.1.0	対応するライブラリバージョンを「2.1.0」に変更 認証エラーの記載を追加 「エラーコード一覧」を見直し
5	2023/08/30	2.1.1	7.5. currentLocation ・ 「Swiftを用いている場合」の記載を追加
6	2023/11/21	3.0.0	対応するライブラリバージョンを「3.0.0」に変更 対応OSに「iOS17」を追加 「Swiftを用いている場合」にcurrentLocationメソッド実装の移行手順を追加 「バッテリー残量」に説明と注釈を追加

## 2. はじめに

本ドキュメントでは、各種のiOSアプリケーションにて、測位に関する精度・省電力に関する機能を提供するiOS位置測位ライブラリのインタフェースを記載しています。  
本仕様書はライブラリバージョン3.0.0について記載しています。それ以前のバージョンについては、以前のバージョンの仕様書を参照してください。

### 2.1. 動作仕様について

標準位置情報サービスと大幅変更位置情報サービス(Significant-Change Location Service)を使用した測位を行うか、領域観測と滞在監視および、大幅変更位置情報サービスを使用した測位を行うかを省電力モード設定で選択できます。

領域観測と滞在監視および、大幅変更位置情報サービスを使用した測位の場合は、標準位置情報サービスと大幅変更位置情報サービスを使用した測位に比べて消費電力が抑えられます。

## 3. 概要

### 3.1. 機能概要

本インターフェースを利用することで、アプリケーションにて測位に関する精度・省電力機能の設定および測位情報を取得することが可能となります。

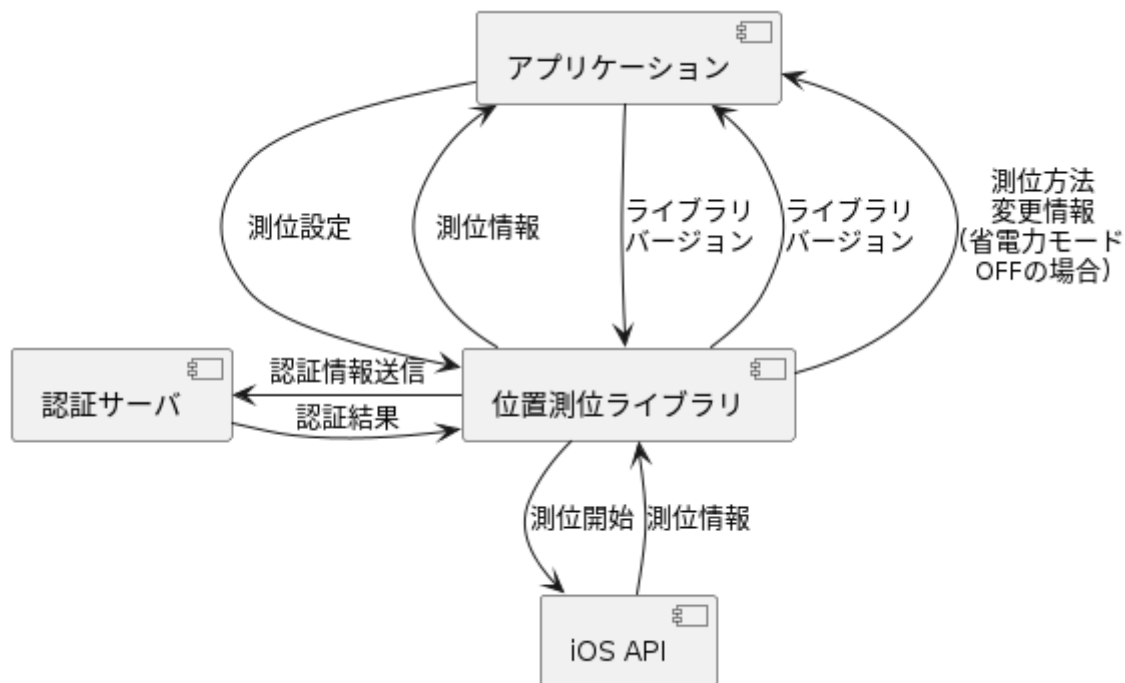


図 1. iOS位置測位ライブラリ概要

### 3.2. 動作環境

#### 3.2.1. 対応OS

iOS15 / iOS16 / iOS17

## 4. ライブラリの利用方法

### 4.1. ライブラリの提供の仕方

本ライブラリは、iOSアプリケーションにスタティックにリンクするライブラリとして提供されます。そのため、ユーザに本ライブラリの機能を使用したアプリケーションを提供する場合は、アプリケーションとリンクした形で提供する必要があります。

#### 4.1.1. ライブラリの形式について

ライブラリは.xcframework形式での提供となります。

XCFrameworkの詳細については下記をご覧ください。

<https://help.apple.com/xcode/mac/11.4/#/dev6f6ac218b>

### 4.2. ライブラリの基本的な使用方法

アプリケーションは、起動時に本ライブラリに対し、defaultLocationManagerのインスタンスを取得することにより、使用可能となります。

その後measuringStartをコールすることにより、指定された測位精度により自動的にGPS,WiFiNetwork、3G/4Gの情報を使い分けて測位した結果が、アプリケーションが指定したコールバック関数に対して位置情報が通知されます。（測位に失敗した場合はNGの情報が通知されます）

