

# Mobilogics Barcode 5.0

Copyright 2009 - 2013 MobiLogics. All rights reserved.

Mon Oct 21 2013 15:00:44



# Chapter 1

## Mobilogics Barcode SDK Manual

### Introduction of SDK

Welcome to use Mobilogics Software Development Kit (SDK). SDK enables you to create applications that run on Mobilogics Products (iPDT5, iScan and aScan). This technology lets you build a single binary that takes you advantage of new features when running on a system that supports them.

You can find out [SDK Release Notes](#) here.

For more information please visit our web site: <http://www.mobilogics.com.tw>

Copyright (c) 2009 - 2013 MobiLogics. All rights reserved.

### How to use

#### Add Barcode Framework to your Project

Use Barcode Framework in your project, just follow these three steps:

1. Add *libstdc++.dylib* to your project.
2. Add *ExternalAccessory* framework to your project.
3. Add *Barcode* framework and *MobilogicsCore* framework to your project.

You must install Mobilogics SDK first, after you installed, you can found Barcode framework and Mobilogics-Core framework in

```
/Library/Frameworks/Mobilogics.framework/Frameworks
```

4. Add

```
#import <Barcode/Framework.h>
```

to your .pch header file.

5. Add link options to your project's **Other Linker Flags**

```
-ObjC  
-all_load
```

6. If you using SDK with Xcode 4.0 or later version.

You need to add array item which named [Supported external accessory protocols] in your Info.plist. And you need add three string item in this array as follow:

Item 0: tw.com.mobilogics.ipdt380

Item 1: tw.com.mobilogics.iscan

Item 2: tw.com.mobilogics.ascan

Item 3: tw.com.mobilogics.iPDT5

## Initialize Barcode Framework

Before you use Barcode Framework, you must initialize framework as follow code :

```
[[MLScanner sharedInstance] setup];
```

In BarcodeExample with ARC enable, we put this code on this method like this:

```
- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)
    launchOptions {

    [[MLScanner sharedInstance] setup];
    .....
    .....
    .....
}
```

## Connect & Disconnect Notification

After you initialize Barcode Framework, you will got notification when accessory connected or disconnected by follow source code:

- add follow source code to got notification from SDK:

```
[[MLScanner sharedInstance]
    addAccessoryDidConnectNotification:self];
[[MLScanner sharedInstance]
    addAccessoryDidDisconnectNotification:self];
[[MLScanner sharedInstance] addReceiveCommandHandler:self];
```

In BarcodeExample, we put this code on this method like this:

```
- (void)viewDidLoad {
    [super viewDidLoad];

    [[MLScanner sharedInstance]
        addAccessoryDidConnectNotification:self];
    [[MLScanner sharedInstance]
        addAccessoryDidDisconnectNotification:self];
    [[MLScanner sharedInstance] addReceiveCommandHandler:self]
    ;
}
```

And you must implement <NotificationHandler> protocol with two methods to handle notification:

```
-(void)connectNotify;    // When you got connected notification, framework will callback here.
-(void)disconnectNotify; // When you got disconnected notification, framework will callback here.
```

- If you want to check connection established, you can use follow source code:

```
[[MLScanner sharedInstance] isConnected]
```

You can read isConnected method on MLConnection class.\n\r

In BarcodeExample, we also use follow source code to determent connected or not:

```
if ([[MLScanner sharedInstance] isConnected]) {
    ...
} else {
    ...
}
```

## Scan

After barcode scanner already connected, you can use follow source code to scan barcode:

```
[[MLScanner sharedInstance] scan];
```

We suggested that detect accessory connected or not. If accessory disconnected then ScanShot command will be ignore.

In BarcodeExample, we put this code on this method like this:

```
- (IBAction)scan:(id)sender {
    [[MLScanner sharedInstance] scan];
}
```

Scan and Receive Barcode String are Asynchronies and independent. So you will receive barcode string not just execute ScanShot command, accessory hardware triggered scan will return barcode string too.

