



# *Barcode Reader*

バーコードリーダー仕様・操作マニュアル  
2014.6 ver.1.0

本マニュアルは、バーコードリーダー（ライブラリ）の UI 設定・機能設定についてご説明いたします。

本マニュアルをご理解していただき、ご使用をお願いいたします。

本ソースは、iPhone 版の API です。

ご利用していただき本 API をアプリ開発組み込みや本アプリ（単体）をご使用したい場合は、ご契約が必要となりますので、ご連絡を下さい。

#### 動作環境

xcode5.1 にて作業をお願いします。

#### 端末対応 os

ios 5.6.7

#### ご利用データ（ソース）

- ・ライブラリ用データ
- ・ sample 1
- ・ sample 2

## バーコードリーダーについて

本バーコードリーダーは、他のバーコードリーダー（API）にはない機能を取り入れています。

## 特 徴

- UI 設定をユーザー側で独自設定可能。
- バーコードの認識枠・カメラ view の画面リサイズが可能。
- カメラ view の移動 他

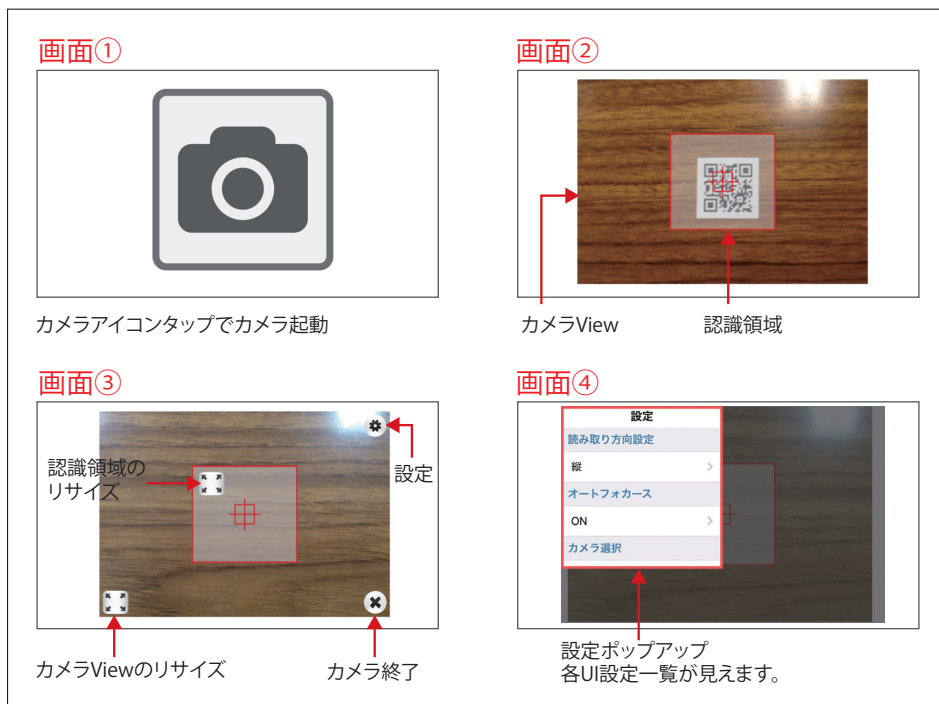
詳しいご説明は次ページ以降に記載してあります。

## 概 要

- ライブラリで提供する機能を紹介
- UI の詳細設定について説明
- 機能設定を説明
- ライブラリを組み入れるための実装を説明

## ■画面構成

下図はライブラリの画面構成です。



※画面① アプリを起動した直後の画面

- カメラボタンを押すとカメラが起動します (画面②)

※画面② 認識画面

- カメラ View のサイズ、表示位置、表示方向を設定することができます。
- 主にカメラ View と認識領域を表す赤い枠があります。
- 枠のセンターラインは認識方向を表します。
- 画面をタップするとライブラリを設定するための UI が表示されます (画面③)

※画面③ ライブラリを設定するための UI

- 主にカメラ View のリサイズ、認識領域のリサイズ、設定ポップアップを呼び出し、カメラを停止などの UI があります。
- 設定ボタンを押すと設定画面が表示されます (画面④)

※画面④ ライブラリの様々な機能を設定する画面

## ■機能概要

本ライブラリは、カメラ機能、バーコード認識機能などを設定する UI を提供するだけでなく、ユーザーが独自の UI で設定できるように様々な API を提供しています。主に次の API があります。