Androidと異なり、位置取得に必要な移動距離を指定することにより指定された距離以上の移動検出で通知を行うことができます。（詳細は後述します）

アプリケーションの終了時にはmeasuringStopをコールしてください。  
これにより、iOSに対して位置測定動作を停止させることができます。

後述する基地局測位を使用している場合、停止を行わずに終了した場合は位置の通知を行うため再起動されます。

位置測位ライブラリは位置測位開始時に認証処理を行います。  
認証には別途発行されたAPIキー、クライアントID、秘密鍵を用いてください。  
認証サーバにて認証を行なった結果、認証不許可（APIキーやクライアントID、秘密鍵が誤っている等）と判断された場合は認証エラーとなります。  
ネットワーク障害や通信タイムアウト（30秒）、サーバまたはライブラリの障害など、認証サーバで認証処理ができない場合は認証エラーにはなりません。

### 4.3. ライブラリ使用時に設定が必要な項目

#### 4.3.1. 位置情報の使用宣言

iOSには特定の機能に対応していない端末へのアプリのインストールを防止する仕組みが存在します。

**Info.plist**の**UIRequiredDeviceCapabilities**キーの値(配列)に機能名を指定することで、その機能に対応していない端末へのインストールを防止できます。

本ライブラリでは位置測位情報サービスの使用が必須のため、この配列に`location-services`の文字列を追加してください。

また、GPSに対応していない端末では、測位精度や測位間隔が大幅に落ちる可能性があります。

そのため、**GPSに対応していない端末にアプリをインストールされたくない場合は同様に`gps`の文字列も追加してください。**

詳細はAppleのドキュメント [デバイス機能の要件](#) を参照してください。

### 4.3.2. 位置取得のためのシステム認証

本ライブラリは位置測位情報サービスを利用するため、以下の2つの項目についてユーザによる事前の許可が必要です。

- アプリ使用時に位置測位を行うことに対するユーザの許可

システムダイアログでユーザに許可を求める必要があります。

このダイアログには「1度だけ許可」「Appの使用中は許可」「許可しない」の選択肢が表示されます。

ユーザに「Appの使用中は許可」を選択させ、許可を得てください。

- バックグラウンドで位置測位を行うことに対するユーザの許可

上記アプリ使用時の位置測位許可後、さらにシステムダイアログで許可を求める必要があります。

このダイアログには「"使用中のみ許可"のままにする」「"常に許可"に変更」の選択肢が表示されます。

ユーザに「"常に許可"に変更」を選択させ、許可を得てください。

測位時にこれらの許可設定が**未選択**の場合、本ライブラリはシステムダイアログを表示しユーザに許可を求めます。(選択済みの場合、どの選択肢を選んだかに関わらずダイアログを表示しません)

また、これらのダイアログではユーザに位置取得の目的を提示することが必須となっており、`Info.plist`に記述した内容がメッセージとして表示されます。

そのため、本ライブラリを使用する際には**必ず**`Info.plist`に以下の2つのキーと値を追加してください。(設定していない場合、ダイアログが表示できないため測位が開始されません)

- `NSLocationWhenInUseUsageDescription`

- • • アプリ使用時の位置測位のシステムダイアログに表示する文字列

- `NSLocationAlwaysAndWhenInUseUsageDescription`

- • • バックグラウンド位置測位のシステムダイアログに表示する文字列

詳細はAppleのドキュメント [Requesting Authorization for Location Services\(英語\)](#) を参照してください。

### 4.3.3. アプリケーション終了時の動作について

大幅変更位置情報サービス（以下このドキュメントでは「基地局測位」と表記します）を利用している場合（最初から指定する場合、標準的な測位方法実行中に節電のため測位方式が変更されている場合）、iOSからアプリケーションが何らかの理由で終了させられてしまった場合や、アプリケーションのタスクを終了した場合でも、利用する基地局の変化（ハンドオーバー）を検出した時に、システムからアプリケーションが再起動させられます。

領域観測機能および滞在監視機能を使用している場合も、領域観測結果または、滞在監視結果を検出した時にアプリケーションが再起動させられます。

アプリケーションが起動された際に、AppDelegateに対して通知される

application:didFinishLaunchingWithOptionsメソッドに

UIApplicationLaunchOptionsLocationKeyが設定されています。

アプリケーションでは、このキーが設定されていた場合measuringStartを再度呼び出してください。これにより、測位を継続することができます。（下記サンプルコードを参照）

```
-(BOOL)application:(UIApplication*)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    NSLog(@"didFinishLaunchingWithOptions");

    // ここに通常の起動時処理を記述する。

    // 以下、位置変更検出時のコード
    id locationValue = [launchOptions
objectForKey:UIApplicationLaunchOptionsLocationKey];

    // 位置変更を検出しているか？
    if (locationValue)
    {
        // locationValueがnilでない場合は位置情報の検出により起動したと判断
        // ここにライブラリを再起動するコードを記述
    }
    return YES;
}
```

