人にやさしく、地球にやさしく
Thoughtfully, Living with the Earth
WELCOM DESIGN
http://www.e-welcom.com/
ISO14001Certified

Android プログラミング

for RS30 Mobile Computers

英文･和文で相違がある場合は、英文を優先して解釈をお願いします

Version 1.02

人にやさしく、地球にやさしく
Thoughtfully, Living with the Earth
WELCOM DESIGN
http://www.e-welcom.com/
ISO14001Certified

ウェルコムデザイン株式会社
本社 神戸市西区井吹台東町1-1-1 西神南センタービル 〒651-2242
Phone. 078-993-6010(代) Fax. 078-993-6020
東京 東京都文京区湯島3-14-9 湯島ビル 〒113-0034
Phone. 03-3836-9411(代) Fax. 03-3836-9412

改訂記録		
改訂番号	改訂日	
Rev.1.00	Jun. 2015	（初版）
Rev.1.01	Jun. 2015	RFID 制御の記述追加。
Rev.1.0.2	Jul. 2015	データ取得方法変更。
Rev.1.0.3	Oct. 2015	英語版 Version1.01 に対応。
Rev.1.0.4	Jan.2016	英語版 Version1.02 に対応。 Decoder_CodeType の戻り値を記述。

目 次

Blank page

はじめに

CipherLab 製品をご利用いただきありがとうございます。
このプログラミングガイドは、リーダモジュールを調節したり、データをキャプチャしたり、Android が搭載された RS30 シリーズモバイルコンピュータを制御したりする Android アプリケーションを構築するために必要な情報が記述されています。
Android フレームワークは、そのようなアプリケーションの作成を容易にします。作成しようとしているアプリケーションに Android コンポーネント(Android クラスライブラリ)をインポートしてください。
使用前にこのガイドを熟読され、またすぐ参照できるよう手元に置いておくことをお勧めします。

開発ツール

Android アプリケーションを開発する前に、以下の要件を自分のマシンを準備してください。
- Java SE 開発キット(JDK、Java SE 7 以上)
- Android SDK
- Android Studio または Eclipse IDE

上記のソフトウェアツールは無料で、それぞれの公式ウェブサイトからダウンロードすることができます。
開発者は、Android のプログラミングの知識を有していることを前提としています。

RFID 制御

RFID 制御につきましては、標準 API を使用していますので、下記リンクを参照してください。

http://code.tutsplus.com/tutorials/reading-nfc-tags-with-android--mobile-17278
http://www.developer.com/ws/android/nfc-programming-in-android.html
http://open-nfc.org/wp/home/documentation/
http://stackoverflow.com/questions/17990618/android-nfc-transceive-using-nfcf-tech-sony-felica

1　バーコードリーダ API

アプリケーション開発時に提供されている” barcodeapi.jar”をプロジェクトにインポートしてください。

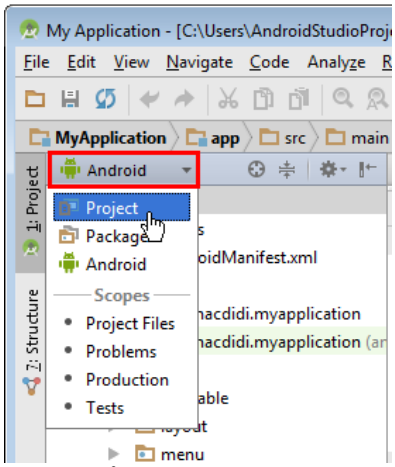
必要なライブラリ

barcodeapi.jar

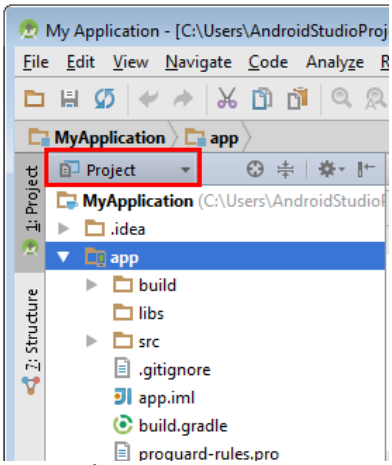
1.1　ライブラリのインポート

1.1.1　ANDROID STUDIO

(1) Android Studio プロジェクトを作成した後、従来のプロジェクトビューに切り替えるには、Android プロジェクトビューのアイコンをクリックします。

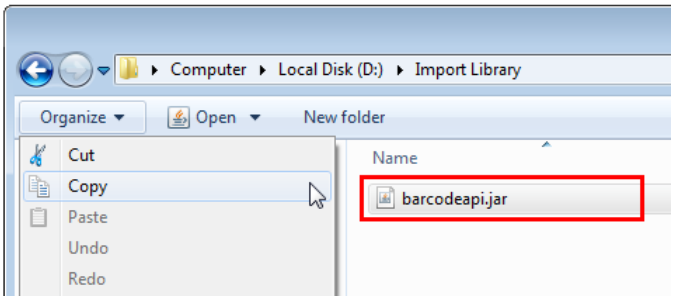


Android プロジェクトビュー

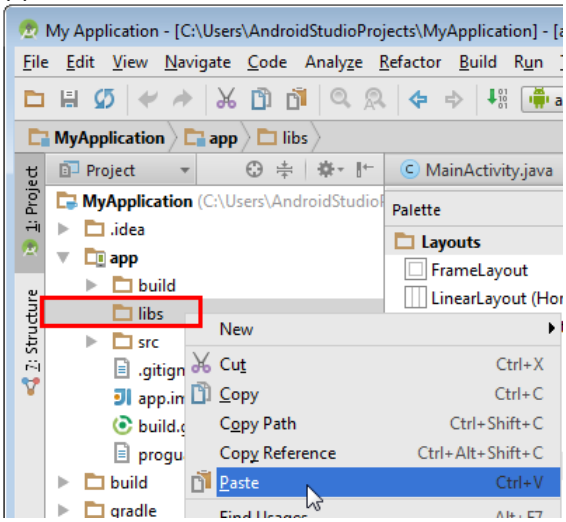


従来のプロジェクトビュー

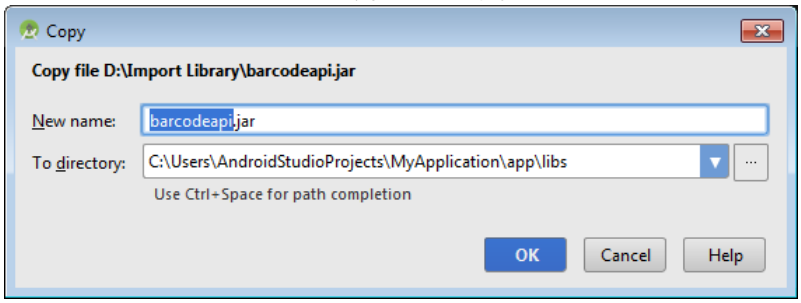
(2) ファイルシステムに提供されている"barcodeapi.jar"ライブラリファイルをコピーします。



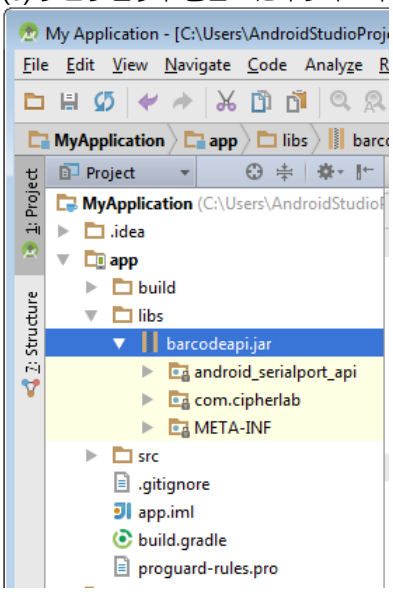
(3) プロジェクトビューの libs フォルダのところで右クリックし、Paste を選択します。



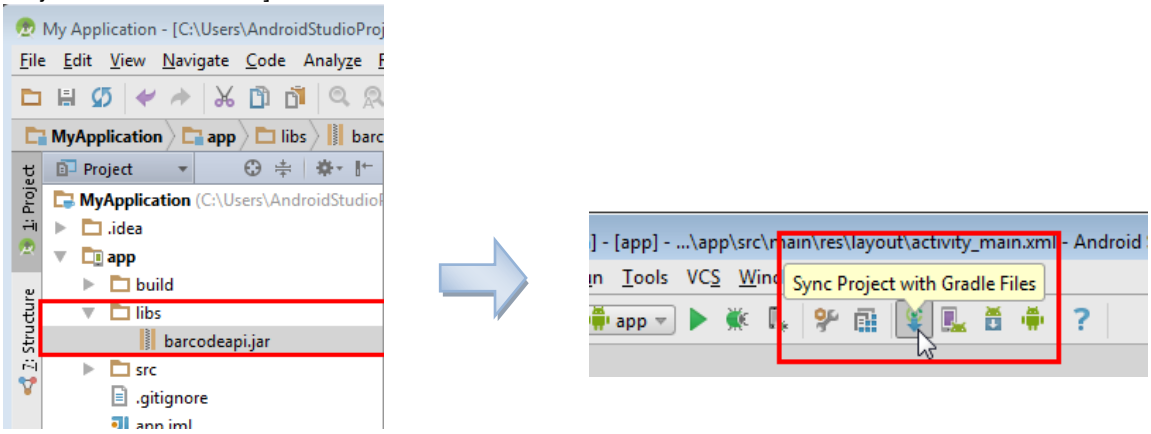
(4) ダイアログにファイル名とコピー先のディレクトリが表示されます。[OK]ボタンをクリックすると、ライブラリファイルのインポートが開始されます。



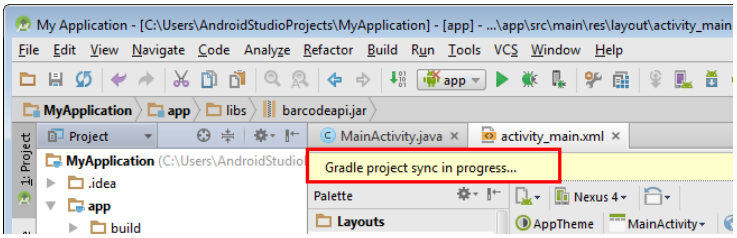
(5) プロジェクトビューにインポートされたライブラリが表示されます。



barcodeapi.jar 項目の下にリストされているすべてのファイルが表示されない場合は、ツールバーの[Sync Project with Gradle Files]ボタンをクリックしてください。

