



Barcode 7.0 Interface Instruction

iDTRONIC GmbH
Donnersbergweg 1
67059 Ludwigshafen
Germany/Deutschland

Phone: +49 621 6690094-0
Fax: +49 621 6690094-9
E-Mail: info@idtronic.de
Web: idtronic.de

Issue 2.0.0 · 2018-05-31
– 28. June 2019 –

Subject to alteration without prior notice.
© Copyright iDTRONIC GmbH 2019
Printed in Germany

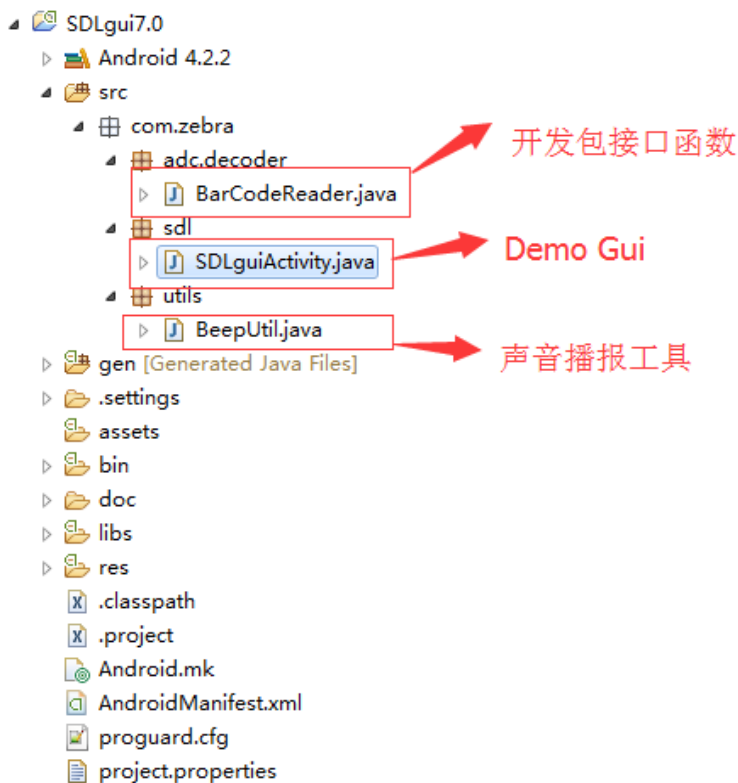
Contents

1	SDK instruction.....	4
2	Open SDK Demo.....	4
3	Reference Demo Gui: SDLguiActivity	4
3.1	Loading SO file	4
3.2	Turn on Scanner	5
3.3	Release Scanner Resources.....	5
3.4	Realization Decoding Interface: BarCodeReader.DecodeCallback, completed scanner decoding.	5
3.5	Example: doDecode	7
3.6	Example of continuous scanning, doHandsFree	7
3.7	Setting scanner parameter, doSetParam.....	8
3.8	Get the sweep dock property value, doGetProp	8
3.9	Disable all Code Types	9
3.10	Recover the scanner to default setting.....	9
3.11	Scanner Key Broadcast Value	9
4	Revision History	9

1 SDK instruction



2 Open SDK Demo



3 Reference Demo Gui: SDLguiActivity

3.1 Loading SO file

```
static
{
    System.loadLibrary("IAL");
    System.loadLibrary("SDL");

    if(android.os.Build.VERSION.SDK_INT >= 19)
        System.loadLibrary("barcodereader44"); // Android 4.4
    else
        if(android.os.Build.VERSION.SDK_INT >= 18)
            System.loadLibrary("barcodereader43"); // Android 4.3
        else
            System.loadLibrary("barcodereader"); // Android 2.3 - Android 4.2
}
```

3.2 Turn on Scanner

```

@Override
protected void onResume()
{
    super.onResume();
    state = STATE_IDLE;
    try {
        dspStat(getResources().getString(R.string.app_name) + " v" + this.getPackageManager
        if(android.os.Build.VERSION.SDK_INT >= 18)
            bcr = BarcodeReader.open(1, getApplicationContext()); // Android 4.3 and above
        else
            bcr = BarcodeReader.open(0); // Android 2.3
        if (bcr == null) {
            dspErr("open failed");
            return;
        }
        bcr.setDecodeCallback(this);
        bcr.setErrorCallback(this);

        bcr.setParameter(765, 0); // MTK must be set
        bcr.setParameter(137, 0); // 可对同一码进行扫描

    } catch (Exception e) {
        dspErr("open excp:" + e);
    }
}

```

前摄

后摄

MTK开发平台必须设置如此

如此设置可对同一码进行扫码，不设置不能对同一码连续扫码

3.3 Release Scanner Resources

```

@Override
protected void onPause()
{
    super.onPause();
    if (bcr != null)
    {
        setIdle();
        bcr.release();
        bcr = null;
    }
}