## Receive Barcode & get battery info

If you want to receive barcode string or get battery info, you must implement [ReceiveCommandHandler](#) protocol. It contains three methods. For more details please reference [ReceiveCommandHandler](#) protocol.

```
- (BOOL)isHandler:(ReceiveCommand *)command;
- (void)handleRequest:(ReceiveCommand *)command;
- (void)handleInformationUpdate;
```

In BarcodeExample, we use follow source code to handle or not barcode string:

```
- (BOOL)isHandler:(ReceiveCommand *)command {
    // we just handle the class which extensions ReceiveCommand class.
    if ([command isKindOfClass:[ReceiveCommand class]]) {
        return TRUE;
    }
    return FALSE;
}

- (void)handleRequest:(ReceiveCommand *)command {
    [label setText:[command getReceiveString]];
}

- (void)handleInformationUpdate {
    if ([[MLScanner sharedInstance] batteryOnCharge]) {
        [batteryChargeLabel setText:@"battery on charge"];
    } else {
        [batteryChargeLabel setText:@"battery not on charge"];
    }

    // get battery info to string, batteryCapacity will return battery percentage ex: 12.5 means 12.5%
    [batteryInfoLabel setText:[[[MLScanner sharedInstance] batteryCapacity]
        stringValue]];

    // get battery info to progress bar
    [batteryInfoProgress setProgress:([[[MLScanner sharedInstance] batteryCapacity]
        floatValue] / 100) animated:YES];
}
```

## Supported iOS old version

You can use old version frameworks on /Library/Frameworks/MobilogicsFrameworks/Support. It include our end of service old version frameworks, maybe useful for develop your apps for old device.

## Supported Mobilogics Accessory

1. iPDT380
2. iScan
3. aScan
4. iPDT5



# Chapter 2

## Release Notes

### Mobilogics SDK Release Notes

#### Version 5.0 (2013-10-12)

- Support ARM64.

#### Version 4.9.2 (2013-10-16)

- Fixed with aScan supported.

#### Version 4.9.1 (2013-10-11)

- Enhance barcode scan performance.
- Update aScan command.
- Support iPDT5.
- Drop AK110.

#### Version 4.9 (2013-08-12)

- Support models with model string.
- Support old models without model string.
- Support 2D barcode big string ( more than 4K bytes ).
- Fixed some minor scan issue and enhance scan performance.

#### Version 4.3.3 (2013-06-29)

- Fixed 2D-Code long string scan error.

#### Version 4.3 (2013-06-24)

- Fixed [MLScanner](#) interfaceOrientationNeedUpsideDown function error.
- Add option for continues barcode scan filter.
- Support new model functions.

### **Version 4.1 (2013-03-14)**

- Fixed bug of switch App from active to background and from background to active many times between in very tiny time will cause App crash because of iOS multithreads race condition error.
- Fixed bug of hardware trigger many times in very tiny time will cause call `getReceiveString` got long string error.
- Support AK110.
- Support iPDT5
- Replace `MLConnection` with new easy to use class: [MLScanner](#).
- Fixed Command Layer architecture to reduce memory used and speed up.
- Fixed and reviewed all model commands.
- Add some model's battery commands.

### **Version 4.0 (2012-10-19)**

- Support iPhone 5 armv7s platform.
- Drop support armv6 or above platform.

### **Version 3.2 (2012-07-12)**

- Support aScan.
- Support multi-accessory.
- Support iOS 6 beta 2 and later.
- Support Xcode 4.4 on OS X Lion.
- Support Xcode 4.5 beta on OS X Mountain Lion.
- Support ARC.
- Still support both Armv6 and Armv7 platform from iOS 4.x ~ 5.x
- Add Full SDK Manual within installed package.
- Remove Project Templates.

### **Version 2.4 (2012-04-25)**

- Fixed compatible with Apple MessageUI framework and change `Connection` to `MLConnection`.
- Fixed some minor issue on `BarcodeExample`.
- Changed release package mode.

### **Version 2.3 (2012-03-16)**

- Fixed some minor bug on iOS5.x sleep mode.
- Support both Armv6 and Armv7 platform from iOS 4.x ~ 5.x.

### **Version 2.2 (2011-12-22)**

- Fixed Barcode Example source code can't work on iOS 5.0.1.
- Fixed iScan on iPad2 iOS 5.x will return empty string



### **Version 2.1 (2011-10-07)**

- Fixed Barcode Framework CPU & Memory extra loading bug.

### **Version 2.0.2 (2011-10-05)**

- Fixed BarcodeExample source code can't running bug.

### **Version 2.0.1 (2011-07-08)**

- Fixed Connection object's behavior to balance cpu loading.
- Clone Connection class to MBConnection class to avoid some other framework had the same name class issue.

### **Version 2.0 (2011-05-09)**

- Fixed Connection connectedAccessory method bug.
- Support Xcode 4.x (include Xcode 4.1 preview 4).