ただし、システムによって再起動されるよりも前にユーザがアプリを起動した場合は、

UIApplicationLaunchOptionsLocationKeyによる判別ができません。

ユーザがOSよりも先にアプリを起動した場合でも測位再開が必要な場合は、アプリの要件に合わせて別途対応を行ってください。



## 4.4. パラメータと測位方式等の関係

表 1に標準位置情報サービスと大幅変更位置情報サービスを使用した測位方式における、測位方法と設定パラメータを記載します。

表 1. 測位方法と設定パラメータ

測位精度 (X)	距離 (Y)	測位方式	GPS	バッテリーレ ベル低下判定時	バッテリー使用量 (見込み)
10>X	Y=0	標準測位	常時	基地局測位	最大
	1<Y<50	距離測位	常時	基地局測位	大（移動なしの場合使用量は中）
	50<Y	距離測位	常時	基地局測位	大（移動なしの場合使用量は中）
5000>X≥10	Y=0		状況に応じて	基地局測位	中（測位精度の値が小さいほど大）
	1<Y<50	距離測位	状況に応じて	基地局測位	中（測位精度の値が小さいほど大）
	50<Y	距離測位	状況に応じて	基地局測位	中（測位精度の値が小さいほど大）
X≥5000	無視	基地局測位	ハンドオーバー 検出時	基地局測位	極小

## 4.5. LocationInfo

位置情報を格納するデータクラスです。

### 4.5.1. プロパティ

表 2 に LocationInfo クラスのプロパティを記載します。

表 2. LocationInfo のプロパティ

プロパティ名	属性	型名	説明
date	readonly	NSString*	測位日付(yyyyMMdd) 端末のタイムゾーン設定に変換された日付です。
time	readonly	NSString*	測位時間(HH:mm:ss) 端末のタイムゾーン設定に変換された時間です。
latitude	readonly	double	緯度
longitude	readonly	double	経度
horizontalAccuracy	readonly	double	緯度経度の測位誤差(m) 値が負値の場合、緯度経度が無効となります。
verticalAccuracy	readonly	double	標高の測位誤差(m) 値が負値の場合、altitudeが無効となります。
altitude	readonly	double	標高(m)
timeStamp	readonly	NSDate*	測位情報のタイムスタンプ 形式はGMTの値です。 ※詳細な測位日時が必要な場合に使用して下さい。
speed	readonly	double	速度(m/s) 値が負値の場合は無効となります。
bearing	readonly	double	方位(度) 北を0とし、東が90、南が180、西が270となります。 値が負値の場合は無効となります。

### 4.5.2. 測位情報取得方法

以下の方法にて測位情報を取得します。

```
-(void)measuringResult:(LocationInfo *)result
status:(LocationManagerStatus)status error:(NSError *)error
{
    NSString *date = result.date;           //測位日
    NSString *time = result.time;           //測位時間

    double latitude = result.latitude;       //緯度
    double longitude = result.longitude;     //経度
    double horizontal = result.horizontalAccuracy; //誤差(m)
```

```
double vertical = result.verticalAccuracy;    //標高誤差(m)
double altitude = result.altitude;           //標高(m)
double speed = result.speed;                 //速度(m/s)
double bearing = result.bearing;            //方位(度)
}
```

## 5. ライブラリ導入方法

既存のアプリにライブラリを導入する場合は、下記の手順で導入をしてください。

### 5.1. ライブラリをプロジェクトに追加

1. 対象プロジェクト内に任意のディレクトリを作成する
2. 作成したディレクトリの中に下記ファイルを入れる
  - libLocationManagerProbe.xcframework
3. Xcode上で対象プロジェクトに作成したディレクトリをドラッグ&ドロップする

### 5.2. 関連ライブラリの導入

1. TARGETS>General>Frameworks, Libraries, and Embedded Content 項目を開き、下記必須フレームワークを追加する
  - CoreLocation.framework

### 5.3. プロジェクトの設定を変更

1. TARGETS>General>Frameworks, Libraries, and Embedded Content 項目を開き、libLocationManagerProbe.xcframeworkのEmbed項目をEmbed & Signに変更する
2. TARGETS>Build Setting>Other Linker Flags 項目を開き、-ObjCを追加する
3. TARGETS>Signing & Capabilitiesを開きBackground ModesのLocation updatesにチェックを入れる
4. info.plistを開き下記を追加する(valueは許可ダイアログに表示するメッセージを追加すること)

項目名	Value
NSLocationAlwaysAndWhenInUseUsageDescription	位置情報利用許可（常に許可）ダイアログに表示する文字列
NSLocationAlwaysUsageDescription	位置情報利用の許可ダイアログ（iOS10以前）に表示する文字列
NSLocationWhenInUseUsageDescription	位置情報利用の許可ダイアログに表示する文字列

### 5.4. Swiftを用いている場合

1. swiftの場合は[アプリ名]-Bridging-Header.hファイルを作成しTARGETS>Build Setting>Objective-C Bridging Headerにファイルのパスを追加する
2. [アプリ名]-Bridging-Header.hファイルには下記を記載する

```
#import <libLocationManagerProbe/LocationManager.h>
```

