

Class

XBLEManager

The `XBLEManager` class handles Bluetooth communication and provides callback blocks for printer-related statuses and operations.

```
#import <Foundation/Foundation.h>
#import <CoreBluetooth/CoreBluetooth.h>
```

XBLEManagerDelegate

Delegate Methods

`xbleDiscoverPeripheral:advertisementData:RSSI:`

Discover Bluetooth Peripheral

```
- (void)xbleDiscoverPeripheral:(CBPeripheral *)peripheral advertisementData:(NSDictionary *)advertisementData RSSI:
(NSNumber *)RSSI;
```

- **peripheral**
The discovered peripheral.
- **advertisementData**
The advertisement data of the peripheral.
- **RSSI**
The signal strength of the peripheral.

`xbleConnectPeripheral:`

Connection Successful

```
- (void)xbleConnectPeripheral:(CBPeripheral *)peripheral;
```

- **peripheral**
The connected peripheral.

`xbleFailToConnectPeripheral:error:`

Connection Failed

```
- (void)xbleFailToConnectPeripheral:(CBPeripheral *)peripheral error:(NSError *)error;
```

- **peripheral**
The peripheral that failed to connect.
- **error**
The error encountered during the connection attempt.

`xbleDisconnectPeripheral:error:`

Disconnection

```
- (void)xbleDisconnectPeripheral:(CBPeripheral *)peripheral error:(NSError *)error;
```

- **peripheral**
The disconnected peripheral.
- **error**
The error encountered during disconnection.

`xbleWriteValueForCharacteristic:error:`

Data Send Successful

```
- (void)xbleWriteValueForCharacteristic:(CBCharacteristic *)characteristic error:(NSError *)error;
```

- **characteristic**
The characteristic used to write data.

- **error**
The error encountered during the write operation, if any.

xbleReceiveValueForCharacteristic:error:

Received Printer Data

```
- (void)xbleReceiveValueForCharacteristic:(CBCharacteristic *)characteristic error:(NSError *)error;
```

- **characteristic**
The characteristic from which data was received.
- **error**
The error encountered during the receive operation.

xbleCentralManagerDidUpdateState:

Bluetooth Central Manager State Update

```
- (void)xbleCentralManagerDidUpdateState:(NSInteger)state;
```

- **state**
The updated state of the central manager.

Properties

Bluetooth Communication Class

```
@interface XBLEManager : NSObject
```

- **name**
The name of the Bluetooth device.

```
@property (nonatomic, copy) NSString *name;
```

- **isConnected**
The connection status of the Bluetooth device.

```
@property (nonatomic, assign) BOOL isConnected;
```

- **isScanning**
The current scanning status of the Bluetooth device.

```
@property (nonatomic, assign) BOOL isScanning;
```

- **writePeripheral**
The peripheral device associated with the write characteristic.

```
@property (nonatomic, strong) CBPeripheral *writePeripheral;
```

- **write_characteristic**
The characteristic used to write data to the peripheral device.

```
@property (nonatomic, strong) CBCharacteristic *write_characteristic;
```

- **read_characteristic**
The characteristic used to read data from the peripheral device.

```
@property (nonatomic, strong) CBCharacteristic *read_characteristic;
```

- **notify_characteristic**
The characteristic used to receive notifications from the peripheral device.

```
@property (nonatomic, strong) CBCharacteristic *notify_characteristic;
```

- **searchFilterUUID**
The UUID used to manually set the filter condition to find a specific printer device.

```
@property (nonatomic, strong) CBUUID *searchFilterUUID;
```

- **characteristicUUID**

The UUID of the characteristic.

```
@property (nonatomic, strong) CBUUID *characteristicUUID;
```

- **delegate**

The delegate that receives Bluetooth manager events.

```
@property (nonatomic, weak) id<XBLEManagerDelegate> delegate;
```

Callback Blocks

- **receiveBlock**

The callback block called when data is received.

```
@property (nonatomic, copy) XBLEManagerReceiveCallbackBlock receiveBlock;
```

- **writeBlock**

The callback block called when data is written.

```
@property (nonatomic, copy) XBLEManagerWriteCallbackBlock writeBlock;
```

- **statusPOSBLOCK**

The callback block called when reporting POS printer status.

```
@property (nonatomic, copy) XBLEPOSPrinterStatusBlock statusPOSBLOCK;
```

- **statusLabelBlock**

The callback block called when reporting label printer status.

```
@property (nonatomic, copy) XBLELabelPrinterStatusBlock statusLabelBlock;
```

- **snBlock**

The callback block called when reporting the printer serial number.

```
@property (nonatomic, copy) XBLEPrinterSNBlock snBlock;
```

- **cashBoxBlock**

The callback block called when reporting cash box status.

```
@property (nonatomic, copy) XBLECashBoxBlock cashBoxBlock;
```

Sure, here's the provided Objective-C code converted to Markdown format:

Method

Singleton object

```
+ (instancetype)sharedInstance;
```

Remove a delegate object

```
- (void)removeDelegate:(id<XBLEManagerDelegate>)delegate;
```

Remove all delegate objects

```
- (void)removeAllDelegates;
```

Start scanning

```
- (void)startScan;
```

Stop scanning

```
- (void)stopScan;
```

Connect to a specific device

```
- (void)connectDevice:(CBPeripheral *)peripheral;
```

Manually disconnect the root peripheral

```
- (void)disconnectRootPeripheral;
```

Send command (Write with Response)

```
- (void)writeCommandWithData:(NSData *)data;
```

Send command with receive callback (Write with Response)

```
- (void)writeCommandWithData:(NSData *)data receiveCallBack:(XBLEManagerReceiveCallBackBlock)receiveBlock;
```

Send command with write callback (Write with Response)

```
- (void)writeCommandWithData:(NSData *)data writeCallBack:(XBLEManagerWriteCallBackBlock)writeBlock;
```

Send command (Write without Response)

```
- (void)writeCommandDataWithoutResponse:(NSData *)data;
```

Set Bluetooth name and key

```
- (void)setBluetoothNameAndKeyWith:(NSString *)btName btKey:(NSString *)btKey;
```

Printer Status (for receipt printer)

```
- (void)printerPOSStatus:(XBLEPOSPrinterStatusBlock)statusBlock;
```

Printer Status (for label printer)

```
- (void)printerLabelStatus:(XBLELabelPrinterStatusBlock)statusBlock;
```

Printer Serial Number

```
- (void)printerSN:(XBLEPrinterSNBlock)snBlock;
```

Cash Box Status

```
- (void)cashBoxCheck:(XBLECashBoxBlock)cashBoxBlock;
```

Get Copyright Information

```
+ (NSString *)GetCopyRight;
```