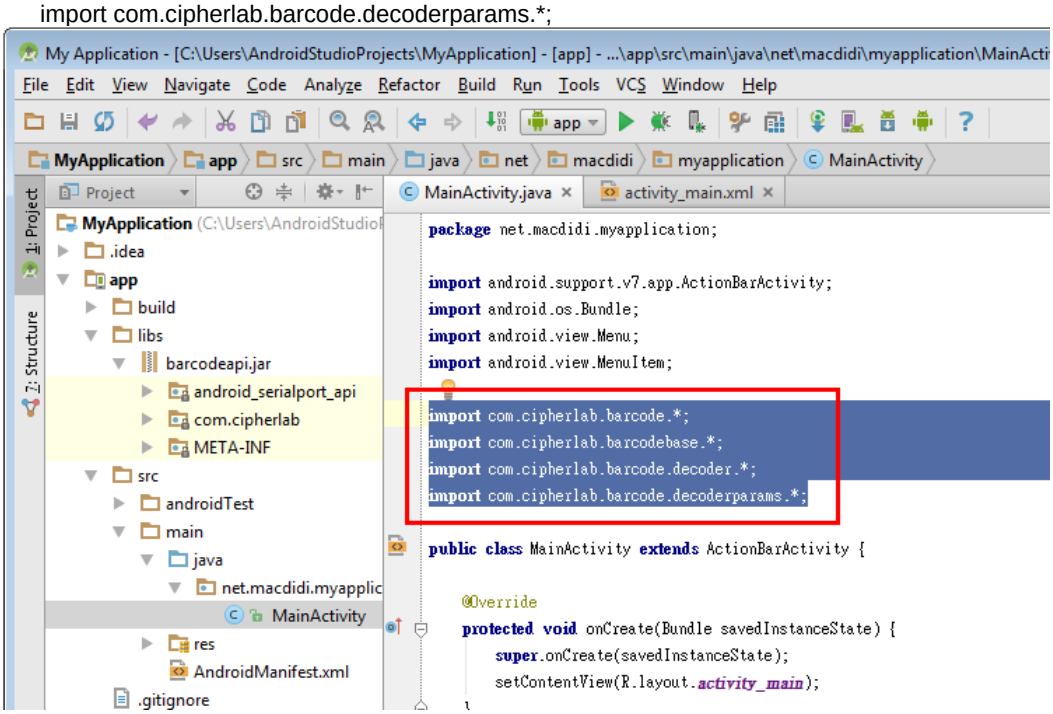


プロジェクトの同期が完了すると、ライブラリに関連したファイルが表示されます。



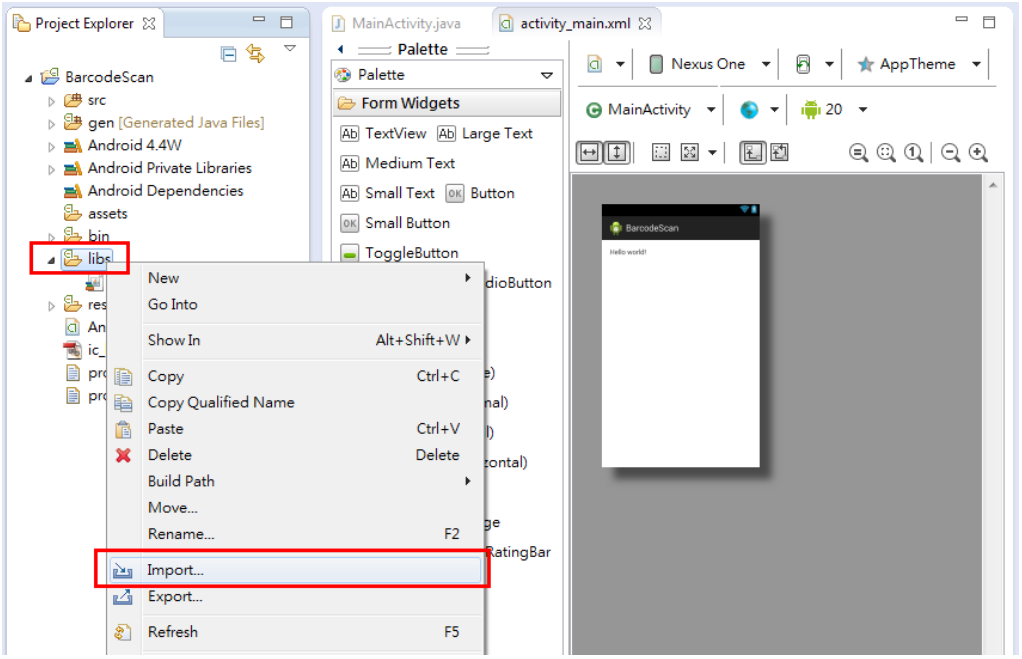
(6) 最後に、次のようにパッケージをインポートする記述を手入力すると、ライブラリのインポート処理が終了します。

```
import com.cipherlab.barcode.*;
import com.cipherlab.barcodebase.*;
import com.cipherlab.barcode.decoder.*;
import com.cipherlab.barcode.decoderparams.*;
```

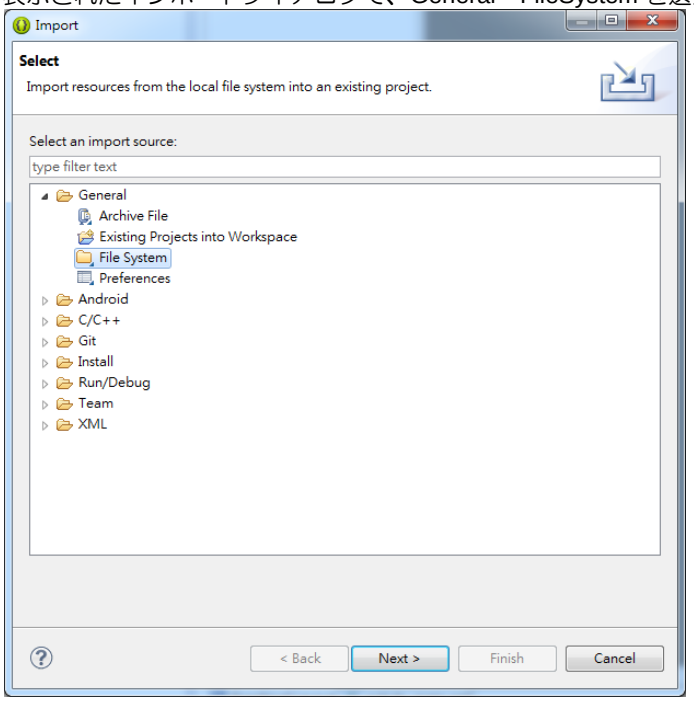


1.1.2 ECLIPSE

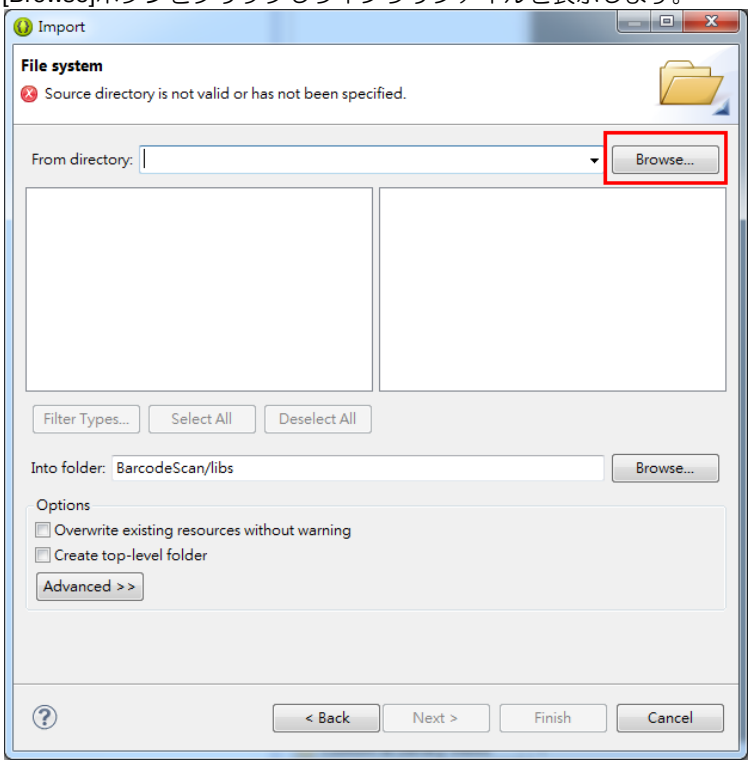
ライブラリファイル(barcodeapi.jar)をファイルシステム上に準備します。そして、以下の手順に従ってください。
(1) プロジェクトエクスプローラで、Android プロジェクトの lib を右クリックし Import を選択します。



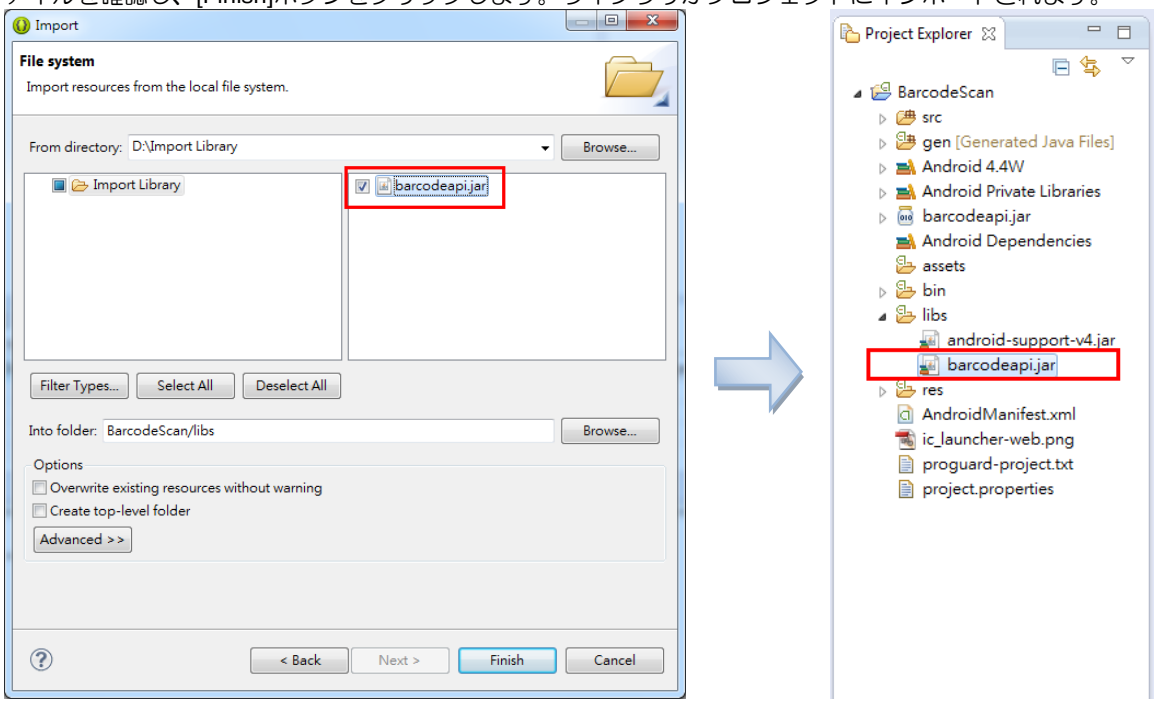
(2) 表示されたインポートダイアログで、General→FileSystem を選択し、[Next]ボタンをクリックします。