エラーが発生した場合は、サンプルアプリと見比べながら設定をご確認ください。

## 6. プロパティ

表 3に位置測位ライブラリのプロパティ一覧を示します。

表 3. プロパティ一覧

プロパティ	型	概要
delegate	id<LocationManagerDelegate>	LocationManagerクラスのデリゲート

### 6.1. delegate

#### 6.1.1. 説明

デリゲートを設定します。

#### 6.1.2. プロパティ宣言

```
@property (nonatomic, retain) id<LocationManagerDelegate> delegate;
```

#### 6.1.3. コメント

- デリゲートはLocationManagerDelegateプロトコルを採用する必要があります。詳細は [Section 8.1](#) LocationManagerDelegate を参照してください。

# 7. API

表 4に位置測位ライブラリのAPI一覧を示します。

表 4. API一覧

API	概要
+ defaultLocationManager	クラスのインスタンス取得
- measuringStart	測位設定を一括設定し、測位を開始
- measuringStop	測位を停止
- libraryVersion	ライブラリのバージョン情報取得
- currentLocation	現在地の取得
- librarySettings	ライブラリに設定されている設定値の取得

## 7.1. defaultManager

### 7.1.1. 説明

クラスのインスタンスを取得します。

### 7.1.2. プロトタイプ宣言

```
+ (LocationManager *)defaultLocationManager;
```

### 7.1.3. パラメータ

なし

### 7.1.4. 戻り値

位置測位ライブラリのオブジェクト

### 7.1.5. コメント

- ・シングルトンパターンを適用したクラスであるため、本APIはアプリケーションが起動してから終了するまで同一のオブジェクトを返します。

## 7.2. measuringStart

### 7.2.1. 説明

測位精度・距離・間隔・を一括設定し、測位を開始します。

本メソッドは、UIスレッド（メインスレッド）以外のスレッド（バックグラウンドスレッド、ワーカース

レッドなど) から実行してください。  
UIスレッドで実行した場合、認証処理が完了するまでUIが停止します。

### 7.2.2. プロトタイプ宣言

```
- (MeasuringStartResult)measuringStart:(NSDictionary *)settings  
error:(NSError **)error;
```

### 7.2.3. パラメータ

settings [l] : 測位設定を指定するDictionary  
DictionaryのKeyと形式を表5に示します。

表 5. 測位設定のDictionary

No.	パラメータ概要	Key (※1)	形式	必須	説明
1	測位精度 (m) (※2)	MeasuringSettingAccuracyKey	NSNumber (float)	-	設定する値を小さくすることにより精度は上がりますが、GPSをより多く使用することになり電力消費が大きくなります。 指定する場合は、0以上の値を指定してください。 指定が無い場合は前回値で動作します。(初回の場合は最高精度(-1))
2	距離 (m) (※3)	MeasuringSettingDistanceKey	NSNumber (float)	-	新しい測位情報を通知するまでに必要な移動距離を位置情報フレームワーク (CoreLocation) に設定します。 指定する場合は、0以上の値を指定してください。 1未満の値を指定した場合は最高頻度 (-1) で動作します。 指定が無い場合は前回値で動作します。(初回の場合は最高頻度(-1))

No.	パラメータ概要	Key (※1)	形式	必須	説明
3	バッテリー残量(%) (※4)	MeasuringSettingBatteryLevelIntervalKey	NSNumber (float)	-	<p>標準位置情報サービスと大幅変更位置情報サービスの測位方法を切り替えるバッテリー残量の閾値。</p> <p>指定した値までバッテリー残量が低下した場合は大幅変更位置情報サービスに切り替わり、指定した値を超えてバッテリーが充電されると標準位置情報サービスに切り替わります。</p> <p>指定する場合は、0以上100以下の値を指定してください。</p> <p>指定が無い場合は、40%で動作します。</p> <p>本パラメータは、省電力モードをOFFで設定した場合に使用されます。省電力モードONの場合は指定不要で、指定しても使用されません。</p>
4	測位切替精度(m)(※5)	MeasuringSettingSignificantChangeAccuracyKey	NSNumber (float)	-	<p>測位方法を標準位置情報サービスから大幅変更位置情報サービスに切り替えるための精度。</p> <p>指定する場合は、0以上の値を指定してください。</p> <p>指定が無い場合は、5000mで動作します。</p> <p>本パラメータは、省電力モードをOFFで設定した場合に使用されます。省電力モードONの場合は指定不要で、指定しても使用されません。</p>
5	監視領域半径(m)(小) (※6)	MeasuringSettingMonitoringAreaSmallRadiusKey	NSNumber (double)	-	<p>領域観測で使用する領域の半径（小サイズ）。</p> <p>指定する場合は、1以上の値を指定してください。</p> <p>指定が無い場合は、100mで設定されます。</p> <p>本パラメータは、省電力モードをONで設定した場合に使用されます。省電力モードOFFの場合は指定不要で、指定しても使用されません。</p>
6	監視領域半径(m)(中) (※6)	MeasuringSettingMonitoringAreaMiddleRadiusKey	NSNumber (double)	-	<p>領域観測で使用する領域の半径（中サイズ）。</p> <p>指定する場合は、1以上の値を指定してください。</p> <p>指定が無い場合は、1000mで設定されます。</p> <p>本パラメータは、省電力モードをONで設定した場合に使用されます。省電力モードOFFの場合は指定不要で、指定しても使用されません。</p>