### **Version 1.5 (2011-04-18)**

- Fixed iPad/iPhone/iPod connection compatible bug.
- Add more compatible with iOS 4.3.x.

### **Version 1.4 (2011-03-31)**

- Add Accessory connection sensitive.

### **Version 1.3 (2011-03-17)**

- Reduce Barcode Framework.
- Restructure Barcode Framework.
- Add External Accessory to DevTool Framework.

### **Version 1.2 (2011-02-19)**

- Fixed minor bugs.

### **Version 1.1 (2011-02-18)**

- SDK Lite add Barcode Example.
- Fixed SDK install scripts error.
- Support iOS v4.3 beta.

### **Version 1.0 (2011-02-16)**

- Fixed minor bugs.

### **Version 1.0RC2 (2011-01-21)**

- Add Barcode Example template.
- Fixed .pch header import setting.

### **Version 1.0RC1 (2011-01-21)**

- Replace NSLog with LogError, LogWarn, LogInfo, LogVerbose functions.
- Fixed Navigation-based template fit new functions.
- Fixed DevTool framework header file to fit template.
- Fixed Barcode framework header file to fit template.
- Add OpenGL ES Application template support.
- Add Split View-based template support.
- Add Tab Bar Application template support.
- Add Utility Application template support.
- Add View-based Application template support.
- Add Window-based Application template support.

### **Version 0.9a.4 (2011-01-07)**

- Fixed minor bugs.

### **Version 0.9a.3 (2011-01-07)**

- Fixed frameowrk to support arm6 platform.

### **Version 0.9a.2 (2011-01-04)**

- Fixed minor bugs.

### **Version 0.9a.1 (2011-01-03)**

- Fixed Navigation-based Application template about within build Unit Test error.

### **Version 0.9a (2010-12-30)**

- Add GHUnit framework to replace GTM Unit Test framework.
- Barcode framework change to new framework release type.
- DevTool framework change to new framework release type.
- Change Navigation-based Application template to fit all frameworks.

### **Version 0.9 (2010-12-28)**

- Fixed Unit Test in iOS 4.2.1 compile error. After this version developer should compile in simulator mode before compile in device mode, because Unit Test can't show assert error in compile in device mode.
- Add MLog to replace NSLog.
- Fixed Navigation-based Application template error.
- Some other template in this version will not work completely, will fixed it next release.

# Chapter 3

## License

### Mobilogics

#### Mobilogics SDK SOFTWARE LICENSE AGREEMENT

PLEASE READ THIS SOFTWARE LICENSE AGREEMENT ("LICENSE") CAREFULLY BEFORE USING Mobilogics SDK. BY USING Mobilogics SDK AS APPLICABLE, YOU ARE AGREEING TO BE BOUND BY THE TERMS OF THIS LICENSE. IF YOU DO NOT AGREE TO THE TERMS OF THIS LICENSE, DO NOT USE THE Mobilogics SDK.

#### Intellectual Property

As between you and Mobilogics, you agree and acknowledge that Mobilogics owns all rights, title and interest in Mobilogics SDK, including without limitation all associated Intellectual Property Rights. "Intellectual Property Rights" means any and all rights existing from time to time under patent law, copyright law, trade secret law, trademark law, unfair competition law, and any and all other proprietary rights, and any and all applications, renewals, extensions and restorations thereof, now or hereafter in force and effect worldwide. You agree to not remove, obscure, or alter Mobilogics or any third party's copyright notice, trademarks, or other proprietary rights notices affixed to or contained within or accessed in conjunction with or through Mobilogics SDK.

You agree that the Mobilogics SDK contain proprietary content, information and material that is owned by Mobilogics and/or its licensors, and is protected by applicable intellectual property and other laws, including but not limited to copyright. You agree that you will not use such proprietary content, information or materials other than for permitted use of Mobilogics SDK or in any manner that is inconsistent with the terms of this License or that infringes any intellectual property rights of a third party or Mobilogics.

