
iOS 住宅 API ライブラリ リファレンス

Version 1.0

目次

1. はじめに.....	1
2. ライブラリ基本情報	1
2.1. 動作環境	1
2.2. ライセンス	2
3. ライブラリの利用方法.....	3
3.1. ライブラリの利用方法.....	3
4. リファレンス.....	5
4.1. コマンドを生成する	5
4.2. コマンドを送信する	6
4.3. 応答メッセージを取得する.....	7
4.4. サンプルコード	11
5. おわりに.....	12
5.1. 著作権・商標について	12

1. はじめに

本ライブラリは、住宅 API を利用したコマンドの生成と応答メッセージの解析を容易に実現する事を目的とした、iOS 向けのライブラリです。

- ・ パラメータを指定する事で、住宅 API のコマンドの生成を容易にします。
- ・ IP アドレス、ポート番号、コマンドのパラメータを指定する事で、コマンドの送信と受信した応答メッセージの解析をライブラリ側で行います。
- ・ 本ライブラリの各処理は同期処理です。

2. ライブラリ基本情報

2.1. 動作環境

本ライブラリは、以下の環境で利用可能です。

開発環境	Xcode 6.1.1
OS	iOS 8.1
言語	objective-c
ARC	有効
対象コマンド	住宅 API version 1.3
応答メッセージの形式	XML

2.2. ライセンス

本ライブラリは BSD ライセンスのもとで公開されています。

Copyright (c) 2015, Daiwa House Industry Co., Ltd.
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the <organization> nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL <COPYRIGHT HOLDER> BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

(日本語訳)

ソースコード形式かバイナリ形式か、変更するかしないかを問わず、以下の条件を満たす場合に限り、再頒布および使用が許可されます。

- * ソースコードを再頒布する場合、上記の著作権表示、本条件一覧、および下記免責条項を含めること。
- * バイナリ形式で再頒布する場合、頒布物に付属のドキュメント等の資料に、上記の著作権表示、本条件一覧、および下記免責条項を含めること。
- * 書面による特別の許可なしに、本ソフトウェアから派生した製品の宣伝または販売促進に、<組織>の名前またはコントリビューターの名前を使用してはならない。

本ソフトウェアは、著作権者およびコントリビューターによって「現状のまま」提供されており、明示黙示を問わず、商業的な使用可能性、および特定の目的に対する適合性に関する暗黙の保証も含め、またそれに限定されない、いかなる保証もありません。著作権者もコントリビューターも、事由のいかんを問わず、損害発生の原因いかんを問わず、かつ責任の根拠が契約であるか厳格責任であるか(過失その他の)不法行為であるかを問わず、仮にそのような損害が発生する可能性を知らされていたとしても、本ソフトウェアの使用によって発生した(代替品または代用サービスの調達、使用の喪失、データの喪失、利益の喪失、業務の中断も含め、またそれに限定されない)直接損害、間接損害、偶発的な損害、特別損害、懲罰的損害、または結果損害について、一切責任を負わないものとします。