No.	パラメータ概要	Key (※1)	形式	必須	説明
7	監視領域半径(m)(大) (※6)	MeasuringSettingMonitoringAreaLargeRadiusKey	NSNumber (double)	-	<p>領域観測で使用する領域の半径（大サイズ）。</p> <p>指定する場合は、1以上の値を指定してください。</p> <p>指定が無い場合は、2500mで設定されます。</p> <p>本パラメータは、省電力モードをONで設定した場合に使用されます。省電力モードOFFの場合は指定不要で、指定しても使用されません。</p>
8	監視領域半径(m)(特大) (※6)	MeasuringSettingMonitoringAreaOversizeRadiusKey	NSNumber (double)	-	<p>領域観測で使用する領域の半径（特大サイズ）。</p> <p>指定する場合は、1以上の値を指定してください。</p> <p>指定が無い場合は、5000mで設定されます。</p> <p>本パラメータは、省電力モードをONで設定した場合に使用されます。省電力モードOFFの場合は指定不要で、指定しても使用されません。</p>
9	監視移動速度(km/h) (低) (※6)	MeasuringSettingMonitoringSlowSpeedKey	NSNumber (double)	-	<p>領域観測で使用する領域の半径を決定する速度閾値(低)。</p> <p>本パラメータは、省電力モードをONで設定した場合に使用されます。省電力モードOFFの場合は指定不要で、指定しても使用されません。</p> <p>指定する場合は、1以上で有効条件(※7)を満たした値を指定してください。</p> <p>条件を満たしていない場合はエラーとなります。</p>
10	監視移動速度(km/h) (中) (※6)	MeasuringSettingMonitoringMiddleSpeedKey	NSNumber (double)	-	<p>領域観測で使用する領域の半径を決定する速度閾値(中)。</p> <p>本パラメータは、省電力モードをONで設定した場合に使用されます。省電力モードOFFの場合は指定不要で、指定しても使用されません。</p> <p>指定する場合は、1以上で有効条件(※7)を満たした値を指定してください。</p> <p>条件を満たしていない場合はエラーとなります。</p>

No.	パラメータ概要	Key (※1)	形式	必須	説明
11	監視移動速度(km/h) (高) (※6)	MeasuringSettingMonitoringHighSpeedKey	NSNumber (double)	-	領域観測で使用する領域の半径を決定する速度閾値（高）。 本パラメータは、省電力モードをONで設定した場合に使用されます。省電力モードOFFの場合は指定不要で、指定しても使用されません。 指定する場合は、1以上で有効条件(※7)を満たした値を指定してください。 条件を満たしていない場合はエラーとなります。
12	省電力モード	MeasuringSettingLowEnergyModeKey	NSNumber (BOOL)	-	YES(1)：省電力モードON。領域観測と滞在監視および、大幅変更位置情報サービスを使用した測位で動作。 NO(0)：省電力モードOFF。標準位置情報サービスと大幅変更位置情報サービスを使用した測位で動作。 本パラメータは、指定が無い場合はNOで設定されます。
13	APIキー	MeasuringSettingApiKey	NSString	○	位置測位ライブラリ認証時に設定するAPIキー。
14	クライアントID	MeasuringSettingClientIdKey	NSString	○	位置測位ライブラリ認証時に設定するクライアントID。
15	秘密鍵	MeasuringSettingSecretKey	NSString	○	位置測位ライブラリ認証時に設定する秘密鍵。
16	認証サーバ環境	AuthEnv	NSNumber(int)	-	位置測位ライブラリの認証を行う環境。 指定する値は、表 6を参照してください。 指定が無い、または表に存在しない値が指定された場合は、顧客本番環境に接続されます。

表 6. 認証サーバ環境の種類

No.	認証サーバ環境の種類	定義名 (※1)	値
1	顧客本番環境	AuthEnv_Pro	0
2	顧客検証環境	AuthEnv_Stg	1
3	内部検証環境	AuthEnv_Test	2

認証エラーの場合、以下のNSErrorが返却されます。  
本エラーを受信した際には、アプリ側にて「測位開始」または「現在地取得(単測位)」を再度実施することで、サーバー認証が再度行われます。（本ライブラリ内では、サーバー認証のリトライ処理は行いません）