- デフォルト UI 全体を表示・非表示 API
- デフォルト UI の一部を表示・非表示 API
- カメラ View リサイズ API
- 認識領域リサイズ API
- カメラオン・オフ API
- 設定ポップアップ呼び出し API
- 読み取り方向設定
- フォーカス設定
- 前後カメラを選択
- カメラをズーム、認識画面でのピッチングでもズームできます。
- 読み取りバーコード種類の選択
- 認識後に鳴らすサウンドをオン・オフ
- バイブレーションをオン・オフ
- デバグ画像を表示・非表示
- カメラライトオン・オフ
- 認識結果表示・非表示

## ■ UI 設定 API

本ライブラリは、カメラの機能を設定する UI が用意されていますが、ユーザーが独自に UI を作成することも出来ます。そのために、不要のデフォルト UI を非表示することができます。

以下は本ライブラリのデフォルト UI を表示・非表示の設定を説明します。

### ■ インタフェース全体の表示・非表示 API

インタフェース全体の表示・非表示を設定するプロパティです。

参照ファイル：BarcodeReaderView.h

設定：property **ShowInterfaceMode** showInterfaceMode

設定値は下図に示します。

設定 Id	対応するボタン
AllwayShowInterface	常に UI を表示
ShowInShortTime	表示した後、自動的に非表示になります。

### ■ 個別インタフェースの表示・非表示 API

カメラ View のリサイズ、移動ボタン、認識領域リサイズボタン、設定ボタン、カメラを停止するボタンを個別に表示・非表示にする API です。

参照ファイル BarcodeReaderView.h

設定メソッド：

- (**void**)hideInterfaceWithId:(**BarcodeInterfaceId**)interfaceId hidden:(**BOOL**)hidden;

hidden=YES の場合は、ボタンを非表示にし、hidden=NO の場合はボタンを表示にします。

interfaceId は表示・非表示したいボタンの Id を表します。

interfaceId は次の表示まとめます。デフォルトでは全ての UI が表示されます。

設定 Id	対応するボタン
SettingButtonId	設定ボタン
CloseCameraButtonId	カメラ停止ボタン
ResizeScanAreaButtonId	認識領域をリサイズするボタン
ResizeCameraButtonId	カメラ View をリサイズするボタン

## ■ 各機能設定 API

### ■ カメラ起動

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)startCamera

起動完了のコールバック（オプション）：- (void)finishedStartCamera

### ■ カメラ停止

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)stopCamera

起動完了のコールバック（オプション）：- (void)finishedStopCamera

### ■ 認識領域のサイズ設定

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)setScanViewWithSize:(CGSize)size

### ■ カメラ View の枠を設定

参照ファイル：BarcodeReaderView.h

起動メソッド：CGRect frame

### ■ カメラ View の自動回転

デバイスが回転する際、自動的にデバイスの状態を取得してカメラ View を自動的に回転する API

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)autoRotate

### ■ ズーム率設定

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)zoomCameraWithZoomFactor:(CGFloat)factor

注：CGFloat factor はズーム率です。

### ■ カメラ指定

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)setCameraWithPosition:(CameraPosition)position

注：CameraPosition position は選択するカメラです。

戻り値	現在の設定モード
FrontCamera	前方カメラを使用
BackCamera	後方カメラを使用

## ■ カメラライト設定

デバイスのバックにあるライトのオン・オフを設定

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)setCameraLight:(BOOL)isOn

## ■ 音声設定

認識が成功したとき、音声を鳴らす設定

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)setSound:(BOOL)playSound

## ■ バイブレーション設定

認識が成功したとき、端末のバイブレーションを鳴らす設定

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)setVibration:(BOOL)playVibration

## ■ フォーカス設定

カメラのフォーカス調整を自動行う設定

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)setCameraFocus:(BOOL)isAutoFocus

## ■ 読み取り方向設定

バーコードの読み取り方向を設定

参照ファイル：BarcodeReaderView.h

起動メソッド：

- (void)setRegconizeDirection:(RegconizeDirection)regconizeDirection

RegconizeDirection は以下の値を設定できます。

戻り値	現在の設定モード
RegconizeOnPostrait	縦方向読み取り
RegconizeOnLandscape	横方向読み取り
RegconizeEither	縦・横両方読み取り