#### Limitation of Liability.

TO THE EXTENT NOT PROHIBITED BY APPLICABLE LAW, IN NO EVENT SHALL Mobilogics BE LIABLE FOR PERSONAL INJURY, OR ANY INCIDENTAL, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES WHATSOEVER, INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF PROFITS, CORRUPTION OR LOSS OF DATA, FAILURE TO TRANSMIT OR RECEIVE ANY DATA, BUSINESS INTERRUPTION OR ANY OTHER COMMERCIAL DAMAGES OR LOSSES, ARISING OUT OF OR RELATED TO YOUR USE OF OR INABILITY TO USE THE Mobilogics SDK AND SERVICES OR ANY THIRD PARTY SOFTWARE OR APPLICATIONS IN CONJUNCTION WITH THE Mobilogics SDK, HOWEVER CAUSED, REGARDLESS OF THE THEORY OF LIABILITY (CONTRACT, TORT OR OTHERWISE) AND EVEN IF Mobilogics HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOME JURISDICTIONS DO NOT ALLOW THE LIMITATION OF LIABILITY FOR PERSONAL INJURY, OR OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS LIMITATION MAY NOT APPLY TO YOU.



## Chapter 4

# Hierarchical Index

### 4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

MLScanner . . . . .	17
<NotificationHandler> . . . . .	21
<ReceiveCommandHandler> . . . . .	22
<ReceiveCommandProtocol> . . . . .	23
ReceiveCommand . . . . .	22



## Chapter 5

# Class Index

### 5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">MLScanner</a>	Create and maintain accessory connection. After you had connection, you can scan barcode through iPDT380, iScan or AScan . . . . .	17
<a href="#">&lt;NotificationHandler&gt;</a>	iPDT380, iScan or aScan connected or disconnected event handler must implement this protocol	21
<a href="#">ReceiveCommand</a>	After barcode scan, you will receive a <a href="#">ReceiveCommand</a> . . . . .	22
<a href="#">&lt;ReceiveCommandHandler&gt;</a>	Receive command handler must implement this protocol . . . . .	22
<a href="#">&lt;ReceiveCommandProtocol&gt;</a>	<a href="#">ReceiveCommand</a> Protocol . . . . .	23





# Chapter 6

## Class Documentation

### 6.1 MLScanner Class Reference

Create and maintain accessory connection. After you had connection, you can scan barcode through iPDT380, iScan or AScan.

Inherits MLEABase.

#### Instance Methods

- (void) - [setup](#)
- (NSArray \*) - [connectedAccessory](#)
- (BOOL) - [isConnected](#)
- (void) - [addAccessoryDidConnectNotification:](#)
- (void) - [removeAccessoryDidConnectNotification:](#)
- (void) - [addAccessoryDidDisconnectNotification:](#)
- (void) - [removeAccessoryDidDisconnectNotification:](#)
- (void) - [addReceiveCommandHandler:](#)
- (void) - [removeReceiveCommandHandler:](#)
- (void) - [scan](#)
- (void) - [batteryRemain](#)
- (void) - [chargeBattery](#)
- (void) - [vibraMotorStrength:](#)
- (void) - [beepSwitch:](#)
- (BOOL) - [interfaceOrientationNeedUpsideDown](#)
- (NSNumber \*) - [batteryCapacity](#)
- (BOOL) - [batteryOnCharge](#)
- (void) - [preventFastInputWithinFastContinueInput:](#)
- (BOOL) - [isPreventFastInput](#)
- (void) - [configSyncSwitch:](#)

#### Class Methods

- (id) + [sharedInstance](#)

#### 6.1.1 Detailed Description

Create and maintain accessory connection. After you had connection, you can scan barcode through iPDT380, iScan or AScan.

Connection is an Translate Layer, it can used to create, maintain connection and execute commands which is define in Command Layer.

## 6.1.2 Method Documentation

### - (void) addAccessoryDidConnectNotification: (NSObject <NotificationHandler> \*) *handler*

Add handler to handle iPDT380, iScan or aScan connected event. Handler must implement [NotificationHandler](#) protocol

If you use UIView, you can add handler in viewDidLoad. Please remove handler in viewDidUnload.

See Also

- [removeAccessoryDidConnectNotification:](#)
- [addAccessoryDidDisconnectNotification:](#)
- [removeAccessoryDidDisconnectNotification:](#)
- [connectNotify \(NotificationHandler-p\)](#)

#### Parameters

<i>handler</i>	The object implement <a href="#">NotificationHandler</a> protocol which can handle accessory connected notification.
----------------	--

### - (void) addAccessoryDidDisconnectNotification: (NSObject <NotificationHandler> \*) *handler*

Add handler to handle iPDT380, iScan or aScan disconnected event. Handler must implement [NotificationHandler](#) protocol

If you use UIView, you can add handler in viewDidLoad. Please remove handler in viewDidUnload.