表 7. 認証エラー内容

code	エラーコード(表 8参照)
localizedDescription	エラーの詳細

表 8. エラーコード一覧

値	説明
1	トークン発行認証失敗
2	認証不許可

(※1)：KeyはLocationManager.hに定義された文字列定数を使用するようにします。

(※2)：標準位置情報サービスと大幅変更位置情報サービスを使用した測位の場合、測位精度(accuracy)の設定値を測位切替精度以上にすると、測位方法は「標準位置情報サービス」ではなく消費電力の少ない「大幅変更位置情報サービス」になります。その際には精度、位置情報更新最小距離、同一位置判定間隔の各パラメータ設定は無効となります。

iOSでは指定された測位精度により自動的にプロバイダが選択されます。

そのためAndroidと異なり、プロバイダの指定は行いません。

現在、10m未満の測位精度が指定されると、GPS常時起動となります。

10m以上5000m未満の場合、下記の範囲の精度で測位を行います。

10~99m 10m近辺の精度

100m~999m 100m近辺の精度

1000m~2999m 1000m近辺の精度

3000m~4999m 3000m前後の精度

これらの精度は基本的に、WiFiのAP情報や基地局の情報から得られる精度で測定します。

しかし、GPSを使用しないわけではありません。（常時起動ではないが、必要に応じて起動する）

実際に使用するタイミングは公開されていませんが、高い精度ほどGPSを使用する回数が増える傾向があるようです。

また、仮に精度を小さい値に設定しても、高精度の結果が必ずしも得られるわけではありません。バッテリー消費を抑える目的ならば、できるだけ大きい値の精度を設定することを推奨します。

(※3)：距離が0または小さすぎる場合は、測位が途切れる場合があります。その際は、距離を10m以上に設定すると改善されることがあります。

(※4)：iOS17以上の場合、OSから取得できるバッテリー残量は端末上に表示されているバッテリー値とは異なり、5%単位で丸められ以下のように±1~2%の範囲で近いほうに寄せられます。

表 9. 端末表示上のバッテリー残量とOSから取得できるバッテリー残量の関係

端末表示	OS取得	説明
0%	0%	そのまま
1%	0%	5%より0%の方が近いため0%となる
2%	0%	5%より0%の方が近いため0%となる
3%	5%	0%より5%の方が近いため5%となる
4%	5%	0%より5%の方が近いため5%となる
5%	5%	そのまま
6%	5%	10%より5%の方が近いため5%となる

端末表示	OS取得	説明
7%	5%	10%より5%の方が近いいため5%となる
8%	10%	5%より10%の方が近いいため10%となる
9%	10%	5%より10%の方が近いいため10%となる
10%	10%	そのまま

測位方式の切り替えはOSから取得できるバッテリー残量で判断しているため、iOS16以前のように端末表示上のバッテリー残量が変わったタイミングで実行されず、OSから取得できるバッテリー残量が5%単位で変化したタイミングに実行されます。

表 10. 設定した値（閾値）に応じた測位方式の切替タイミング（バッテリー残量60～80%のみ抜粋）

端末表示	OS取得	【閾値】 60～64%	【閾値】 65～69%	【閾値】 70～74%	【閾値】 75～79%
80%	80%				
79%					
78%					標準位置変更
77%	75%				大幅変更位置変更
76%					
75%					
74%					
73%				標準位置変更	
72%	70%			大幅変更位置変更	
71%					
70%					
69%					
68%			標準位置変更		
67%	65%		大幅変更位置変更		
66%					
65%					
64%					
63%		標準位置変更			
62%	60%	大幅変更位置変更			
61%					
60%					

(※5)：標準位置情報サービスと大幅変更位置情報サービスを使用した測位の場合、iOSではOSが提供する機能として「基地局測位」があります。

この機能は、通常の位置情報取得機能をOFFにして携帯が基地局間をハンドオーバーした場合にのみ現在位置を測定する機能です。

位置情報に関する電力消費量の大幅な減少が見込まれ、iOSアプリケーションでは通常この機能の使用が

推奨されています。

本ライブラリでは、指定された測位精度を判定しこの値を超えていた場合は自動的に標準位置情報サービス測位ではなく、大幅変更位置情報サービス測位を使用し消費電力を抑えます。

ライブラリに指定する測位精度をこの値を境に上下させることにより意図的に測位方法を変更することもできます。

例えば、通常は広域地図を表示していて頻繁な更新が必要ない場合はこの値より大きい値を測位精度として指定します。ユーザの要求により細かい精度で表示する必要が発生したら、この値より低い値を設定しなおします。

(※6)：領域観測と滞在監視および、大幅変更位置情報サービスを使用した測位の場合に使用されます。監視領域半径（小/中/大/特大）と監視移動速度(低/中/高)の関係を以下に示します。

例：以下の値で設定した場合。

監視領域半径(小)：300m

監視領域半径(中)：1000m

監視領域半径(大)：5000m

監視領域半径(特大)：10000m

監視移動速度(低)：30km/h

監視移動速度(中)：60km/h

監視移動速度(高)：100km/h

表 11. 監視移動速度と監視領域半径

監視移動速度（時速）	30km/h未満	30km/h以上～ 60km/h未満	60km/h以上～ 100km/h未満	100km/h以上
監視領域半径（m）	300m	1000m	5000m	10000m

(※7)：監視移動速度設定の有効条件

監視移動速度(低/中/高)の有効条件は以下の通り。

表 12. 監視移動速度

	有効条件1	有効条件2	有効条件3	有効条件4
低	未設定または0	設定あり	設定あり	設定あり
中	未設定または0	未設定または0	設定あり	設定あり
高	未設定または0	未設定または0	未設定または0	設定あり