## ■ バーコードタイプの絞り込み

読み取り対象のバーコードタイプが決まった場合は、読み取りタイプを制限することで読み込む時間をより短縮できます。

参照ファイル：BarcodeReaderView.h

起動メソッド：

- **(void)**setRegconizeBarcode:(BarcodeFormat)barcodeFormat  
regconize:(BOOL)regconize

regconize=YES 読み込む、regconize=NO 読み込まない

**BarcodeFormat**：設定したいバーコードの種類です。次の値を設定できます。

	設定値	バーコードタイプ
0	kBarcodeFormatAztec	Aztec 2次元バーコード
1	kBarcodeFormatCodabar	CODABAR 1次元バーコード
2	kBarcodeFormatCode39	Code 39 1次元
3	kBarcodeFormatCode93	Code 93 1次元
4	kBarcodeFormatCode128	Code 128 1次元
5	kBarcodeFormatDataMatrix	Data Matrix 2次元
6	kBarcodeFormatEan8	EAN-8 1次元
7	kBarcodeFormatEan13	EAN-13 1次元
8	kBarcodeFormatITF	ITF 1次元
9	kBarcodeFormatMaxiCode	MaxiCode 2次元
10	kBarcodeFormatPDF417	PDF417 1次元
11	kBarcodeFormatQRCode	QR Code 2D
12	kBarcodeFormatRSS14	RSS 14 1次元
13	kBarcodeFormatRSSEExpanded	RSS EXPANDED 1次元
14	kBarcodeFormatUPCA	UPC-A 1次元
15	kBarcodeFormatUPCE	UPC-E 1次元
16	kBarcodeFormatUPCEANExtension	UPC/EAN extension 1次元

注：本ライブラリ Limited,Composite タイプを対応しません。



## ■ 設定画面の呼び出し

本ライブラリは上記の機能を設定するためのデフォルト UI を用意しています。独自の UI がない場合は、こちらを利用して設定することができます。

参照ファイル：BarcodeReaderView.h

起動メソッド：- (void)showSettingPopup

## ■ 認識領域の枠を独自に描画

認識領域は UIView の拡張クラスとして実装されるので、独自の UIView を置き換えれば、枠の描画、センターアイコンの描画などを独自に実装できます。

参照ファイル：CaptureView.h

設定プロパティ：@property (nonatomic, assign) UIView \*scanAreaView

## ■ 認識結果の取得

本ライブラリは上記の機能を設定するためのデフォルト UI を用意しています。独自の UI がない場合は、こちらを利用して設定することができます。

参照ファイル：CaptureView.h

起動メソッド：- (void)captureResult:(BarcodeResult\*)result

実装結果：BarcodeResult \*result;