See Also

- [addAccessoryDidConnectNotification:](#)
- [removeAccessoryDidConnectNotification:](#)
- [removeAccessoryDidDisconnectNotification:](#)
- [disconnectNotify \(NotificationHandler-p\)](#)

#### Parameters

<i>handler</i>	The object implement <a href="#">NotificationHandler</a> protocol which can handle accessory disconnected notification.
----------------	---

### - (void) addReceiveCommandHandler: (NSObject <ReceiveCommandHandler> \*) *handler*

Add handler to handle receive command layer object which receive from iPDT380, iScan or aScan. Send Command and Receive Command is Asynchronous, so after you send command, you can fire and forget.

Handler must implement [ReceiveCommandHandler](#) protocol

If you use UIView, you can add handler in viewDidLoad. Please remove handler in viewDidUnload.

See Also

- [removeReceiveCommandHandler:](#)  
[ReceiveCommandHandler](#)

#### Parameters

<i>handler</i>	The object implement <a href="#">ReceiveCommandHandler</a> protocol which can handle receive command.
----------------	---

### - (NSNumber \*) *batteryCapacity*

get battery capacity. before you get this information, you must call batteryRemain function first.

#### Return values

<i>NSNumber</i>	return battery capacity information with percentage number, over 100% means in charging. For example: 12.5 means 12.5%.
-----------------	---

#### See Also

- [batteryOnCharge](#)

#### - (BOOL) batteryOnCharge

get status of battery on charge or not.

#### Return values

<i>YES</i>	battery on charge.
<i>NO</i>	battery not on charge.

#### See Also

- [batteryCapacity](#)

#### - (void) batteryRemain

Get scanner accessory battery info

#### See Also

- [batteryOnCharge](#)

- [batteryCapacity](#)

#### - (void) beepSwitch: (BOOL) status

switch scanner accessory beep on or off

#### Parameters

<i>status</i>	TRUE turn beep on.
<i>status</i>	FALSE turn beep off.

#### - (void) chargeBattery

set scanner accessory battery charge

#### - (void) configSyncSwitch: (BOOL) on

Sync Switch for 30 pin connector

#### - (NSArray \*) connectedAccessory

Get the connected accessories array.

#### Return values

<i>NSArray*</i>	The Array include connected accessories which you supported.
-----------------	--

#### - (BOOL) interfaceOrientationNeedUpsideDown

detect app layout need upside down or not

#### Return values

<i>TRUE</i>	if app layout need upsidedown
<i>FALSE</i>	if app layout do't need upsidedown

#### - (BOOL) isConnected

Detect scanner accessory is connected or not.

#### Return values

<i>TRUE</i>	If one of iPDT380, iScan or aScan is connected and work fine.
<i>FALSE</i>	If nor iPDT380, iScan or aScan is not connected, or device error. In this status, execute command will be ignore.

#### - (BOOL) isPreventFastInput

return prevent contiune fast scan process flag.

#### Return values

<i>YES</i>	trun on
<i>NO</i>	turn off

#### - (void) preventFastInputWithinFastContinuelInput: (BOOL) *on*

use this method to turn on/off prevent conitune fast scan process. If your barcode contains 'CE', please turn this option to FALSE.

#### Parameters

<i>on</i>	TRUE turn prevent conitune fast scan process on, default is on.
<i>on</i>	FALSE turn prevent conitune fast scan process off.

#### - (void) removeAccessoryDidConnectNotification: (NSObject <NotificationCenter> \*) *handler*

Remove handler not to handle iPDT380, iScan or aScan connected event After you remove handler, you will not trigger with iPDT380, iScan or aScan connected event.

See Also

- [addAccessoryDidConnectNotification:](#)
- [addAccessoryDidDisconnectNotification:](#)
- [removeAccessoryDidDisconnectNotification:](#)
- [connectNotify \(NotificationCenter-p\)](#)

#### Parameters

<i>handler</i>	The object implement <a href="#">NotificationCenter</a> protocol.
----------------	---

#### - (void) removeAccessoryDidDisconnectNotification: (NSObject <NotificationCenter> \*) *handler*

Remove handler not to handle iPDT380, iScan or aScan disconnected event. After you remove handler, you will not trigger with iPDT380 or iScan disconnected event.