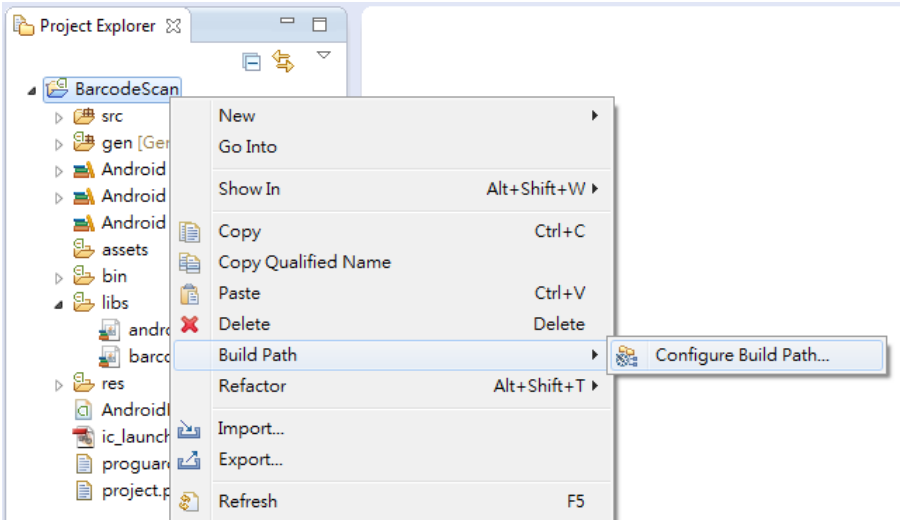
(3) [Browse]ボタンをクリックしライブラリファイルを表示します。



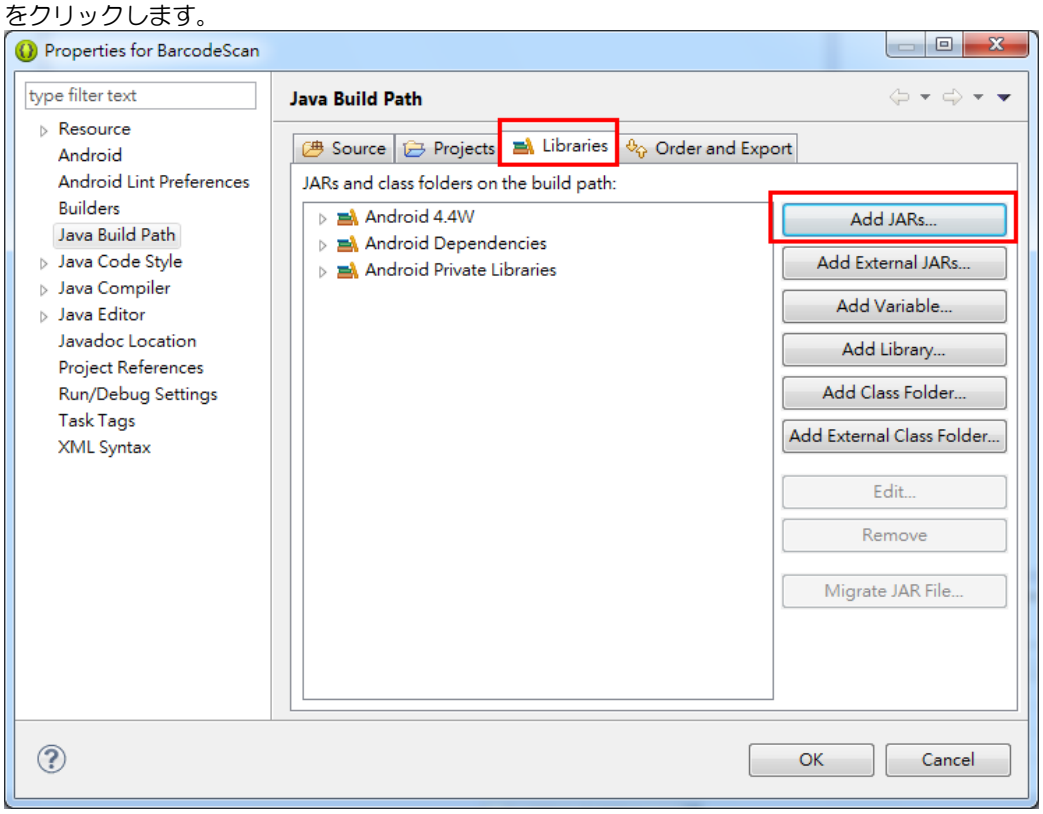
(4) 例えば、"D:\Import Library"は、ライブラリファイルが置かれているディレクトリです。右ペインの.JAR ファイルを確認し、[Finish]ボタンをクリックします。ライブラリがプロジェクトにインポートされます。



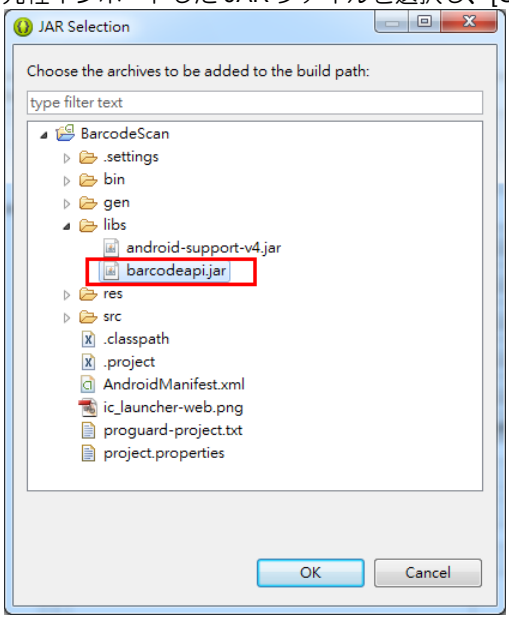
(5) ライブラリがビルドパス上にない場合。プロジェクト名を右クリックし、表示されたメニューで Build Path → Configure Build Path を選択してください。



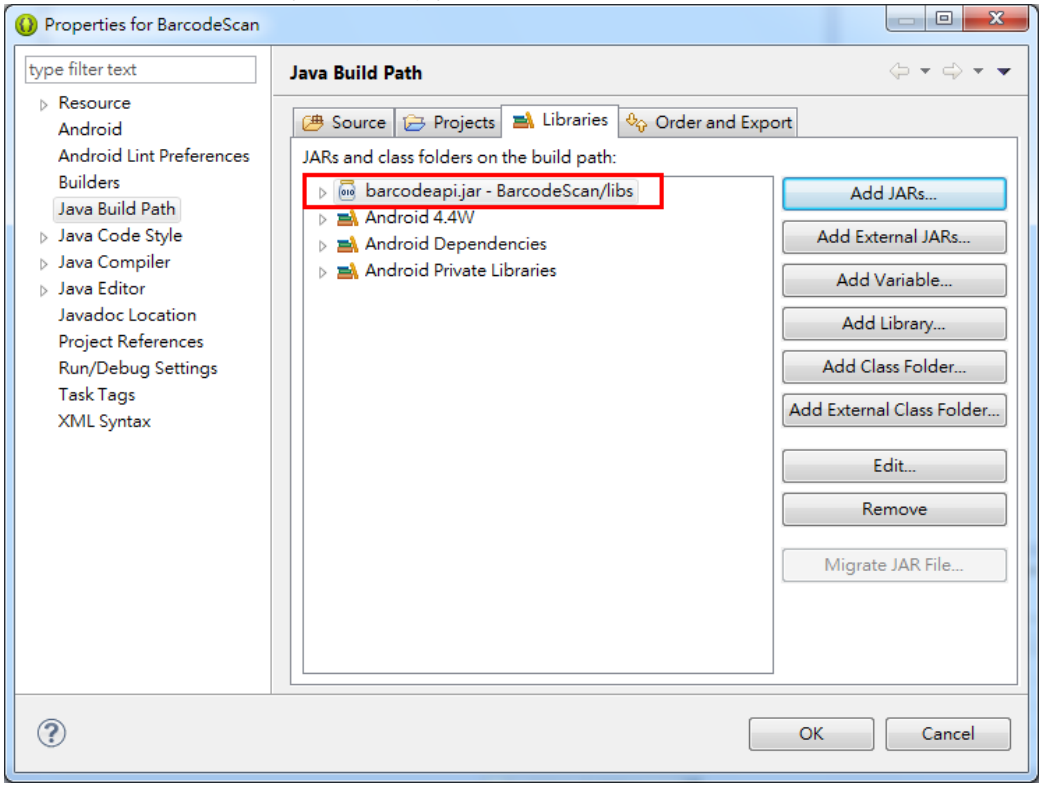
(6) プロジェクトのプロパティウィンドウが表示されますので、Libraries タブをクリックし、[Add JARs]ボタンをクリックします。



(7) 先程インポートした JAR ファイルを選択し、[OK]ボタンをクリックします。



(8) ライブラリがビルドパスに表示されます。



1.2　リーダの初期化／確認

1.2.1　初期化

InitInstance	
目的	いずれかの API を使用する前に、ReaderManager インスタンスを作成します。
書式	ReaderManager InitInstance (Context context);
使用例	private ReaderManager mReaderManager; `mReaderManager = ReaderManager.InitInstance(this);`
戻り値	成功時には ReaderManager インスタンスを取得します。失敗すると null が返ります。
備考	この関数は、リーダモジュールを準備する関数であり、他の関数の前に呼び出す必要があります。
参照	GetActive, SetActive, GetReaderType, ResetReaderToDefault

Release	
目的	リソースを解放します。
書式	void Release();
使用例	mReaderManager.Release();
参照	InitInstance, ResetReaderToDefault

1.2.2 デバイス起動

GetActive

目的 リーダの状態を取得します。

書式 boolean GetActive();

使用例 boolean bRet = mReaderManager.GetActive();

戻り値 取得に成功した場合、リーダのアクティブ状態を返します。

False	Disable
True	Enable

参照 InitInstance, SetActive, GetReaderType

SetActive

目的 使用するリーダモジュールのアクティブ状態を設定します。

書式 ClResult SetActive(boolean bActive);

引数 bActive
[入力]デバイスのアクティブ状態。

False	Disable
true	Enable

使用例
```
boolean bRet = mReaderManager.GetActive();
if (bRet==false){
    ClResult clRet = mReaderManager.SetActive(true);
}
```

戻り値 成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

備考 トリガキーのトリガイベントはアクティブなデバイスでのみ発生します。アクティブでないデバイスではイベントが発生しません。
セットするリーダが見つからない場合、エラーは発生しません。

参照 GetActive, InitInstance, GetReaderType

1.2.3 リーダ種別

GetReaderType

目的 搭載されているリーダの種別を取得します。

書式 BcReaderType GetReaderType();

使用例
```
INT nRdrType;
nRdrType = GetReaderType();
```

戻り値 取得に成功した場合、以下のリーダ種別を返します。

Moto_1D_SE955
Moto_2D_4500
CL_1D_SM1
Moto_1D_SE965

参照 InitInstance, GetActive, SetActive

1.3 データ取得

1.3.1 データ出力設定

処理データ

Set_ReaderOutputConfiguration()は、デコードされたバーコードデータにアタッチするための情報をセットします。

情報	解説
コード種別	バーコード種別。Set_ReaderOutputConfiguration()のパラメータ showCodeType を参照してください。
プリフィックスコード	プリフィックスコードがない場合、値は 0 となります。 Set_ReaderOutputConfiguration()のパラメータ szPrefixCode を参照してください。
デコードデータ	デコードされたバーコードデータ。
サフィックスコード	サフィックコードがない場合、値は 0 となります。 Set_ReaderOutputConfiguration()のパラメータ szSuffixCode を参照してください。
コードの長さ	デコードされたバーコードデータの長さ(プリフィックス／サフィックスコードを除く)。Set_ReaderOutputConfiguration()のパラメータ showCodeLen を参照してください。