## ■ 認識結果表示・非表示

カメラ画面上に認識の結果を表示・非表示の設定

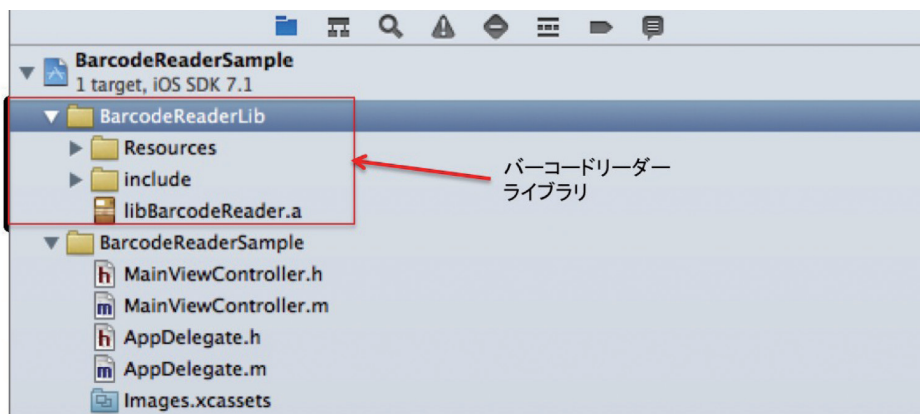
参照ファイル：CaptureView.h

設定プロパティ：BOOL showResults

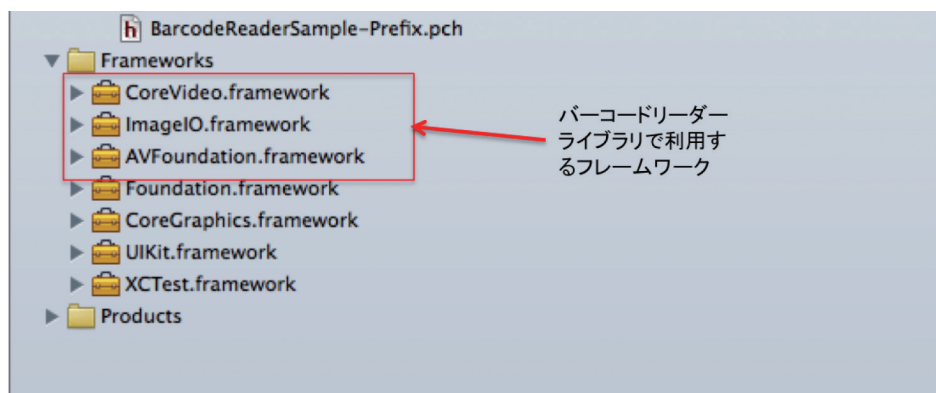
## ■ ライブラリの利用について

XCode プロジェクトに追加

下図のようにプロジェクトにライブラリのフォルダを追加



プロジェクトに必要なフレームワークを追加



## ■ 実 装

以下のようにインスタンスを生成します。

```
BarcodeReaderView *instance = [[BarcodeReaderView alloc]  
initWithFrame:rect];
```

BarcodeReaderView は UIView の派生クラスですので、使用したい UIView に subview として追加することができます。

```
[viewController.view addSubview:instance];
```

デバイスを回転させる場合は、回転する度に自動回転メソッドを実行してください。

```
[instance autoRotate];
```

注：UIViewcontroller の UIView に直接追加しない場合は、デバイスが回転した後、追加した UIView の回転に応じてカメラ View のサイズなどを再設定する必要があります。

読み込み結果を取得したい場合は delegate を設定必要があります。

```
instance.delegate = self;
```

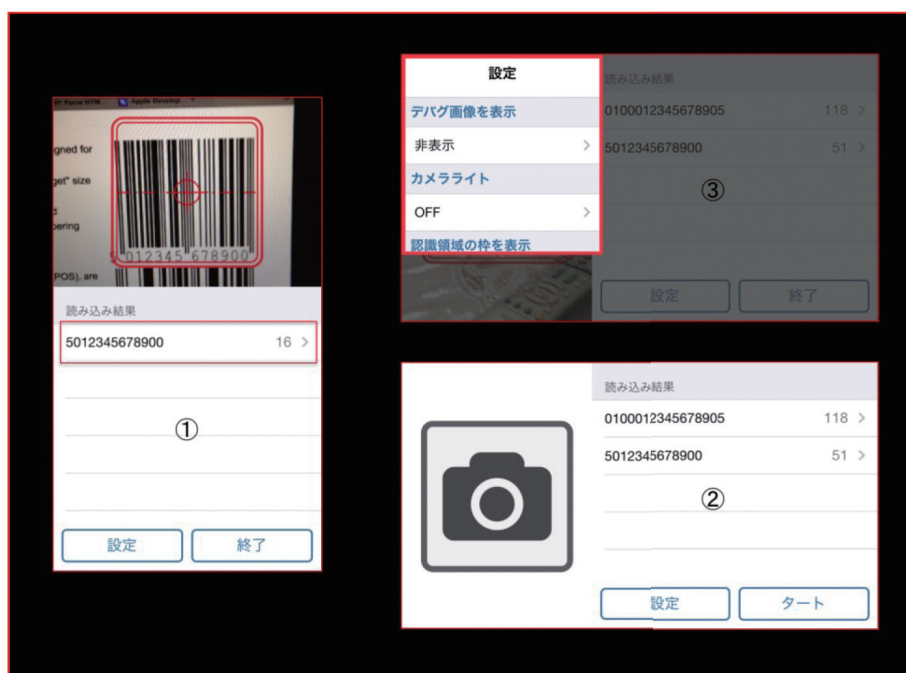
## ■ サンプル 1

サンプル 1 は下図のように読み込むカメラ、読み込んだデータの結果を表示、設定画面を表示する設定ボタン、カメラ起動終了ボタンから構成されます。

本サンプルはライブラリのデフォルト UI を使わずにアプリ側の「設定」ボタン、「スタート」ボタンで API を呼び出しています。

結果の右側の数値は、同じバーコードを読み込んだ回数（認識スピードを理解していただく為に記入してあります）です。

※ 上記の機能は、MainViewController クラスに実装されます。



## ■ カメラ View の初期化

```
// カメラのフレームを定義
CGRect rect = self.view.bounds;
rect.size.height *= CameraHeightCoe;
// インスタンスを生成
_barcodeReader = [[BarcodeReaderView alloc] initWithFrame:rect];
// ライブラリのデフォルト UI を非表示にする
_barcodeReader.showInterfaceMode = HideInterface;
// カメラ画面上に読み込み結果を表示しないように設定
_barcodeReader.showResults = NO;
// 結果を取得するために delegate を設定
_barcodeReader.delegate = self;
[self.view addSubview:_barcodeReader];
```