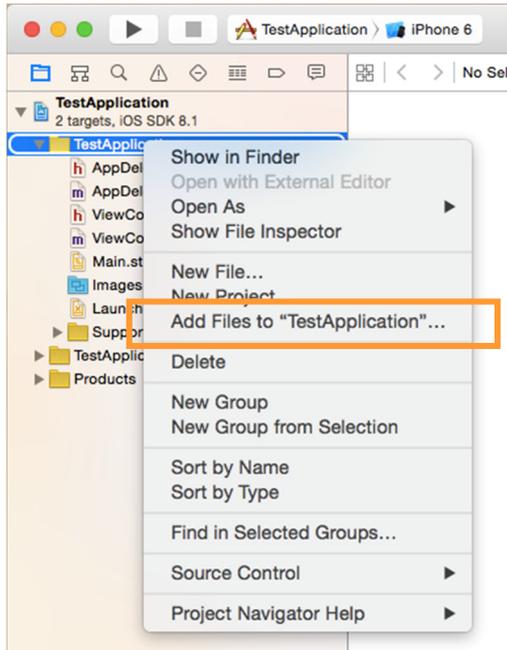
3. ライブラリの利用方法

Xcode 6.1.1 を使用してアプリケーションにライブラリを追加する方法は、以下のとおりです。

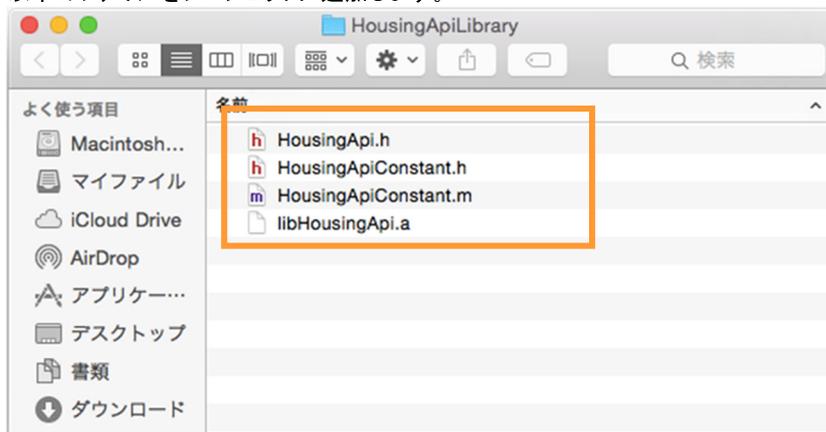
3.1. ライブラリの利用方法

3.1.1. プロジェクトにライブラリを追加する

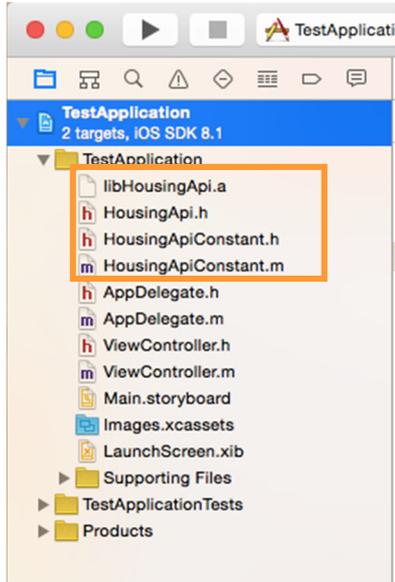
1. アプリケーションのプロジェクトフォルダを右クリックし、[Add Files to "アプリケーションのプロジェクト名"]を選択します。



2. 以下のファイルをプロジェクトに追加します。



ファイル追加後のプロジェクトの状態



3. ライブラリを利用するクラスで、[HousingApi.h]をインポートします。
4 リファレンスに記載の各処理が利用できるようになります。

```
1 //  
2 // ViewController.m  
3 // TestApplication  
4 //  
5 // Created by DaiwaHouse on 2015/02/05.  
6 // Copyright (c) 2015 DaiwaHouse. All rights reserved.  
7 //  
8  
9 #import "ViewController.h"  
10 #import "HousingApi.h"  
11  
12 @interface ViewController () {  
13 }  
14 }  
15  
16 @end  
17  
18 @implementation ViewController  
19  
20 - (void)viewDidLoad {
```

4. リファレンス

4.1. コマンドを生成する

4.1.1. コマンドの生成方法

コマンドは、HousingApiConstant に定義された固定値を組み合わせて作成します。

- **Sample**

```
//主幹瞬時電力(W)のコマンドの生成
// /smart/rest/request?deviceid=lite.boardMeter_1_1&type=get&key=instantPower
NSString* command = [NSString stringWithFormat:kLocalCommandRestGet
                    ,kLocalDeviceIdMainBoard
                    ,kLocalKeyBoardInstantPower];
```

```
//エアコンの動作状態、温度設定、湿度設定、動作モード、風量設定、スイング設定を取得
// /smart/rest/request?deviceid=lite.aircon_1_1&type=get&
    key=operationStatus,temeprature,humidity,currentMode,airFlow,swing

//取得対象の全ての Key 値を配列に格納
NSArray* airconKeys = @[kLocalKeyAirconOperationStatus, kLocalKeyAirconTemperature,
                        kLocalKeyAirconHumidity, kLocalKeyAirconCurrentMode,
                        kLocalKeyAirconAirFlow, kLocalKeyAirconSwing];

//カンマ区切りの文字列を生成
NSString* key = [airconKeys componentsJoinedByString:kLocalCommandDelimiter];
//定義値からコマンドを生成
NSString* command = [NSString stringWithFormat:kLocalCommandRestGet
                    ,kLocalDeviceIdAircon
                    ,key];
```

4.2. コマンドを送信する

4.2.1. – sendCommand: port: command: timeoutInterval: error

指定の IP アドレス、ポート番号に対してコマンドを送信し、応答結果をライブラリ内部で保持します。送信が成功したかどうかを呼び元に返却します。

- **Declaration**

```
- (BOOL) sendCommand: (NSString*) ipAddress
                port: (NSString*) portNo
            command: (NSString*) command
    timeoutInterval: (NSTimeInterval) timeoutInterval
            error: (NSError**) error;
```

