



M3 Mobile SM10 Scanner SDK Guide book

Release date: July, 2015

©2015 M3 MOBILE Co., Ltd. All Rights Reserved.

Copyright and Agreement

WARNING: All contents of this SDK manual are protected by the copyright laws and all rights are reserved. Unauthorized distribution or copying is strictly prohibited.

M3 Mobile does not guarantee the quality and performance of the programs written in unsupported programming language. For supported development tools and languages, please refer to Development Tool and Requirements section.

Requirements

The following software must be installed

Microsoft Windows 7 (32-bit and 64-bit) or Microsoft© Windows 8 (32-bit and 64-bit) or Microsoft© Windows 8.1 (32-bit and 64-bit)

Java Development Kit (JDK) v7u7 or higher

Android Developer Tools (ADT) v22.6.0 or higher

Android SDK 4.3.1 (API Version 18)

Official site provides the Android SDK

<http://developer.android.com/>

Devices Supported

The following device has been used for validation:

SM10 - 4.3.1 (JellyBean)

Revision History

Scanner SDK Version 1.0.0

Add Official site link

Release date. 2015-05-20

Release data. 2015-07-13

Classes - APIs

- Barcode
 - void setScanner(Boolean enable)
 - void scanStart()
 - void scanDispose()
- Barcode.Symbology
 - int getCodeType(int Symbology)
 - int setCodeType(int Symbology, int value)
- Barcode.Symbology.Chinese_2of5
 - static final int nCode = 408
- Barcode.Symbology.CODABAR
 - static final int nCode = 7
- Barcode.Symbology.CODE_11
 - static final int nCode = 10
- Barcode.Symbology.CODE_128
 - static final int nCode = 8
- Barcode.Symbology.CODE_39
 - static final int nCode = 0
- Barcode.Symbology.CODE_93
 - static final int nCode = 9
- Barcode.Symbology.Discrete_2of5
 - static final int nCode = 5
- Barcode.Symbology.EAN_13
 - static final int nCode = 3
- Barcode.Symbology.EAN_8
 - static final int nCode = 4
- Barcode.Symbology.GS1_DATABAR_14
 - static final int nCode = 338
- Barcode.Symbology.GS1_DATABAR_LIMITED
 - static final int nCode = 339
- Barcode.Symbology.GS1_DATABAR_EXPANED
 - static final int nCode = 340
- Barcode.Symbology.Interleaved_2of5
 - static final int nCode = 6
- Barcode.Symbology.MSI
 - static final int nCode = 11
- Barcode.Symbology.UPC_A
 - static final int nCode = 1

- Barcode.Symbology.UPC_E
 - static final int nCode = 2
- Barcode.Symbology.UPC_E1
 - static final int nCode = 12
- BarcodeBroadcast
- BarcodeListener
 - void onBarcode(String barcode)
 - void onGetSymbology(int nSymbol, int nVal)
- BarcodeManager
 - void addListener(BarcodeListener bl)
 - void removeListener(BarcodeListener bl)
 - void dismiss()

Tutorial

1. Initialization

```
import com.m3.scannerlib.Barcode;
import com.m3.scannerlib.BarcodeListener;
import com.m3.scannerlib.BarcodeManager;
import com.m3.scannerlib.Barcode.Symbology;

private Barcode mBarcode = null;
private BarcodeListener mListener = null;
private BarcodeManager mManager = null;
private Symbology mSymbology = null;

mBarcode = new Barcode(this);
mManager = new BarcodeManager(this);
mSymbology = mBarcode.getSymbologyInstance();
mBarcode.setScanner(true);

mListener = new BarcodeListener() {
    @Override
    public void onBarcode(String strBarcode) {
        Log.i("ScannerTest", "result="+strBarcode);
    }
    @Override
    public void onGetSymbology(int nSymbol, int nVal) {
        Log.i("ScannerTest - onGetSymbology", "result="+nSymbol + ", " + nVal);
        edSymNum.setText(Integer.toString(nSymbol));
    }
};

mManager.addListener(mListener);
```

- AndroidManifest.xml

```
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
```

2. Start and Stop reading barcode

```
public void onClick(View vw) {
    int id = vw.getId();

    if(id == R.id.startread){
        mBarcode.scanStart();
    }else if(id == R.id.stopread){
        mBarcode.scanDispose();
    }
}
```

3. Barcode Type Setting (Symbology)

```
// Getter
int nValue = mSymbology.getSymbology(Barcode.Symbology.CODE_11.nCode);
// Setter
mSymbology.setSymbology(Barcode.Symbology.CODE_11.nCode, 1);
```

4. Close

```
mManager.removeListener(mListener);  
mManager.dismiss();  
mBarcode.setScanner(false);
```

APIs

- **Barcode class**

```
void setScanner(boolean enable)
```

Enable Scanner status or disable.

Parameter

enable

Set Scanner status.

Return

Void

```
void scanStart()
```

Shooting the beam for barcode reading

Parameter

None

Return

void

```
void scanDispose()
```

Stop the beam

Parameter

None

Return

void

- **Barcode.Symbology class**

```
int getCodeType(int Symbology)
```

Description

Get current Symbology status.

Callback event's returned by BarcodeListener.onGetSymbology.

Parameter

<i>Symbology</i>	Barcode.Symbology.(Symbology Classes).nCode corresponded to int type integer.
------------------	---

Return

Current Symbology status.

```
boolean setCodeType(int Symbology, int value)
```

Description

Set current Symbology status.

Callback event's returned by BarcodeListener.onGetSymbology.

Parameter

<i>Symbology</i>	Barcode.Symbology.(Symbology Classes).nCode corresponded to int type integer.
------------------	---

<i>Value</i>	Required Symbology type
--------------	-------------------------

Return

True or false

● BarcodeManager class

```
void addListener(BarcodeListener bl)
```

Description

Add User Instance of BarcodeListener class for both Barcode reading result and Symbology set result.

Parameter

<i>bl</i>	BarcodeListener class instance to get callback event
-----------	--

Return

void

```
void removeListener(BarcodeListener bl)
```

Description

Remove the user instance of BarcodeListener class added.

Parameter

bl BarcodeListener class instance to be removed.

Return

void

```
void dismiss()
```

Description

Terminate BarcodeManager.

Parameter

None

Return

void