## ■ 結果取得

```
// BarcodeResult 読み込んだ結果
- (void)captureResult:(BarcodeResult *)result
{
    // 結果を取り込む
    [self addObject:result];
    [tableView reloadData];
}

- (void)addObject:(BarcodeResult*)result {

    for (int i=0;i<[_results count]; i++) {

        BarcodeResultEx *r = [_results objectAtIndex:i];
        // 読み込んだバーコードは既に存在するかどうかを判定
        if ([r.result.code isEqualToString:result.code] &&
            r.result.barcodeFormat==result.barcodeFormat) {
            // 読み込んだバーコードなら読み込む回数をカウントする
            r.count ++;
            [_results replaceObjectAtIndex:i withObject:r];
            return;
        }
    }

    // まだ読み込んでいないバーコードなら、結果の配列に追加する。
    BarcodeResultEx *exResult = [[BarcodeResultEx alloc] initWithResult:result];
    [_results addObject:exResult];
}
```

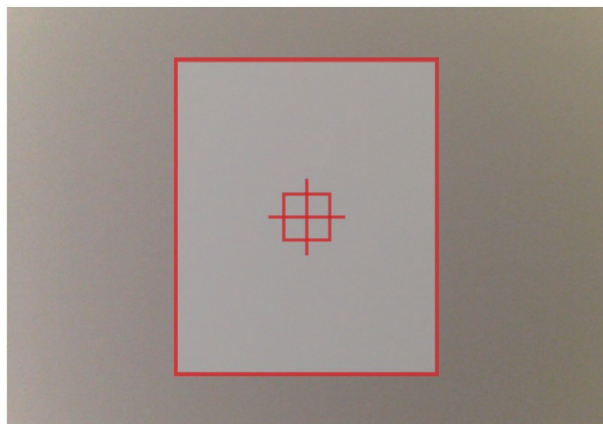
## ■ カメラ開始・終了

```
- (void)pushedClose
{
    // ボタンを押したとき、カメラが起動中なら終了させる
    if (_barcodeReader.running) {

        [_barcodeReader stopCamera];
        [_closeButton setTitle:@" タート " forState:UIControlStateNormal];
    } else {
        // ボタンを押したとき、カメラが停止中なら起動させる
        [_barcodeReader startCamera];
        [_closeButton setTitle:@" 終了 " forState:UIControlStateNormal];
    }
}
```

## ■ サンプル 2

サンプル2は、認識領域を示す枠を独自で描画するサンプルです。  
MyScanView クラスは枠を描画します。  
下図は、アプリの画面を表します。



## ■ カメラ View を初期化

```
- (void)initCamera {  
  
    // カメラのフレームを定義  
    CGRect rect = self.view.bounds;  
    rect.size.height = 0.5*rect.size.height;  
    rect.origin.y = 0.25*rect.size.height;  
  
    // インスタンスを生成  
    _barcodeReader = [[BarcodeReaderView alloc] initWithFrame:rect];  
    _barcodeReader.delegate = self;  
    [self.view addSubview:_barcodeReader];  
  
    // 認識の範囲を示す View のサイズを決定  
    rect.origin = CGPointMake(0.2*rect.size.width, 0.2*rect.size.height);  
    rect.size = CGSizeMake(0.6*rect.size.width, 0.6*rect.size.height);  
    // 枠を描画する View のインスタンスを生成  
    _scanAreaView = [[MyScanAreaView alloc] initWithFrame:rect];  
    // カメラ View に追加  
    _barcodeReader.scanAreaView = _scanAreaView;  
}
```

## ■ 読み込み可能な各サンプルバーコード

各規格バーコードサンプルを用意いたしました。



AZTEC



QR Code



DATAMATRIX



ITF



EAN13



CodaBar



ご質問・ご相談等ありましたら、お気軽にお申し付け下さい。

制作：株式会社ワトム

〒101-0054 東京都千代田区神田錦町3丁目9番地4階

電話：03-3294-4001 制作部：03-5282-4487

FAX：03-3518-4355 URL：<http://www.watom.jp/>