- **Parameters**

ipAddress

コマンドの送信先 IP アドレス

portNo

コマンドの送信先ポート番号

command

送信するコマンド

timeoutInterval

コマンドを送信する時にタイムアウトするまでの時間。秒単位で設定可能です。

error

メソッドを実行した結果、エラーが発生した場合にエラーオブジェクトが格納されます。メソッドの呼び元から参照可能です。

- **Return value**

コマンドの送信結果

送信に成功し、応答メッセージを取得する事が出来た場合は **true** を返却します。

- **Sample**

```
//画面の入力項目から IP アドレスとポート番号を取得
NSString* ipAddress = txtIpAddress.text;
NSString* portNo = txtPortNo.text;

//定義値からコマンドを生成
NSString* command = [NSString stringWithFormat:kLocalCommandRestGet
                , kLocalDeviceIdMainBoard
                , kLocalKeyBoardInstantPower];

HousingApi* housingApi = [[HousingApi alloc] init];
//IP アドレス、ポート番号、コマンドを渡すと、ライブラリ内で HTTP リクエストを生成して送信し、
//応答メッセージを解析する。
NSError* error = nil;
BOOL connectResulst = [housingApi sendCommand:ipAddress
                port:portNo
            command:command
    timeoutInterval:30
            error:&error];
```

4.3. 応答メッセージを取得する

4.3.1. - getResponseResult:

コマンド送信後、受信した応答メッセージから応答結果を表すタグの情報を取得します。
引数には、応答結果のタグまでの全てのタグを記述します。

- **Declaration**

```
- (BOOL) getResponseResult: (NSString*) rootTag  
    , ...NS_REQUIRES_NIL_TERMINATION;
```

- **Parameters**

rootTag

XML 形式の応答メッセージのルートのタグ

...

応答結果が格納されているタグまでの、ルートタグ以降のタグを全て指定します。

- **Return value**

引数に指定されたタグの値を BOOL 値に変換して返却します。
タグの指定に誤りがあると、正しい応答結果を取得する事は出来ません。
応答メッセージが受信できていない場合は、**false** を返却します。

- **Sample**

```
//<result>タグの情報を取得します。引数には、<result>タグまでの XML タグを  
//全て指定します。「resultset」「result」を設定しています。  
BOOL result = [housingApi getResponseResult:kLocalElementResultset  
                , kLocalElementResult  
                , nil];
```

4.3.2. – getResponseError:

コマンド送信後、受信した応答メッセージからエラーメッセージを表すタグの情報を取得します。引数には、エラーメッセージのタグまでの全てのタグを記述します。

※4.2.1– sendCommand: port: command: timeoutInterval: error の戻り値が false の場合の、エラー判定には本メソッドではなく、– sendCommand: port: command: timeoutInterval: error のパラメータ **error** を使用する事を推奨します。

- **Declaration**

```
- (BOOL) getResponseError: (NSString*) rootTag  
    , ...NS_REQUIRES_NIL_TERMINATION;
```

- **Parameters**

rootTag

XML 形式の応答メッセージのルートのタグ

...

エラーメッセージが格納されているタグまでの、ルートタグ以降のタグを全て指定します。

- **Return value**

引数に指定されたタグの値を NSString に変換して返却します。タグの指定に誤りがあると、正しい応答結果を取得する事は出来ません。応答メッセージが取得できていない場合は、**-9999** を返却します。

- **Sample**

```
//<message>タグの情報を取得します。 引数には、<message>タグまでの XML タグを  
//全て指定します。「resultset」「message」を設定しています。  
NSString* errorMsg = [housingApi getResponseError:kLocalElementResultset  
    , kLocalElementMessage  
    , nil];
```

4.3.3. – responseData:

コマンド送信後、受信した応答メッセージから指定のタグの情報を取得します。
引数には、取得したいデータのタグまでの全てのタグを記述します。

- **Declaration**

```
- (id) responseData: (NSString*) rootTag  
    , ...NS_REQUIRES_NIL_TERMINATION;
```

- **Parameters**

rootTag

XML 形式の応答メッセージのルートのタグ

...