有効条件1：中または高が設定されていると無効となります。

有効条件2：高が設定されていると無効となります。

有効条件3：設定値の大小関係が「小<中」でない場合は無効となります。

有効条件4：設定値の大小関係が「小<中<高」でない場合は無効となります。

## 7.2.4. 戻り値

表 13. 戻り値

リターンコード	値	説明
MeasuringStartResult_Error	-1	開始失敗

リターンコード	値	説明
MeasuringStartResult_Standard	1	標準測位方式で開始
MeasuringStartResult_SignificantChange	2	大幅変更測位方式で開始
MeasuringStartResult_RegionVisit	3	領域観測および、滞在観測を使用した測位方式で開始

## 7.2.5. コメント

- 測位中に本関数が呼ばれた場合、測位を停止して新たに測位を開始します。
- application:didFinishLaunchingWithOptionsがUIApplicationLaunchOptionsLocationKeyを伴って起動された場合は終了中にiOSが位置の移動を検出したことを意味しています。  
そのため、新たな測位情報を取得するため本関数を呼び出してください。
- 測位の結果はデリゲートに通知されます。  
詳細は[Section 8.1](#) LocationManagerDelegate を参照してください。
- 測位方法の変更はデリゲートに通知されます。  
詳細は[Section 8.1](#) LocationManagerDelegate を参照してください。
- 設定パラメータに関して、必須パラメータがない場合、型が異なる場合、または説明欄に記載の範囲以外の値を入力した場合は、パラメータ設定エラーとなり、測位開始せずに処理を終了します。
- 測位開始に失敗した場合、エラー内容をコンソールログを出力します。  
コンソールログは、「[parameterCheck]エラー内容」という形式で出力します。

## 7.3. measuringStop

### 7.3.1. 説明

測位を停止します。

### 7.3.2. プロトタイプ宣言

```
-(void)measuringStop;
```

### 7.3.3. パラメータ

なし

### 7.3.4. 戻り値

なし

### 7.3.5. コメント

測位の完全な終了を意図している場合は、アプリケーション終了時に必ず呼び出してください。

## 7.4. libraryVersion

### 7.4.1. 説明

ライブラリのバージョン情報を返します。

### 7.4.2. プロトタイプ宣言

```
-(NSString *)libraryVersion;
```

### 7.4.3. パラメータ

なし

### 7.4.4. 戻り値

ライブラリのバージョン

デバッグビルドされたライブラリの場合は、バージョン文字列の後ろに[“ debug”]が付加されます。

### 7.4.5. コメント

戻り値の形式は、x.x.xとなります。



## 7.5. currentLocation

### 7.5.1. 説明

現在地の位置（単測位）を返します。

measuringStartメソッドで測位を開始し、測位中の状態でも利用可能です。

本メソッドは、UIスレッド（メインスレッド）以外のスレッド（バックグラウンドスレッド、ワーカースレッドなど）から実行してください。

UIスレッドで実行した場合、認証処理が完了するまでUIが停止します。

### 7.5.2. プロトタイプ宣言

<認証サーバ環境指定なし>

```
-(LocationInfo *)currentLocation:(double)accuracy apiKey:(NSString *)apiKey clientId:(NSString*)clientId secret:(NSString*)secret error:(NSError **)error;
```

<認証サーバ環境指定あり>

```
-(LocationInfo *)currentLocation:(double)accuracy apiKey:(NSString *)apiKey clientId:(NSString*)clientId secret:(NSString*)secret authEnv:(NSInteger)authEnv error:(NSError **)error;
```

### 7.5.3. パラメータ

accuracy [I] : 測位精度

apiKey [I] : APIキー

clientId [I] : クライアントID

secret [I] : 秘密鍵

authEnv [I] : 認証サーバ環境(表 6参照)

※引数指定なし、または表に存在しない値が指定された場合は、顧客本番環境に接続されます。

error [O] : 位置取得時にエラーになった際のエラー情報

エラーの場合、以下のNSErrorが返却されます。

本エラーを受信した際には、アプリ側にて「測位開始」または「現在地取得(単測位)」を再度実施することで、サーバー認証が再度行われます。（本ライブラリ内では、サーバー認証のリトライ処理は行いません）

表 14. 認証エラー内容

code	エラーコード(表 15参照)
localizedDescription	エラーの詳細

表 15. エラーコード一覧

値	説明
1	トークン発行認証失敗
2	認証不許可
3	測位エラー
4	バリデーションエラー

## 7.5.4. 戻り値

測位情報LocationInfoクラスのインスタンス。

LocationInfoについては[Section 4.5 LocationInfo](#) を参照。

取得出来なかった場合はnilが返ります。

nilが返却された場合は、引数のerrorにエラー情報が含まれます。

## 7.5.5. Swiftを用いている場合

ライブラリのバージョン3.0.0にて、SwiftコードからcurrentLocationメソッドを呼び出す時の実装方法が変更となりました。

バージョン2.1.0以前のライブラリを利用している場合、以下を参照し、移行を行なってください。

### <移行手順>

- currentLocationメソッドでエラーが発生した場合、例外をスローせず、引数のerrorにエラー内容をセットするようにしました。引数のerrorがnilかどうかでエラー有無を判断してください。
- currentLocationメソッドが例外をスローすることはなくなりましたので、メソッドを囲っていたdo-try〜catchは不要となります。