[注]：シーケンスによるデータフィールドが含まれています。
[コード種別]、[プリフィックスコード]、[デコードデータ]、[サフィックスコード] 、[コードの長さ]

Get_ReaderOutputConfiguration

目的　現在のデータ出力フォーマットを取得します。

書式　ClResult Get_ReaderOutputConfiguration(ReaderOutputConfiguration settings)

引数　※印はデフォルト値です。

Enable_State enableKeyboardEmulation
[入/出力] 入力テキストとしてデータをエミュレートし、アクティブなアプリケーションに渡すかどうかを指定する値。

KeyboardEmulationType .None	キーボードエミュレーション無効。デコードされたデータはブロードキャストインテントメッセージによって送信されます。
KeyboardEmulationType .InputMethod(※)	入力メソッドによるキーボードエミュレーション有効。
KeyboardEmulationType .KeyEvent	キーイベントによるキーボードエミュレーション有効。

OutputEnterWay autoEnterWay
[入/出力] デコード後の文字の自動接辞。

OutputEnterWay.Disable	無効。
OutputEnterWay.SuffixData (※)	デコードデータの末尾に追加。(デコードデータ+入力文字)
OutputEnterWay.PreffixData	デコードデータの先頭に追加。(入力文字+デコードデータ)

OutputEnterChar autoEnterChar
[入/出力]自動接辞する文字。

OutputEnter.None	なし
OutputEnter.Return(※)	CR。(= 0x0d)
OutputEnter.Tab	タブ
OutputEnter.Comma	カンマ(=0x2c)
OutputEnter.Semicolon	セミコロン(=0x3b)

Enable_State showCodeType
[入/出力] データレコード内のバーコード種類の送信。

Enable_State.FALSE(※)	送信しない。
Enable_State.TRUE	送信する。

Enable_State showCodeLen
[入/出力] データレコード内のバーコードの長さの送信。

Enable_State.FALSE(※)	送信しない。
Enable_State.TRUE	送信する。

String szPrefixCode
[入/出力] プリフィックスコードが格納される String 変数。

String szSuffixCode
[入/出力] サフィックスコードが格納される String 変数。

int useDelim
[入/出力] 使用中の区切り文字を指定する ASCII 値。

0(※)	区切り文字なし。
1～127	UID とデータ間の区切り文字。

戻り値　成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

使用例　ReaderOutputConfiguration settings = new ReaderOutputConfiguration();
mReaderManager.Get_ReaderOutputConfiguration(settings);

備考　リーダの種類と関連するリーダの設定によっては、出力レコードのフィールドが異なる場合があります。
バーコードリーダからデータを取得したとき、データフィールドが含まれる場合があります。
[コード種別]、[プリフィックスコード]、[デコードデータ]、[サフィックスコード] 、[コードの長さ]

コード種別	showCodeType の値が TRUE の場合のみ出力されます。
プリフィックスコード	szPrefixCode の値が 0 でない場合のみ出力されます。
デコードデータ	enableKeyboardEmulation の値が InputMethod または KeyEvent の場合に出力されます。
サフィックスコード	szSufixCode の値が 0 でない場合のみ出力されます。
コードの長さ	showCodeLen の値が TRUE の場合のみ出力されます。 (プリフィックス／サフィックスコードは含まれません。)

参照　Set_ReaderOutputConfiguration

Set_ReaderOutputConfiguration

目的　データ出力フォーマットを設定します。

書式　ClResult Set_ReaderOutputConfiguration(ReaderOutputConfiguration settings)

引数　※印はデフォルト値です。

Enable_State enableKeyboardEmulation
[入/出力] 入力テキストとしてデータをエミュレートし、アクティブなアプリケーションに渡すかどかを指定する値。

KeyboardEmulationType .None	キーボードエミュレーション無効。デコードされたデータはブロードキャストインテントメッセージによって送信されます。
KeyboardEmulationType .InputMethod(※)	入力メソッドによるキーボードエミュレーション有効。
KeyboardEmulationType .KeyEvent	キーイベントによるキーボードエミュレーション有効。

OutputEnterWay autoEnterWay
[入/出力] デコード後の文字の自動接辞。

OutputEnterWay.Disable	無効。
OutputEnterWay.SuffixData (※)	デコードデータの末尾に追加。(デコードデータ+入力文字)
OutputEnterWay.PreffixData	デコードデータの先頭に追加。(入力文字+デコードデータ)

OutputEnterChar autoEnterChar
[入/出力]自動接辞する文字。

OutputEnter.None	なし
OutputEnter.Return(※)	CR。(= 0x0d)
OutputEnter.Tab	タブ
OutputEnter.Space	スペース(=0x20)
OutputEnter.Comma	カンマ(=0x2c)
OutputEnter.Semicolon	セミコロン(=0x3b)

Enable_State showCodeType
[入/出力] データレコード内のバーコード種類の送信。

Enable_State.FALSE(※)	送信しない。
Enable_State.TRUE	送信する。

Enable_State showCodeLen
[入/出力] データレコード内のバーコードの長さの送信。

Enable_State.FALSE(※)	送信しない。
Enable_State.TRUE	送信する。

String szPrefixCode
[入/出力] プリフィックスコードが格納されている String 変数。

String szSuffixCode
[入/出力] サフィックスコードが格納されている String 変数。

int useDelim
[入/出力] 使用中の区切り文字を指定する ASCII 値。

0(※)	区切り文字なし。
1～127	UID とデータ間の区切り文字。

戻り値　成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

使用例

```
ReaderOutputConfiguration settings = new ReaderOutputConfiguration();
settings.enableKeyboardEmulation = Enable_State.TRUE;
settings.autoEnterWay = OutputEnterWay.SuffixData;
settings.autoEnterChar = OutputEnterChar.Return;
settings.showCodeLen = Enable_State.TRUE;
settings.showCodeType = Enable_State.TRUE;
settings.szPrefixCode = "PreStr";
settings.szSuffixCode = "SufStr";
settings.useDelim = ':';
mReaderManager.Set_ReaderOutputConfiguration(settings);
```

備考

リーダの種類と関連するリーダの設定によっては、出力レコードのフィールドが異なる場合があります。
バーコードリーダからデータを取得したとき、データフィールドが含まれる場合があります。
[コード種別]、[プリフィックスコード]、[デコードデータ]、[サフィックスコード] 、[コードの長さ]

コード種別	showCodeType の値が TRUE の場合のみ出力されます。
プリフィックスコード	szPrefixCode の値が 0 でない場合のみ出力されます。
デコードデータ	enableKeyboardEmulation の値が InputMethod または KeyEvent の場合に出力されます。
サフィックスコード	szSufixCode の値が 0 でない場合のみ出力されます。
コードの長さ	showCodeLen の値が TRUE の場合のみ出力されます。 (プリフィックス／サフィックスコードは含まれません。)

参照 Get_ReaderOutputConfiguration

SoftScanTrigger

目的

物理的なトリガキーの動作をエミュレートします。
事前に以下の手順を行ってください。
1. android.content.ContextWrapper.registerReceiver 関数をコールし、CipherLab 特定の文字列"com.cipherlab.barcode.GeneralString.Intent_SOFTTRIGGER_DATA"を登録します。
2. Android BroadcastReceiver()関数をコールし、登録した文字列を取得します。
3. 受信インテントからデータを取り出します。

書式 void SoftScanTrigger();

使用例

```
public class MainActivity extends Activity {
   private IntentFilter filter;
   Button b1 = null;
   ReaderManager m_RM = null;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      b1 = (Button) findViewById(R.id.button1);
      b1.setOnClickListener(new OnClickListener() {

         @Override
         public void onClick(View v) {
            if (m_RM != null)
            {
               m_RM.SoftScanTrigger();
            }
         }
```

```
        });
        m_RM = ReaderManager.InitInstance(this);
        filter = new IntentFilter();
        filter.addAction(GeneralString.Intent_SOFTTRIGGER_DATA);
        registerReceiver(myDataReceiver, filter);
    }

    @Override
    protected void onDestroy() {
        // TODO Auto-generated method stub
        super.onDestroy();
        unregisterReceiver(myDataReceiver);
    }
    private final BroadcastReceiver myDataReceiver = new BroadcastReceiver() {

        @Override
        public void onReceive(Context context, Intent intent) {
            if (intent.getAction().equals(GeneralString.Intent_SOFTTRIGGER_DATA)) {
                // Fetch data from the intent
                String sDataStr = intent.getStringExtra(GeneralString.BcReaderData);
                Toast.makeText(MainActivity.this, "Decoded data is " + sDataStr,
                Toast.LENGTH_SHORT).show();
            }
        }
    };
}
```

1.3.2 リーダサービスバージョン

Get_BarcodeServiceVer

目的 リーダサービスのバージョンを取得します。

書式 String Get_BarcodeServiceVer();

使用例 String ver = mReaderManager.Get_BarcodeServiceVer();

1.4　ステータス表示操作

NotificationParams()での設定に応じて、デコード成功時に音をや鳴らしたり、バイブを振動させたりします。

1.4.1　通知設定

Get_NotificationParams

目的　通知設定を取得します。

書式　ClResult Get_NotificationParams (NotificationParams settings);

引数　※印はデフォルト値です。

BeepType ReaderBeep
[入/出力]再生する音を指定する値。

BeepType.Mute
BeepType.Default
BeepType.Hwandsw
BeepType.MenuPop
BeepType.MsgBox
BeepType.Notify
BeepType.VoiceBeep
BeepType.Alarm2
BeepType.Alarm3
BeepType.LowBatt

Enable_State enableVibrator
[入/出力] 読取り成功時のバイブ有無。

Enable_State.FALSE(※)	バイブなし
Enable_State.TRUE	バイブあり

int vibrationCounter
[入/出力]バイブの振動時間。

0	バイブなし
1～10	デフォルトは 1 (0.5 秒刻み)

int ledDuration
[入/出力] 読取り成功時の LED 点灯有無。

0(※)	点灯なし
1～5000	点灯時間(ミリ秒)

使用例　NotificationParams settings = new NotificationParams();
mReaderManager.Get_NotificationParams(settings);

戻り値　成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

参照　Set_NotificationParams

Set_NotificationParams

目的　通知設定を構成します。

書式　ClResult Set_NotificationParams (NotificationParams settings);

引数　※印はデフォルト値です。

BeepType ReaderBeep
[入/出力]再生する音を指定する値。

BeepType.Mute
BeepType.Default
BeepType.Hwandsw
BeepType.MenuPop
BeepType.MsgBox
BeepType.Notify
BeepType.VoiceBeep
BeepType.Alarm2
BeepType.Alarm3
BeepType.LowBatt