エラーメッセージが格納されているタグまでの、ルートタグ以降のタグを全て記述する。

- **Return value**

引数に指定されたタグの情報を返却します。

指定のタグを持つデータの状態により、戻り値の型が変化します。

指定のタグの中に複数のタグの情報を含む場合、戻り値の型は **NSDictionary** です。

指定のタグの中に複数の値を含む場合、戻り値の型は **NSArray** です。

指定のタグが一つの値しか持たない場合、戻り値の型は **NSString** です。

※タグごとのデータの状態は、各 API の仕様書よりご確認ください。

タグの指定に誤りがあると、正しい応答結果を取得する事は出来ません。

応答メッセージが取得できていない場合や、指定のタグの情報が存在しないは、**nil**を返却します。

- **Sample**

指定のタグの中に複数のタグの情報を含む場合

```
//<data>タグ内の情報を取得します。 引数には、<data>タグまでの XML タグを  
//全て指定します。「resultset」「dataset」「data」を設定しています。  
id data = [housingApi getResponseData:kLocalElementResultset  
          , kLocalElementDataSet  
          , kLocalElementData, nil];  
  
if ([data isKindOfClass:[NSDictionary class]]) {  
    NSDictionary* dataDic = (NSDictionary*) data;  
    NSString* result = dataDic[kLocalElementResult]; //data タグ内の<result>の値  
    NSString* key = dataDic[kLocalElementKey]; //data タグ内の<key>の値  
    NSString* value = dataDic[kLocalElementValue]; //data タグ内の<value>の値  
}
```

指定のタグの中に複数の値を含む場合

```
//<data>タグ内の情報を取得します。 引数には、<data>タグまでの XML タグを全て指定しま  
す。「resultset」「dataset」「data」を設定しています。  
id data = [housingApi getResponseData:kLocalElementResultset  
          , kLocalElementDataSet  
          , kLocalElementData, nil];  
  
NSString* response = @"";  
if ([datas isKindOfClass:[NSArray class]]) {  
    NSArray* dataList = (NSArray*) datas;  
    NSInteger dataCount = [dataList count];  
    for (int i = 0; i < dataCount; i++) {  
        id data = dataList[i];  
        if ([data isKindOfClass:[NSDictionary class]]) {  
            NSString* result = data[kLocalElementResult]; //data タグ内の<result>の値  
            NSString* key = data[kLocalElementKey]; //data タグ内の<key>の値  
            NSString* value = data[kLocalElementValue]; //data タグ内の<value>の値  
        }  
    }  
}
```

4.4. サンプルコード

4.4.1. コマンドの生成から応答メッセージの受信までのサンプル

```

//画面の入力項目から IP アドレスとポート番号を取得
NSString* ipAddress = txtIpAddress.text;
NSString* portNo = txtPortNo.text;

//主幹瞬時電力(W)のコマンドの生成
// /smart/rest/request?deviceid=lite.boardMeter_1_1&type=get&key=instantPower
NSString* command = [NSString stringWithFormat:kLocalCommandRestGet
                    , kLocalDeviceIdMainBoard
                    , kLocalKeyBoardInstantPower];

HousingApi* housingApi = [[HousingApi alloc] init];
//IP アドレス、ポート番号、コマンドを渡すと、ライブラリ内で HTTP リクエストを生成して送信し、
//応答メッセージを解析する。
NSError* error = nil;
BOOL connectResult = [housingApi sendCommand:ipAddress
                             port:portNo
                             command:command
                             timeoutInterval:30
                             error:&error];

//送信結果をチェック
if (connectResult) {
    //<result>タグの情報を取得します。 引数には、<result>タグまでの XML タグを
    //全て指定します。「resultset」「result」を設定しています。
    BOOL result = [housingApi getResult:kLocalElementResultset
                  , kLocalElementResult
                  , nil];

    if (result) {
        //<data>タグ内の情報を取得します。 引数には、<data>タグまでの XML タグを
        //全て指定します。「resultset」「dataset」「data」を設定しています。
        id data = [housingApi getResult:kLocalElementResultset
                  , kLocalElementDataSet
                  , kLocalElementData, nil];

        if ([data isKindOfClass:[NSDictionary class]]) {
            NSDictionary* dataDic = (NSDictionary*) data;
            NSString* result = dataDic[kLocalElementResult]; //data タグ内の<result>の値
            NSString* key = dataDic[kLocalElementKey]; //data タグ内の<key>の値
            NSString* value = dataDic[kLocalElementValue]; //data タグ内の<value>の値
        }
        //応答メッセージが取得できた場合の処理を記述
    } else {
        //<message>タグの情報を取得します。 引数には、<message>タグまでの XML タグを
        //全て指定します。「resultset」「message」を設定しています。
        NSString* errorMsg = [housingApi getResult:kLocalElementResultset
                             , kLocalElementMessage
                             , nil];

        //エラー時の処理を記述
    }
} else {
    //送信処理が失敗した時の処理
    //sendComand が返却したエラー情報から、判断して処理を行う。
    if (error.userInfo != nil) {
        NSString* errorMsg = error.userInfo[NSLocalizedStringKey];
        //エラー時の処理を記述
    }
}
}

```

5. おわりに

5.1. 著作権・商標について

本ライブラリの著作権は、大和ハウス工業株式会社が所有しています。

本ライブラリは、著作権法および国際条約の規約によって保護されています。

iOS は、米国および他の国々で登録された Apple Inc.の商標です。
その他、製品名などの固有名詞は、各社の商標または登録商標です。