### 【ライブラリバージョン2.1.0以前の実装イメージ】

```
do {
    let result = try locationManager.default()?.currentLocation(...)
    // 正常処理
} catch let error {
    let err = (error as NSError)
    // エラー処理
}
```

### 【ライブラリバージョン3.0.0以降の実装イメージ】

```
var error: NSError?
let result = try locationManager.default()?.currentLocation(..., error:
&error)
```

```
if (error == nil) {  
    // 正常処理  
} else {  
    // エラー処理  
}
```

### 7.5.6. コメント

error情報については[Section 8.1.4.3](#) パラメータ のエラーコード一覧を参照。  
取得出来た場合、errorはnilで返ります。

## 7.6. librarySettings

### 7.6.1. 説明

ライブラリに設定されている各設定値を返します。

### 7.6.2. プロトタイプ宣言

```
-(NSDictionary *) librarySettings:(BOOL)isOperationValue;
```

### 7.6.3. パラメータ

isOperationValue [I] : 取得する設定情報を指定します。  
YESを指定すると動作値を返します。  
NOを指定すると設定値を返します。

### 7.6.4. 戻り値

measuringStartメソッドで指定するNSDictionaryと同じ情報を返します。

詳細については[Section 7.2.3](#) パラメータ を参照。

未設定の値について、初期値がある設定は初期値が返却され、初期値が無い設定は返却する情報に含まれません。

### 7.6.5. コメント

本メソッドはmeasuringStartメソッド実行後に正常に値が取得できるメソッドです。

## 8. プロトコル

表 16に位置測位ライブラリのプロトコル一覧を示します。

表 16. プロトコル一覧

プロトコル	メソッド	実装の必要性	概要
LocationManagerDelegate	- measuringResult:status:error:	required	測位情報を通知するコールバック関数
	- changeMeasureStatus:reason:	optional	測位方法変更を通知するコールバック関数

### 8.1. LocationManagerDelegate

#### 8.1.1. 説明

LocationManagerクラスのデリゲートのプロトコル。

#### 8.1.2. プロトコル宣言

```
@protocol LocationManagerDelegate <NSObject>
```

#### 8.1.3. コメント

なし

## 8.1.4. measuringResult:status:error:

### 8.1.4.1. 説明

測位情報を通知します。

### 8.1.4.2. プロトタイプ宣言

```
- (void)measuringResult:(LocationInfo *)locationInfo  
status:(LocationManagerStatus)status error:(NSError *)error;
```

### 8.1.4.3. パラメータ

result [I] : 測位情報LocationInfoクラスのインスタンス  
LocationInfoについては [Section 4.5 LocationInfo](#) を参照

status [I] : 測位結果のステータス

error [I] : 測位失敗時のエラーコード

コールバック時に各パラメータに設定される結果を [表 17](#) に示します。

表 17. 測位結果

結果	result	status	error
測位成功	測位情報	0 : LocationManagerStatusSuccess	nil
測位失敗	nil	2 : LocationManagerStatusError	エラーコード

[表 18](#) にエラーコード一覧を示します。

表 18. 測位後のエラー情報

No.	エラーコード	値	説明
1	kCLErrorLocationUnknown	0	位置が取得できない
2	kCLErrorDenied	1	位置情報サービスの利用を許可されていない
3	kCLErrorHeadingFailure	3	デバイスの正確な向きが識別できない
4	上記以外	上記以外	その他のエラー

### 8.1.4.4. 戻り値

なし

### 8.1.4.5. コメント

なし

## 8.1.5. changeMeasureStatus:reason:

### 8.1.5.1. 説明

測位方法が変更したことを通知します。

標準位置情報サービスと大幅変更位置情報サービスを使用した測位方式で動作している場合に通知されます。

### 8.1.5.2. プロトタイプ宣言

```
- (void)changeMeasureStatus:(MeasuringType)measureStatus  
reason:(MeasuringTypeChangeReason)reason;
```

### 8.1.5.3. パラメータ

measureStatus [I] : 測位方法の種類を返します  
測位方法の種類を [表 19](#) に示します

reason [I] : 測位方法が変更した理由を返します  
変更理由の種類を [表 20](#) に示します

表 19. 測位方法の種類

No.	測位方法の種類	定義名 (※1)	値
1	標準位置情報サービス測位	MeasuringType_Standard	0
2	大幅変更位置情報サービス測位	MeasuringType_Significant	1

表 20. 測位方法変更理由

No.	測位方法変更理由の種類	定義名 (※1)	値
1	バッテリー残量が閾値よりも低下、もしくは増加	MeasuringTypeChange_Battery	0

(※1) : KeyはLocationManager.hに定義された文字列定数を使用するようにする

### 8.1.5.4. 戻り値

なし

### 8.1.5.5. コメント

なし

## 9. シーケンス

測位開始～位置取得～測位停止のシーケンスを下記に示す。