Enable_State enableVibrator
[入/出力] 読取り成功時のバイブ有無。

Enable_State.FALSE(※)	バイブなし
Enable_State.TRUE	バイブあり

int vibrationCounter
[入/出力]バイブの振動時間。

0	バイブなし
1～10	デフォルトは 1 (0.5 秒刻み)

int ledDuration
[入/出力] 読取り成功時の LED 点灯有無。

0(※)	点灯なし
1～5000	点灯時間(ミリ秒)

使用例

```
NotificationParams settings = new NotificationParams();
settings.enableReaderBeep = Enable_State.TRUE;
settings.enableVibrator = Enable_State.TRUE;
settings.ledDuration = 500; //ms
settings.vibrationCounter = 1; //500ms * count
mReaderManager.Set_NotificationParams(settings);
mReaderManager.Get_NotificationParams(settings);
```

戻り値　成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

参照　Get_NotificationParams

1.5　スキャンエンジン設定

1.5.1 プリファレンス

Get_Decoders_Status

目的　バーコードごとのリーダの読取り(有効/無効)を取得します。

書式　ClResult Get_Decoders_Status(Readers settings)

引数　※印はデフォルト値です。
[入/出力]各値はリーダが対応するバーコードをデコードできるかどうかを指定します。

Enable_State enableAustrailianPostal

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableAztec

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableCompositeCC_AB

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

Enable_State enableCompositeCC_C

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableCompositeTlc39

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

Enable_State enableCode11

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableCode39

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableCode93

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableCode128

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableCodabar

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableChinese2Of5

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableDataMatrix

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableDutchPostal

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableEanJan8

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableEanJan13

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableGs1128

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableGs1DataBar14

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableGs1DataBarLimited

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableGs1DataBarExpanded

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableGs1DatabarToUpcEan

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableIsbt128

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableIndustrial2Of5

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableInterleaved2Of5

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableJapanPostal

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableKorean3Of5

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

Enable_State enableMatrix2Of5

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableMaxiCode

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableMicroPDF417

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

Enable_State enableMicroQR

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableMsi

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enablePDF417

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableQRcode

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableTriopticCode39

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

Enable_State enableUpcA

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableUpcE

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableUpcE1

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

Enable_State enableUccCoupon

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

Enable_State enableUKPostal

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableUPUFICSPostal

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

Enable_State enableUSPostnet

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableUSPlanet

Enable_State.FALSE	読取り無効
Enable_State.TRUE(※)	読取り有効

Enable_State enableUSPSPostal

Enable_State.FALSE(※)	読取り無効
Enable_State.TRUE	読取り有効

戻り値　成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

使用例

```
Decoders settings = new Decoders();
mReaderManager.Get_Decoders_Status(settings);
if (Enable_State.NotSupport == settings.enableAustrailianPostal) {
   // 1D is not supported
}
```

備考　Enable_State.FALSE はリーダがそのバーコードをデコードしないことを意味します。
Enable_State.TRUE はリーダがそのバーコードをデコードすることを意味します。

参照　Set_Decoders_Status

Set_Decoders_Status

目的　バーコードごとのリーダの読取り(有効/無効)を設定します。

書式　ClResult Set_Decoders_Status(Readers settings)

引数　42 のバーコードに対応しています。詳細は Get_Decoders_Status を参照してください。

戻り値　成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

使用例

```
Decoders settings = new Decoders();
settings.enableAustrailianPostal = Enable_State.FALSE;
settings.enableAztec = Enable_State.FALSE;
settings.enableChinese2Of5 = Enable_State.FALSE;
settings.enableCodabar = Enable_State.FALSE;
settings.enableCode11 = Enable_State.FALSE;
settings.enableCode128 = Enable_State.FALSE;
settings.enableCode39 = Enable_State.FALSE;
settings.enableCode93 = Enable_State.FALSE;

if (ClResult.S_ERR == mReaderManager.Set_Decoders_Status(settings))
   Toast.makeText(this, "Set_Decoders_Status was
   failed",Toast.LENGTH_SHORT).show();
else
   Toast.makeText(this, "Set_Decoders_Status was
   successful ",Toast.LENGTH_SHORT).show();
```

備考　Enable_State.FALSE はリーダがそのバーコードをデコードしないことを意味します。
Enable_State.TRUE はリーダがそのバーコードをデコードすることを意味します。

参照　Get_Decoders_Status

Get_UserPreferences

目的　バーコードリーダのプリファレンス設定を取得します。

書式　ClResult Get_UserPreferences(UserPreference settings)

引数　※印はデフォルト値です。

int addonSecurityLevel
[入/出力] アドオンのデコード(自動識別)有効の場合、UPC/EAN 読み取りのデコードのセキュリティレベル。

2～30	デフォルトは 10(読取り回数)

Enable_State displayMode
[入/出力] 表示有効/無効。

Enable_State.FALSE(※)	無効
Enable_State.TRUE	有効

int laserOnTime
[入/出力] スキャン時のバーコードのデコード最大時間。

500～9900	デフォルトは 3000(ミリ秒)

InverseType negativeBarcodes

[入/出力] 白黒反転バーコード読取りを指定する値。

InverseType.RegularOnly(※)	通常バーコードのみ。
InverseType.InverseOnly	反転バーコードのみ。
InverseType.Autodetect	自動判別。

Enable_State pickListMode

[入/出力] 正確なデコードのためのピックリストモード有効/無効。

Enable_State.FALSE(※)	無効
Enable_State.TRUE	有効

RedundancyLevel redundancyLevel

[入/出力] デコードの冗長性。バーコードの品質が悪い場合は高い冗長性を選択してください。

RedundancyLevel.One (※)	以下のバーコードを、正常なデコードのために 2 度読取り照合を行います。 バーコード種別 / コードの長さ Codabar / すべて MSI / 4 文字以下 Industrial 25 (Discrete 25) / 8 文字以下 Interleaved 25 / 8 文字以下
RedundancyLevel.Two	すべてのバーコードで正常なデコードのために 2 度読取り照合を行います。
RedundancyLevel.Three	すべてのバーコードで正常なデコードのために 2 度読取り照合を行います。ただし、以下のバーコードでは正常なデコードのために 3 度読取り照合を行います。 バーコード種別 / コードの長さ MSI / 4 文字以下 Industrial 25 (Discrete 25) / 8 文字以下 Interleaved 25 / 8 文字以下
RedundancyLevel.Four	すべてのバーコードで正常なデコードのために 3 度読取り照合を行います。

ScanAngleType scanAngle

[入/出力] 走査角度。

ScanAngleType.Narrow	狭角(35 度)
ScanAngleType.Wide (※)	広角(47 度)

SecurityLevel securityLevel

[入/出力] Code 128、Code 93、UPC/EAN のようなデルタバーコードを読取るときに、いくつかの印刷品質の問題を修正するデコードのセキュリティレベル。

SecurityLevel.Zero (※)	レベル 0 – デフォルト。スキャンエンジンの性能を最大限にいかすことで、ほとんどの仕様の範囲内のバーコードをデコードできます。
SecurityLevel.One	レベル 1 - デコードミスが発生する場合はこのオプションを選択します。
SecurityLevel.Two	レベル 2 – レベル 1 でもデコードミスが発生する場合はこのオプションを選択します。
SecurityLevel.Three	レベル 3 – レベル 2 でもデコードミスが発生する場合はこのオプションを選択します。ただし、このオプションを選択するとデコード性能が劣ります。バーコードの品質を向上させることをお勧めします。

int timeoutBetweenSameSymbology

[入/出力] 2 度続けて同じバーコードを読むことができるまでの最小時間。誤って同じバーコードを 2 度続けて読むことを阻止するための設定です。コンティニュアスモード時に適用されます。

0～9900	デフォルトは 1000(ミリ秒)

TransmitCodeIDType transmitCodeIdChar
[入/出力] コード ID 文字送信有無。

TransmitCodeIDType.None(※)	送信無し。
TransmitCodeIDType.AimCodeID	AIM コード ID 文字送信。

TriggerType triggerMode
[入/出力] スキャンモード。

TriggerType.LevelMode(※)	レベルモード
TriggerType.ContinuousMode	連続モード
TriggerType.AutoAimMode	自動照準モード

Enable_State decodingillumniation
[入/出力]読取りやすくするためにバーコード取り込み時にフラッシュ証明を点灯させるかどうか。

Enable_State.FALSE	照明なし。
Enable_State.TRUE(※)	照明あり。

Enable_State decodingAimingPattern
[入/出力] すべてのバーコード読取りでの照準パターン有無。

Enable_State.FALSE	デコード照準パターンなし。
Enable_State.TRUE(※)	デコード照準パターンあり。

InterCharacterGapSize interCharGapSize
[入/出力] 一般的には非常に小さい、Code 39 と Codabar ための文字間のギャップサイズを指定する値。各種のバーコード印刷技術により、ギャップサイズが規定よりも大きいものも作成され、スキャナがデコードできない場合があります。このような場合、文字間ギャップ大を設定して仕様外のバーコードを読めるようにします。

0x06(※)	通常の文字間隔。
0x0A	大きい文字間隔。

戻り値 成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

使用例

```
UserPreference settings = new UserPreference();
mReaderManager.Get_UserPreferences(settings);
if (Enable_State.NotSupport == settings.displayMode)
{


}
```

参照 Set_UserPreferences

Set_UserPreferences

目的 バーコードリーダのプリファレンスを設定します。

書式 ClResult Set_UserPreferences(UserPreference settings)

引数 14 のプリファレンス設定は次の通りです。詳細は Get_UserPreferences を参照してください。

int addonSecurityLevel
Enable_State displayMode
int laserOnTime
InverseType negativeBarcodes
Enable_State pickListMode
RedundancyLevel redundancyLevel
ScanAngleType scanAngle
SecurityLevel securityLevel
int timeoutBetweenSameSymbology
TransmitCodeIDType transmitCodeIdChar
TriggerType triggerMode

Enable_State decodingillumniation
Enable_State decodingAimingPattern
InterCharacterGapSize interCharGapSize

戻り値 成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

使用例

```
UserPreference settings = new UserPreference();
settings.addonSecurityLevel = 2;
settings.laserOnTime = 3000;
settings.negativeBarcodes = InverseType.AutoDetect;
settings.scanAngle = ScanAngleType.Wide;

if (ClResult.S_ERR == mReaderManager.Set_UserPreferences(settings))
   Toast.makeText(this, " Set_UserPreferences was failed",
   Toast.LENGTH_SHORT).show();
else
   Toast.makeText(this, " Set_UserPreferences was successful ",
   Toast.LENGTH_SHORT).show();
```

参照 Get_UserPreferences

1.5.2 読取り設定

Get_Symbology

目的 バーコードシンボルのインターフェイスインスタンスを実装することで、バーコードシンボルの設定を取得します。

書式 ClResult Get_Symbology(SymbologyInterface settings)