See Also

- [addAccessoryDidConnectNotification:](#)
- [removeAccessoryDidConnectNotification:](#)
- [addAccessoryDidDisconnectNotification:](#)
- [disconnectNotify \(NotificationCenter-p\)](#)

#### Parameters

<i>handler</i>	The object implement <a href="#">NotificationHandler</a> protocol.
----------------	--

#### - (void) removeReceiveCommandHandler: (NSObject <ReceiveCommandHandler> \*) *handler*

Remove handler which handle receive command layer object event After you remove handler, you will not trigger with iPDT380, iScan or aScan receive command event.

See Also

- [addReceiveCommandHandler:](#)  
[ReceiveCommandHandler](#)

#### Parameters

<i>handler</i>	The object implement <a href="#">ReceiveCommandHandler</a> protocol.
----------------	--

#### - (void) scan

Tigger scanner accessory to scan barcode

#### - (void) setup

Set the framework initialize and setup. Before you start using other function, you must call this method once.

#### + (id) sharedInstance

Get MLConnection instance. You can't alloc or new MLConnection instance. You can used this method to get a MLConnection instance.

Return values

<i>MLConnection*</i>	MLConnection instance.
----------------------	------------------------

#### - (void) vibraMotorStrength: (NSUInteger) *i*

set scanner accessory vibra motor strength

Parameters

<i>i</i>	0 turn vibra motor off
<i>i</i>	1 turn vibra motor strength small
<i>i</i>	2 turn vibra motor strength medium
<i>i</i>	3 turn vibra motor strength big

## 6.2 <NotificationHandler> Protocol Reference

iPDT380, iScan or aScan connected or disconnected event handler must implement this protocol.

### Instance Methods

- (void) - [connectNotify](#)
- (void) - [disconnectNotify](#)

### 6.2.1 Detailed Description

iPDT380, iScan or aScan connected or disconnected event handler must implement this protocol.

## 6.2.2 Method Documentation

### - (void) connectNotify

When hardware connected, it will callback this method ( if you addAccessoryDidConnectNotification to ML-Connection ).

See Also

- addAccessoryDidConnectNotification: (MLScanner)

### - (void) disconnectNotify

When hardware disconnected, it will callback this method ( if you addAccessoryDidDisconnectNotification to ML-Connection ).

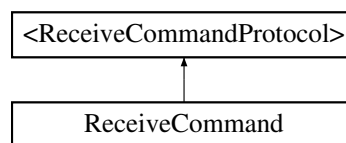
See Also

- addAccessoryDidDisconnectNotification: (MLScanner)

## 6.3 ReceiveCommand Class Reference

After barcode scan, you will receive a [ReceiveCommand](#).

Inheritance diagram for ReceiveCommand:



## Instance Methods

- (NSString \*) - [receiveString](#)

### 6.3.1 Detailed Description

After barcode scan, you will receive a [ReceiveCommand](#).

You can use [[ReceiveCommand](#) receiveString] to get barcode string.

See Also

- isHandler: (ReceiveCommandHandler-p)
- handleRequest: (ReceiveCommandHandler-p)

## 6.3.2 Method Documentation

### - (NSString \*) receiveString [required], [inherited]

Got receive barcode string.

Return values

NSString*	barcode string
-----------	----------------

## 6.4 <ReceiveCommandHandler> Protocol Reference

Recevie command handler must implement this protocol.

## Instance Methods

- (BOOL) - [isHandler:](#)
- (void) - [handleRequest:](#)
- (void) - [handleInformationUpdate](#)

### 6.4.1 Detailed Description

Recevie command handler must implement this protocol.

### 6.4.2 Method Documentation

#### - (void) [handleInformationUpdate](#)

After call [MLScanner](#) system trigger command, for example like batteryRemain. You will got Barcode Framework callback notification in this method.

#### - (void) [handleRequest:](#) ([ReceiveCommand \\*](#)) *command*

After Barcode framework recevie command and isHandler: returns TRUE, it will call this method and let handler take care this command.

Parameters

<i>command</i>	The object implement <a href="#">ReceiveCommandProtocol</a> protocol which can handle receive command.
----------------	--

See Also

- [isHandler:](#)

#### - (BOOL) [isHandler:](#) ([ReceiveCommand \\*](#)) *command*

After Barcode framework recevie command, it will callback this method to make sure that handler will handle it or not. If return TRUE, framework will call method: [handleRequest:](#).