引数 この関数にはオブジェクトとして実装 38 バーコードシンボルのパラメータが用意されています。

```
Codabar
{
   Enable_State transmitCheckDigit;
   CodabarDigitAlgorithm verifyCheckDigit;
   NOTISEditingType notisEditingType;
   Enable_State enable;
   int length1;
   int length2;
   Enable_State clsiEditing;
   Enable_State notisEditing;
}

Code11
{
   Enable_State enable;
   int length1;
   int length2;
   NumberOfCheck numberOfCheckDigits;
   Enable_State transmitCheckDigit;
}

Code39
{
   Enable_State enable;
```

```
    int length1;
    int length2;
    Enable_State checkDigitVerification;
    Enable_State transmitCheckDigit;
    Enable_State fullASCII;
    Enable_State convertToCode32;
    Enable_State convertToCode32Prefix;
}

TriopticCode39
{
    Enable_State enable
}

Korean3Of5
{
    Enable_State enable
}

Code93
{
    Enable_State enable;
    int length1;
    int length2;
}

Code128
{
    Enable_State enable
}

GS1128
{
    Enable_State enable;
    char fieldSeparator;        //ranging from 0 to 127
}

ISBT128
{
    Enable_State enable;
    ISBTConcatenationType concatenation;
    int concatenationRedundancy;        //ranging from 2 to 20
}

Chinese2Of5
{
    Enable_State enable;
}

Industrial2Of5
{
    Enable_State enable;
    int length1;
    int length2;
}
```

```
Interleaved2Of5
{
   Enable_State enable;
   int length1;
   int length2;
   Enable_State checkDigitVerification;
   Enable_State transmitCheckDigit;
   Enable_State convertToEan13;
}

Matrix2Of5
{
   Enable_State enable;
   int length1;
   int length2;
   int redundancy;
   Enable_State checkDigitVerification;
   Enable_State transmitCheckDigit;
}

UccCoupon
{
   Enable_State enable;
}

GS1DataBar14
{
   Enable_State enable;
   Enable_State convertToUpcEan;
}

GS1DataBarLimited
{
   int securityLevel;
   Enable_State enable;
   Enable_State convertToUpcEan;
}
```

[注]：GS1DataBar14 と GS1dataBarLimited にある convertToUpcEan はどちらかの値を変更するともう片方の値も変更します。

```
GS1DataBarExpanded
{
   Enable_State enable;
   char fieldSeparator;       //ranging from 0 to 127
}

Msi
{
   Enable_State enable;
   int length1;
   int length2;
   MsiDigitOption checkDigitOption;
   Enable_State transmitCheckDigit;
   DigitAlgorithm checkDigitAlgorithm;
```

```
}

Ean8
{
   Enable_State enable;
   AddonsType addon2;
   AddonsType addon5;
   Enable_State transmitCheckDigit;
   Enable_State convertToEan13;
}

Ean13
{
   Enable_State enable;
   AddonsType addon2;
   AddonsType addon5;
   Enable_State convertToISBN;
   Enable_State convertToISSN;
   ISBNFormat booklandISBNFormat;
   Enable_State transmitCheckDigit;
}
```

[注]：EAN8 と EAN13 にある addon2、addon5 と transmitCheckDigit はどちらかの値を変更するともう片方の値も変更します。

```
UpcA
{
   Enable_State enable;
   AddonsType addon2;
   AddonsType addon5;
   Enable_State transmitCheckDigit;
   Preamble transmitSystemNumber;
   Enable_State convertToEan13
}

UpcE
{
   Enable_State enable;
   AddonsType addon2;
   AddonsType addon5;
   Enable_State transmitCheckDigit;
   Preamble transmitSystemNumber;
   Enable_State convertToUpcA;
}

UpcE1
{
   Enable_State enable;
   AddonsType addon2;
   AddonsType addon5;
   Enable_State transmitCheckDigit;
   Preamble transmitSystemNumber;
   Enable_State convertToUpcA;
}
```

[注]：UpcA、UpcE と UpcE1 にある addon2 と addon5 はどちらかの値を変更すると残り２つの値も変更します。

```
Composite
{
   Enable_State enableCc_C;
   Enable_State enableCc_AB;
   Enable_State enableTlc39;
   UpcMode enableUpcMode;
   Enable_State enableEmulationMode;
}

USPostal
{
   Enable_State enablePlanet;
   Enable_State enablePostnet;
   Enable_State transmitCheckDigit;
}

UKPostal
{
   Enable_State enable;
   Enable_State transmitCheckDigit;
}

JapanPostal
{
   Enable_State enable;
}

AustralianPostal
{
   Enable_State enable;
}

DutchPostal
{
   Enable_State enable;
}

USPSPostal
{
   Enable_State enable;
}

UPUFICSPostal
{
   Enable_State enable;
}

PDF417
{
   Enable_State enable;
   TransmitMode transmitMode;
   char escapeCharacter;
   Enable_State transmitControlHeader;
}
```

```
MicroPDF417
{
   Enable_State enable;
   Enable_State code128Emulation;
}

DataMatrix
{
   Enable_State enable;
   char fieldSeparator;       //ranging from 0 to 127
   MatrixMirrorImage mirrorImage;
}

MaxiCode
{
   Enable_State enable;
}

QRCode
{
   Enable_State enable;
}

MicroQR
{
   Enable_State enable;
}

Aztec
{
   Enable_State enable;
}
```

戻り値　成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

使用例

```
Codabar settings = new Codabar();
if (ClResult.Err_NotSupport == mReaderManager.Get_Symbology(settings))
{
   // to verify whether the symbology is supported
}

// if disabled, enable it and then configure it via Set_Symbology
if (Codabar.enable == Enable_State.FALSE)
{
   Codabar.enable = Enable_State.TRUE;
}
```

参照　Set_Symbology

Set_Symbology

目的	バーコードシンボルのインターフェイスインスタンスを実装することで、バーコードシンボル設定を設定します。
書式	ClResult Set_Symbology(SymbologyInterface settings)
引数	この関数にはオブジェクトとして 38 バーコードシンボルのパラメータが用意されています。詳細は Get_Symbology を参照してください。
戻り値	成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

使用例	Codabar settings = new Codabar();
	Codabar.enable = Enable_State.TRUE;
	if (ClResult.S_ERR == mReaderManager.Set_Symbology(settings))
	Toast.makeText(this, "Set_Symbology was failed", Toast.LENGTH_SHORT).show();
	else
	Toast.makeText(this, "Set_Symbology was successful",
	Toast.LENGTH_SHORT).show();

参照	Get_Symbology

1.5.3 CODABAR クラス

```
public  Codabar
{
    public  Enable_State transmitCheckDigit;
    public  CodabarDigitAlgorithm verifyCheckDigit;
    public  NOTISEditingType notisEditingType;
    public  Enable_State enable;
    public  int length1;
    public  int length2;
    public  Enable_State clsiEditing;
    public  Enable_State notisEditing;
}
```

データ型	メンバ名	説明
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE
CodabarDigitAlgorithm	verifyCheckDigit	チェックディジット検査方法。 NONE(※) Modulo_16 Modulo_7DR Modulo_Both [注]: Modulo_7DR は CODABAR の総桁数が 19 以下でなければなりません。また CODABAR の最初の桁は 8 以下でなければなりません。
NOTISEditingType	notisEditingType	NOTIS 編集フォーマット送信と送信方法。 NONE(※) ABCD abcd
Enable_State	enable	Codabar 有効/無効。 TRUE(※) FALSE
Int	length1	バーコードの長さ。 デフォルトは 4。(0～55)
Int	length2	バーコードの長さ。 デフォルトは 55。(0～55)
Enable_State	clsiEditing	CLSI 編集有無。 TRUE FALSE(※)
Enable_State	notisEditing	NOTIS 編集有無。 TRUE FALSE(※)

1.5.4 CODE11 クラス

```
public  Code11
{
    public  Enable_State enable;
    public  int length1;
    public  int length2;
    public  NumberOfCheck numberOfCheckDigits;
    public  Enable_State transmitCheckDigit;
}
```

データ型	メンバ名	説明
Enable_State	enable	Code 11 有効/無効。 TRUE(※) FALSE
int	length1	バーコードの長さ。 デフォルトは 4。(0～55)
int	length2	バーコードの長さ。 デフォルトは 55。(0～55)
NumberOfCheck	numberOfCheckDigits	チェックディジット検査。 NONE(※) ONE TWO
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE FALSE(※)

1.5.5 CODE39 クラス

```
public  Code39
{
    public  Enable_State enable;
    public  Enable_State checkDigitVerification;
    public  Enable_State transmitCheckDigit;
    public  Enable_State fullASCII;
    public  Enable_State convertToCode32;
    public  Enable_State convertToCode32Prefix;
    public  int length1;
    public  int length2;
}
```

データ型	メンバ名	説明
Enable_State	enable	Code39 有効/無効。 TRUE(※) FALSE
Enable_State	checkDigitVerification	チェックディジット検査有無。 TRUE FALSE(※)
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE FALSE(※)
Enable_State	fullASCII	Code39 フルアスキーのサポート有無。 TRUE FALSE(※)
Enable_State	convertToCode32	Code39 から Code32(Italian Pharmacode)への変換有無。 TRUE FALSE(※)
Enable_State	convertToCode32Prefix	Code32 プリフィックスの送信有無。 TRUE FALSE(※)
int	length1	バーコードの長さ。 デフォルトは 4。(0～55)
int	length2	バーコードの長さ。 デフォルトは 55。(0～55)

1.5.6 TRIOPTICCODE39 クラス

```
public  TriopticCode39
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	TriopticCode39 有効/無効。 TRUE(※) FALSE

1.5.7 KOREAN3OF5 クラス

```
public  Korean3Of5
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	Korean3of5 有効/無効。 TRUE(※) FALSE

1.5.8 CODE93 クラス

```
public  Code93
{
    public  Enable_State enable;
    public  int length1;
    public  int length2;
}
```

データ型	メンバ名	説明
Enable_State	enable	Code 93 有効/無効。 TRUE(※) FALSE
Int	length1	バーコードの長さ。 デフォルトは 4。(0～55)
Int	length2	バーコードの長さ。 デフォルトは 55。(0～55)

1.5.9 CODE128 クラス

```
public  Code128
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	Code128 有効/無効。 TRUE(※) FALSE

1.5.10 GS1128 クラス

```
public  GS1128
{
    public  Enable_State enable;
    public  char fieldSeparator;
}
```

データ型	メンバ名	説明
Enable_State	enable	Code128 有効/無効。 TRUE(※) FALSE
char	fieldSeparator	フィールドセパレータの適用有無。0～127 で指定します。デフォルトは 0。

1.5.11 ISBT128 クラス

```
public  ISBT128
{
    public  Enable_State enable;
    public  ISBTConcatenationType concatenation;
    public  int concatenationRedundancy;
}
```

データ型	メンバ名	説明
Enable_State	enable	ISBT128 有効/無効。 TRUE(※) FALSE
ISBTConcatenationType	concatenation	デコードした ISBT バーコードの連結。 Disable Enable Auto(※)
int	concatenationRedundancy	ISBT 連結が自動になっている場合の連結の冗長性(2～20 回)。デフォルトは 10。

1.5.12 CHINESE2OF5 クラス

```
public  Chinese2Of5
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	Chinese25 有効/無効。 TRUE(※) FALSE

1.5.13 INDUSTRIAL2OF5 クラス

```
public  Industrial2Of5
{
    public  Enable_State enable;
    public  int length1;
    public  int length2;
}
```

データ型	メンバ名	説明
Enable_State	enable	Industrial25 有効/無効。 TRUE(※) FALSE
int	length1	バーコードの長さ。 デフォルトは 4。(0～55)
int	length2	バーコードの長さ。 デフォルトは 55。(0～55)