Parameters

<i>command</i>	The object implement <a href="#">ReceiveCommandProtocol</a> protocol which can return will handle receive command or not.
----------------	---

Return values

<i>TRUE</i>	The handler will handle this command.
<i>FALSE</i>	The handler will not handle this command.

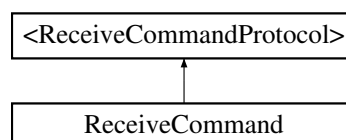
See Also

- [handleRequest:](#)

## 6.5 <ReceiveCommandProtocol> Protocol Reference

[ReceiveCommand](#) Protocol.

Inheritance diagram for <ReceiveCommandProtocol>:



## Instance Methods

- (NSString \*) - [receiveString](#)

### 6.5.1 Detailed Description

[ReceiveCommand](#) Protocol.

### 6.5.2 Method Documentation

- (NSString \*) **receiveString**      [required]

Got receive barcode string.

Return values

<i>NSString*</i>	barcode string
------------------	----------------



# Index

- <NotificationHandler>, [21](#)
- <ReceiveCommandHandler>, [22](#)
- <ReceiveCommandProtocol>, [23](#)
- addAccessoryDidConnectNotification:
  - MLScanner, [18](#)
- addAccessoryDidDisconnectNotification:
  - MLScanner, [18](#)
- addReceiveCommandHandler:
  - MLScanner, [18](#)
- batteryCapacity
  - MLScanner, [18](#)
- batteryOnCharge
  - MLScanner, [19](#)
- batteryRemain
  - MLScanner, [19](#)
- beepSwitch:
  - MLScanner, [19](#)
- chargeBattery
  - MLScanner, [19](#)
- configSyncSwitch:
  - MLScanner, [19](#)
- connectNotify
  - NotificationHandler-p, [22](#)
- connectedAccessory
  - MLScanner, [19](#)
- disconnectNotify
  - NotificationHandler-p, [22](#)
- handleInformationUpdate
  - ReceiveCommandHandler-p, [23](#)
- handleRequest:
  - ReceiveCommandHandler-p, [23](#)
- interfaceOrientationNeedUpsideDown
  - MLScanner, [19](#)
- isConnected
  - MLScanner, [20](#)
- isHandler:
  - ReceiveCommandHandler-p, [23](#)
- isPreventFastInput
  - MLScanner, [20](#)
- MLScanner, [17](#)
  - addAccessoryDidConnectNotification:, [18](#)
  - addAccessoryDidDisconnectNotification:, [18](#)
  - addReceiveCommandHandler:, [18](#)
  - batteryCapacity, [18](#)
  - batteryOnCharge, [19](#)
  - batteryRemain, [19](#)
  - beepSwitch:, [19](#)
  - chargeBattery, [19](#)
  - configSyncSwitch:, [19](#)
  - connectedAccessory, [19](#)
  - interfaceOrientationNeedUpsideDown, [19](#)
  - isConnected, [20](#)
  - isPreventFastInput, [20](#)
  - preventFastInputWithinFastContinueInput:, [20](#)
  - removeAccessoryDidConnectNotification:, [20](#)
  - removeAccessoryDidDisconnectNotification:, [20](#)
  - removeReceiveCommandHandler:, [21](#)
  - scan, [21](#)
  - setup, [21](#)
  - sharedInstance, [21](#)
  - vibraMotorStrength:, [21](#)
- NotificationHandler-p
  - connectNotify, [22](#)
  - disconnectNotify, [22](#)
- preventFastInputWithinFastContinueInput:
  - MLScanner, [20](#)
- ReceiveCommand, [22](#)
  - receiveString, [22](#)
- ReceiveCommandHandler-p
  - handleInformationUpdate, [23](#)
  - handleRequest:, [23](#)
  - isHandler:, [23](#)
- ReceiveCommandProtocol-p
  - receiveString, [24](#)
- receiveString
  - ReceiveCommand, [22](#)
  - ReceiveCommandProtocol-p, [24](#)
- removeAccessoryDidConnectNotification:
  - MLScanner, [20](#)
- removeAccessoryDidDisconnectNotification:
  - MLScanner, [20](#)
- removeReceiveCommandHandler:
  - MLScanner, [21](#)
- scan
  - MLScanner, [21](#)
- setup
  - MLScanner, [21](#)
- sharedInstance
  - MLScanner, [21](#)
- vibraMotorStrength:

MLScanner, [21](#)