1.5.14 INTERLEAVED2OF5 クラス

```
public  Interleaved2Of5
{
    public  Enable_State enable;
    public  int length1;
    public  int length2;
    public  I20f5CheckDigitVerification checkDigitVerification;
    public  Enable_State transmitCheckDigit;
    public  Enable_State convertToEan13;
}
```

データ型	メンバ名	説明
Enable_State	enable	Interleaved25 有効/無効。 TRUE(※) FALSE
int	length1	バーコードの長さ。 デフォルトは 4。(0～55)
int	length2	バーコードの長さ。 デフォルトは 55。(0～55)
I20f5CheckDigitVerification	checkDigitVerification	チェックディジット検査。 Disable(※) USS OPCC
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE FALSE(※)
Enable_State	convertToCode13	Interleaved25 から EAN-13 への変換有無。 TRUE FALSE(※)

1.5.15 MATRIX2OF5 クラス

```
public  class Matrix 25
{
    public  Enable_State enable;
    public  int length1;
    public  int length2;
    public  Enable_State redundancy;
    public  Enable_State checkDigitVerification;
    public  Enable_State transmitCheckDigit;
}
```

データ型	メンバ名	説明
Enable_State	enable	Interleaved25 有効/無効。 TRUE(※) FALSE
int	length1	バーコードの長さ。 デフォルトは 4。(0～55)
int	length2	バーコードの長さ。 デフォルトは 55。(0～55)
Enable_State	redundancy	デコードの冗長性有無。 TRUE FALSE(※)
Enable_State	checkDigitVerification	チェックディジット検査有無。 TRUE FALSE(※)
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE FALSE(※)

1.5.16 UCCCOUPON クラス

```
public  UccCoupon
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	Ucc Coupon 有効/無効。 TRUE FALSE(※)

1.5.17 GS1DATABAR14 クラス

```
public  GS1DataBar14
{
    public  Enable_State enable;
    public  Enable_State convertToUpcEan;
}
```

データ型	メンバ名	説明
Enable_State	enable	GS1 DataBar-14 有効/無効。 TRUE(※) FALSE
Enable_State	convertToUpcEan	RSS から UPC/EAN への変換有無。 TRUE FALSE(※)

1.5.18 GS1DATABARLIMITED クラス

```
public  GS1DataBarLimited
{
    int  securityLevel;
    public  Enable_State enable;
    public  Enable_State convertToUpcEan;
}
```

データ型	メンバ名	説明
int	securityLevel	GS1 DataBar Limited 読取り時のセキュリティレベル。 デフォルトは 3。(1～4)
Enable_State	enable	GS1 DataBar Limited 有効/無効。 TRUE(※) FALSE
Enable_State	convertToUpcEan	RSS から UPC/EAN への変換有無。 TRUE FALSE(※)

1.5.19 GS1DATABAREXPANDED クラス

```
public  GS1DataBarExpanded
{
    public  Enable_State enable;
    public  char fieldSeparator;
}
```

データ型	メンバ名	説明
Enable_State	enable	GS1 DataBar Expanded 有効/無効。 TRUE(※) FALSE
char	fieldSeparator	フィールドセパレータの適用有無。0～127 で指定します。デフォルトは 0。

1.5.20 MSI クラス

```
public  class Msi
{
    public  Enable_State enable;
    public  int length1;
    public  int length2;
    public  MsiDigitOption checkDigitOption;
    public  Enable_State transmitCheckDigit;
    public  DigitAlgorithm checkDigitAlgorithm;
}
```

データ型	メンバ名	説明
Enable_State	enable	MSI 有効/無効。 TRUE(※) FALSE
int	length1	バーコードの長さ。 デフォルトは 4。(0～55)
int	length2	バーコードの長さ。 デフォルトは 55。(0～55)
MsiDigitOption	checkDigitOption	チェックディジット確認方法。 Onedigit(※) Twodigits
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE
Enable_State	checkDigitAlgorithm	適用するアルゴリズム。 Modulo_10_11(※) DoubleModulo_10

1.5.21 EAN8 クラス

```
public  class Ean8
{
    public  Enable_State enable;
    public  AddonsType addon2;
    public  AddonsType addon5;
    public  Enable_State transmitCheckDigit;
    public  Enable_State convertToEan13;
}
```

データ型	メンバ名	説明
Enable_State	enable	EAN-8 有効/無効。 TRUE(※) FALSE
AddonsType	addon2	アドオン 2 処理方法。 IgnoresAddon (※) AutoDiscriminate
AddonsType	addon5	アドオン 5 処理方法。 IgnoresAddon (※) AutoDiscriminate
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE
Enable_State	convertToCode13	EAN-8 から EAN-13 への変換有無。 TRUE FALSE(※)

1.5.22 EAN13 クラス

```
public  class Ean13
{
    public  Enable_State enable;
    public  AddonsType addon2;
    public  AddonsType addon5;
    public  Enable_State convertToISBN;
    public  Enable_State convertToISSN;
    public  ISBNFormat booklandISBNFormat;
    public  Enable_State transmitCheckDigit;
}
```

データ型	メンバ名	説明
Enable_State	enable	EAN-13 有効/無効。 TRUE(※) FALSE
AddonsType	addon2	アドオン 2 処理方法。 IgnoresAddon (※) AutoDiscriminate
AddonsType	addon5	アドオン 5 処理方法。 IgnoresAddon (※) AutoDiscriminate
Enable_State	convertToISBN	EAN-13 から ISBN への変換有無。 TRUE FALSE(※)
Enable_State	convertToISSN	EAN-13 から ISSN への変換有無。 TRUE FALSE(※)
ISBNFormat	booklandISBNFormat	Bookland EAN 有効時の Bookland データのフォーマット。 ISBN_10 ISBN_13
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE

1.5.23 UPCA クラス

```
public  class Upca
{
    public  Enable_State enable;
    public  AddonsType addon2;
    public  AddonsType addon5;
    public  Enable_State transmitCheckDigit;
    public  Preamble transmitSystemNumber;
    public  Enable_State convertToEan13;
}
```

データ型	メンバ名	説明
Enable_State	enable	UPC-A 有効/無効。 TRUE(※) FALSE
AddonsType	addon2	アドオン 2 処理方法。 IgnoresAddon (※) AutoDiscriminate
AddonsType	addon5	アドオン 5 処理方法。 IgnoresAddon (※) AutoDiscriminate
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE
Preamble	transmitSystemNumber	(送信)UPC-A プリアンブル検証。 None SysNumOnly(※) SysNumAndCtyCode
Enable_State	convertToEan13	EAN-13 への変換有無。 TRUE FALSE(※)

1.5.24 UPCE クラス

```
public  class Upce
{
    public  Enable_State enable;
    public  AddonsType addon2;
    public  AddonsType addon5;
    public  Enable_State transmitCheckDigit;
    public  Preamble transmitSystemNumber;
    public  Enable_State convertToUpcA;
}
```

データ型	メンバ名	説明
Enable_State	enable	UPC-E 有効/無効。 TRUE(※) FALSE
AddonsType	addon2	アドオン 2 処理方法。 IgnoresAddon (※) AutoDiscriminate
AddonsType	addon5	アドオン 5 処理方法。 IgnoresAddon (※) AutoDiscriminate
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE
Preamble	transmitSystemNumber	(送信)UPC-E プリアンブル検証。 None SysNumOnly(※) SysNumAndCtyCode
Enable_State	convertToUpcA	UPC-A への変換有無。 TRUE FALSE(※)

1.5.25 UPCE1 クラス

```
public  class Upce1
{
    public  Enable_State enable;
    public  AddonsType addon2;
    public  AddonsType addon5;
    public  Enable_State transmitCheckDigit;
    public  Preamble transmitSystemNumber;
    public  Enable_State convertToUpcA;
}
```

データ型	メンバ名	説明
Enable_State	enable	UPC-E1 有効/無効。 TRUE(※) FALSE
AddonsType	addon2	アドオン 2 処理方法。 IgnoresAddon (※) AutoDiscriminate
AddonsType	addon5	アドオン 5 処理方法。 IgnoresAddon (※) AutoDiscriminate
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE
Preamble	transmitSystemNumber	(送信)UPC-E1 プリアンブル検証。 None SysNumOnly(※) SysNumAndCtyCode
Enable_State	convertToUpcA	UPC-A への変換有無。 TRUE FALSE(※)

1.5.26 COMPOSITE クラス

```
public  class Composite
{
    public  Enable_State enableCc_C;
    public  Enable_State enableCc_AB;
    public  Enable_State enableTlc39;
    public  UpcMode enableUpcMode;
    public  Enable_State enableEmulationMode;
}
```

データ型	メンバ名	説明
Enable_State	enableCc_C	Compsite CC-C 有効/無効。 TRUE(※) FALSE
Enable_State	enableCc_AB	Compsite CC-A/B 有効/無効。 TRUE FALSE(※)
Enable_State	enableTlc39	Compsite TLC-39(TCIF Linked)有効/無効。 TRUE FALSE(※)
UpcMode	enableUpcMode	(送信)UPC 結合有無。 NeverLinksUPC AlwaysLinksUPC(※) Auto
Enable_State	enableEmulationMode	UCC/EAN Composite Code の場合の GS-1 エミュレーションモード有効/無効。 TRUE FALSE(※)

1.5.27 USPOSTAL クラス

```
public  class USPostal
{
    public  Enable_State enablePlanet;
    public  Enable_State enablePostnet;
    public  Enable_State transmitCheckDigit;
}
```

データ型	メンバ名	説明
Enable_State	enablePlanet	US Planet 有効/無効。 TRUE(※) FALSE
Enable_State	enablePostnet	US Postnet 有効/無効。 TRUE(※) FALSE
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE

1.5.28 UKPOSTAL クラス

```
public  class UKPostal
{
    public  Enable_State enable;
    public  Enable_State transmitCheckDigit;
}
```

データ型	メンバ名	説明
Enable_State	enable	UK Postal 有効/無効。 TRUE(※) FALSE
Enable_State	transmitCheckDigit	チェックディジット送信有無。 TRUE(※) FALSE

1.5.29 JAPANPOSTAL クラス

```
public  class JapanPostal
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	Japan Postal 有効/無効。 TRUE(※) FALSE

1.5.30 AUSTRALIANPOSTAL クラス

```
public  class AustralianPostal
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	Australian Postal 有効/無効。 TRUE(※) FALSE

1.5.31 DUTCHPOSTAL クラス

```
public  class DutchPostal
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	Dutch Postal 有効/無効。 TRUE(※) FALSE

1.5.32 USPSPOSTAL クラス

```
public  class USPSPostal
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	USPS Postal 有効/無効。 TRUE(※) FALSE

1.5.33 UPUFICSPOSTAL クラス

```
public  class UPUFICSPostal
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	UPUFICS Postal 有効/無効。 TRUE(※) FALSE

1.5.34 PDF417 クラス

```
public  class PDF417
{
    public  Enable_State enable;
    public  TransmitMode transmitMode;
    public  Enable_State escapeCharacter;
    public  Enable_State transmitControlHeader;
}
```

データ型	メンバ名	説明
Enable_State	enable	PDF417 有効/無効。 TRUE(※) FALSE
TransmitMode	transmitMode	デコード処理方法。 BufferAllSymbols TransmitAnySymbolInSet PassthroughAllSymbols(※)
Enable_State	escapeCharacter	スペース文字使用。 TRUE FALSE(※)
Enable_State	transmitControlHeader	制御ヘッダ送信有無。 TRUE FALSE(※)

1.5.35 MICROPDF417 クラス

```
public  class MicroPDF417
{
    public  Enable_State enable;
    public  Enable_State code128Emulation;
}
```

データ型	メンバ名	説明
Enable_State	enable	MicroPDF417 有効/無効。 TRUE(※) FALSE
Enable_State	code128Emulation	特定の MicroPDF417 用 Code 128 エミュレーション有効/無効。 TRUE FALSE(※)

1.5.36 DATAMATRIX クラス

```
public  class DataMatrix
{
    public  Enable_State enable;
    public  char fieldSeparator;
    public  MatrixMirrorImage mirrorImage;
}
```

データ型	メンバ名	説明
Enable_State	enable	Data Matrix 有効/無効。 TRUE(※) FALSE
char	fieldSeparator	フィールドセパレータの適用有無。0～127 で指定します。デフォルトは 0。
MatrixMirrorImage	mirrorImage	DataMatrix ミラー(左右反転)バーコード有効/無効。 Never(※) Always Auto

1.5.37 MAXICODE クラス

```
public  class MaxiCode
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	MaxiCode 有効/無効。 TRUE(※) FALSE

1.5.38 QRODE クラス

```
public  class QRCode
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	QR Code 有効/無効。 TRUE(※) FALSE

1.5.39 MICROQR クラス

```
public  class MicroQR
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	MicroQR 有効/無効。 TRUE(※) FALSE

1.5.40 AZTEC クラス

```
public  class Aztec
{
    public  Enable_State enable;
}
```

データ型	メンバ名	説明
Enable_State	enable	Aztec 有効/無効。 TRUE(※) FALSE

1.6 リーダのリセット

ResetReaderToDefault

目的 リーダのモジュールをリセットします。

書式 ClResult ResetReaderToDefault ()

使用例

```
if (ClResult.S_ERR == mReaderManager.ResetReaderToDefault())
{
   Toast.makeText(this, "ResetReaderToDefault was failed",
   Toast.LENGTH_SHORT).show();
}
else
{
   Toast.makeText(this, "ResetReaderToDefault was done!",
   Toast.LENGTH_SHORT).show();
}
```

戻り値 成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

備考 リーダのリセットに約 2 秒かかります。

参照 InitReader

2　SAM API

アプリケーション開発時に提供されている” SamAPI.jar”をプロジェクトにインポートしてください。ライブラリをインポートする方法は「1.1.ライブラリのインポート」を参照してください。

必要なライブラリ

SamAPI.jar

2.1 SAM サービスのバインド

InitInstance

目的 SAM サービスをバインドします。

書式 SamManager InitInstance(Context context);

引数 context は使用中です。

使用例

```
private SamManager m_SM;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    m_SM = SamManager.InitInstance(this);
}
```

戻り値 成功の場合 SamManager のインスタンスが返ります。失敗の場合 Null が返ります。

ExecuteApdu

目的 SAM サービスをバインドします。

書式 ClResult ExecuteApdu(int[] cmd, ApduOutputData outputData);

引数

int[] cmd
[入力]int 型配列(APDU コマンド)。
ApduOutputData outputData
[出力] ApduOutputData オブジェクト。

使用例

```
int[] cmd={0x00,0x01,0x02,0x03,0x04,0x05};

ApduOutputData outputData=new ApduOutputData();
{
    tvOutput.setText("Len is " + outputData.length + "\nData is " +
    intArrayToHex(outputData.outputData));
 }
//Details of the ApduOutputData object
//public class ApduOutputData {
//              public int[] outputData;
//              public int length;
//}
```

戻り値 成功の場合 ClResult.S_OK が返ります。失敗の場合 ClResult.S_ERR が返ります。

Release

目的	SAM サービスを開放します。
書式	void Release();

使用例	@Override
	protected void onDestroy() {
	super.onDestroy();
	m_SM.Release();
	}

参照	InitInstance

2.2 サービス情報

Get_SamService

目的	デバイスの SAM サービスのバージョンを取得します。
書式	String Get_SamServiceVer();
使用例	String ver = m_SM.Get_SamServiceVer();

付録 1　戻り値一覧

値	意味
ClResult.S_OK	要求が正常に終了しました。
ClResult.S_ERR	不明なエラー。
ClResult.ERR_NotSupport	シンボルがサポートされていません。
ClResult.ERR_InvalidParameter	パラメータの誤りです。

付録 2　スキャナエンジン設定

RS30 シリーズモバイルコンピュータは以下のリーダ種別をサポートしています。リーダの可用性はモバイルコンピュータのハードウェアに依存します。

1D	CCD	SM1
1D	レーザー	SE955
2D	イメージャ	SE4500

対応シンボル体系

搭載されているスキャンエンジンにより、以下のシンボル体系をサポートしています。

		CCD	レーザー	2D
Codabar		○	○	○
Code 11		×	○	○
Code 39	Code 39	○	○	○
	Trioptic Code 39	×	○	○
	Italian Pharmacode (Code 32)	○	○	○
Code 93		○	○	○
Code 128	Code 128	○	○	○
	GS1-128 (EAN-128)	○	○	○
	ISBT 128	○	○	○
Code 2 of 5	Chinese 25	×	○	○
	Industrial 25 (Discrete 25)	○	○	○
	Interleaved 25	○	○	○
	Convert Interleaved 25 to EAN-13	×	○	○
	Matrix 25	×	×	○
Composite Code	Composite CC-A/B	×	×	○
	Composite CC-C	×	×	○
	Compostie TLC 39	×	×	○
GS1 DataBar (RSS)	GS1 DataBar-14 (RSS-14)	○	○	○
	GS1 DataBar Limited (RSS Limited)	○	○	○
	GS1 DataBar Expanded (RSS Expanded)	○	○	○
	Convert to UPC/EAN	×	○	○
Inverse	Inverse 1D barcodes	×	×	○
Korean 3 of 5		×	×	○
MSI		○	○	○
Postal Codes	Australian Postal	×	×	○
	Japan Postal	×	×	○
	Netherlands KIX Code	×	×	○
	US Postnet	×	×	○
	US Planet	×	×	○
	UK Postal	×	×	○
EAN/UPC	EAN-8	○	○	○
	EAN-8 Extend	○	○	○
	EAN-13	○	○	○
	Bookland EAN (ISBN)	○	○	○
	ISSN EAN	×	×	○
	UPC-A	○	○	○
	UPC-E	○	○	○
	Convert UPC-E to UPC-A	○	○	○
	UPC-E1	○	○	○
	Convert UPC-E1 to UPC-A	○	○	○
2D Symbologies	Aztec	×	×	○
	Data Matrix	×	×	○
	Maxicode	×	×	○
	MicroPDF417	×	×	○
	MicroQR	×	×	○
	PDF417	×	×	○
	QR Code	×	×	○

付録 3　サンプルコード

```
package com.example.cipherlab;

import android.app.Activity;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Bundle;
import android.widget.TextView;

public class MainActivity extends Activity {

    private TextView tv1 = null;
    private IntentFilter filter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        tv1 = (TextView)findViewById(R.id.tv1);

        // Register an intent filter to get the intent we want.
        filter = new IntentFilter();
        filter.addAction("com.cipherlab.barcodebaseapi.PASS_DATA_2_APP");
        registerReceiver(myDataReceiver, filter);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        unregisterReceiver(myDataReceiver);
    }

    // Create a broadcast object to get the intent sent from the service.
    private final BroadcastReceiver myDataReceiver = new BroadcastReceiver() {
        @Override
        public void onReceive(Context context, Intent intent) {
        // If the intent of the Intent_SOFTTRIGGER_DATA string is received,
        // the following statements are excuted.
            if (intent.getAction().equals("com.cipherlab.barcodebaseapi.PASS_DATA_2_APP")) {

                tv1.setText("");

                // Fetch the data along with the intent.
                String data = intent.getStringExtra("Decoder_Data");
                // Fetch the original data along with the intent (not change //to UTF-8 Format)
                byte [] buffer = intent.getByteArrayExtra("Decoder_DataArray");
                // Fetch the code type along with the intent
                int iCodeType = intent.getIntExtra("Decoder_CodeType", 0);

                // Display the data.
                tv1.setText(data);

            }

        }
    };
}
```

[注]：enableKeyboardEmulation は「KeyboardEmulationType.InputMethod」または「KeyboardEmulationType.KeyEvent」に設定されているものとします。なので、このサンプルでは、その後、ブロードキャストインテントメッセージを受け取ることができます。
詳しくは、Set_ReaderOutputConfiguration を参照してください。

参考

『付録 3　サンプルコード』中の「intent.getIntExtra("Decoder_CodeType", 0)」で取得できるバーコード種別の値は次の通りです。

バーコード種類	値
Unknown	0
Composite_CC_A	0x2F
Composite_CC_B	0x37
Korean_3_of_5	0x38
ISSN	0x39
ISBT_128_Concatenation	0x3F
ISBT_128	0x40
Code_39	0x41
Italian_Pharmacode	0x42
French_Pharmacode	0x43
Interleaved_2_of_5	0x45
Matrix_2_of_5	0x46
Codabar	0x47
Code_93	0x48
Code_128	0x49
UPC_E0	0x4A
UPC_E0_with_Addon_2	0x4B
UPC_E0_with_Addon_5	0x4C
EAN8	0x4D
EAN8_with_Addon_2	0x4E
EAN8_with_Addon_5	0x4F
EAN13	0x50
EAN13_with_Addon_2	0x51
EAN13_with_Addon_5	0x52
MSI	0x53
Plessey	0x54
EAN_128	0x55
Telepen	0x5A
GS1_DataBar14	0x5B
GS1_DataBarLimited	0x5C
GS1_DataBarExpanded	0x5D
UPC_A	0x5E
UPC_A_with_Addon_2	0x5F
UPC_A_with_Addon_5	0x60
UPC_E1	0x61
UPC_E1_with_Addon_2	0x62
UPC_E1_with_Addon_5	0x63
TLC_39	0x64
Trioptic_Code_39	0x65
Bookland	0x66
Code_11	0x67
Code_39_Full_ASCII	0x68
IATA25	0x69
Industrial_2_of_5	0x6A
PDF417	0x6B
Micro_PDF	0x6C
Data_Matrix	0x6D
Maxicode	0x6E
QR_Code	0x6F
US_Postnet	0x70
US_Planet	0x71
UK_Postal	0x72
Japan_Postal	0x73
Australian_Postal	0x74
Dutch_Postal	0x75
Composite_CC_C	0x76
Macro_PDF	0x77
Coupon_Code	0x78
Chinese_2_of_5	0x79

Aztec	0x7A
Micro_QR_Code	0x7B
USPS_4CB_One_Code_Intelligent_Mail	0x7C
UPU_FICS_Postal	0x7D
Macro_Micro_PDF417	0x7E

Blank page