```

3.4 Realization Decoding Interface: BarcodeReader.DecodeCallback, completed scanner decoding.

```

// BarcodeReader.DecodeCallback override
public void onDecodeComplete(int symbology, int length, byte[] data, BarcodeReader reader)
{
    if (state == STATE_DECODE)
        state = STATE_IDLE;

    // Get the decode count
    if (length == BarcodeReader.DECODE_STATUS_MULTI_DEC_COUNT)
        decCount = symbology;

    if (length > 0)
    {
        if (isHandsFree() == false && isAutoAim() == false)
            bcr.stopDecode();

        ++decodes;

        if (symbology == 0x69) // signature capture
        {
            if (sigcapImage)
            {
                Bitmap bmSig = null;
                int schdr = 6;
                if (length > schdr)
                    bmSig = BitmapFactory.decodeByteArray(data, schdr, length - schdr);

                if (bmSig != null)
                    snapScreen(bmSig);

            }
            else
                dspErr("OnDecodeComplete: SigCap no bitmap");
        }
        decodeStatString += new String("[ " + decodes + " ] type: " + symbology + " len: " + length);
        decodeDataString += new String(data);
    }
    else
    {
    }
}

```

```

        if (symbology == 0x99) //type 99?
        {
            symbology = data[0];
            int n = data[1];
            int s = 2;
            int d = 0;
            int len = 0;
            byte d99[] = new byte[data.length];
            for (int i=0; i<n; ++i)
            {
                s += 2;
                len = data[s++];
                System.arraycopy(data, s, d99, d, len);
                s += len;
                d += len;
            }
            d99[d] = 0;
            data = d99;
        }
        decodeStatString += new String("[ " + decodes + " ] type: " + symbology + " len: " + length);
        decodeDataString += new String(data);
        dspStat(decodeStatString);
        dspData(decodeDataString);

        if(decCount > 1) // Add the next line only if multiple decode
        {
            decodeStatString += new String(" ; ");
            decodeDataString += new String(" ; ");
        }
        else
        {
            decodeDataString = new String("");
            decodeStatString = new String("");
        }
    }

    if (beepMode)
        beep();
}
else // no-decode
{
    dspData("");
    switch (length)
    {
        case BarCodeReader.DECODE_STATUS_TIMEOUT:
            dspStat("decode timed out");
            break;

        case BarCodeReader.DECODE_STATUS_CANCELED:
            dspStat("decode cancelled");
            break;

        case BarCodeReader.DECODE_STATUS_ERROR:
        default:
            dspStat("decode failed");
            break;
    }
}
}
//}
},

```

3.5 Example: doDecode

```
private void doDecode()
{
    if (setIdle() != STATE_IDLE)
        return;

    state = STATE_DECODE;
    decCount = 0;
    decodeDataString = new String("");
    decodeStatString = new String("");
    dspData("");
    int status = bcr.getNumProperty(BarcodeReader.PropertyNum.ENGINE_STATUS);
    decodeStatString = ("[Decoding] Engine Status 0x" + Integer.toHexString(status));
    dspStat(decodeStatString);
    decodeStatString = "";

    try
    {
        bcr.startDecode(); // start decode (callback gets results)
    }
    catch (Exception e)
    {
        dspErr("open excp:" + e);
    }
}
```

3.6 Example of continuous scanning, doHandsFree

```
private void doHandsFree()
{
    if (setIdle() != STATE_IDLE)
        return;

    int ret = bcr.startHandsFreeDecode(BarcodeReader.ParamVal.HANDSFREE);
    if (ret != BarcodeReader.BCR_SUCCESS)
        dspStat("startHandFree FAILED");
    else
    {
        trigMode = BarcodeReader.ParamVal.HANDSFREE;
        state = STATE_HANDSFREE;

        decodeDataString = new String("");
        decodeStatString = new String("");
        dspData("");
        dspStat("HandsFree decoding");
    }
}
```

3.7 Setting scanner parameter, doSetParam

```
private int doSetParam(int num, int val)
{
    String s = "";
    int ret = bcr.setParameter(num, val);
    if (ret != BarcodeReader.BCR_ERROR)
    {
        if (num == BarcodeReader.ParamNum.PRIM_TRIG_MODE)
        {
            trigMode = val;
            if (val == BarcodeReader.ParamVal.HANDSFREE)
            {
                s = "HandsFree";
            }
            else if (val == BarcodeReader.ParamVal.AUTO_AIM)
            {
                s = "AutoAim";
                ret = bcr.startHandsFreeDecode(BarcodeReader.ParamVal.AUTO_AIM);
                if (ret != BarcodeReader.BCR_SUCCESS)
                {
                    dspErr("AUtoAim start FAILED");
                }
            }
            else if (val == BarcodeReader.ParamVal.LEVEL)
            {
                s = "Level";
            }
        }
        else if (num == BarcodeReader.ParamNum.IMG_VIDEOVF)
        {
            if (snapPreview=(val == 1) )
                s = "SnapPreview";
        }
    }
    else
        s = " FAILED (" + ret + ")";

    dspStat("Set #" + num + " to " + val + " " + s);
    return ret;
}
```

3.8 Get the sweep dock property value, doGetProp

```
private void doGetProp()
{
    setIdle();
    String sMod = bcr.getStrProperty(BarcodeReader.PropertyNum.MODEL_NUMBER).trim();
    String sSer = bcr.getStrProperty(BarcodeReader.PropertyNum.SERIAL_NUM).trim();
    String sImg = bcr.getStrProperty(BarcodeReader.PropertyNum.IMGKIT_VER).trim();
    String sEng = bcr.getStrProperty(BarcodeReader.PropertyNum.ENGINE_VER).trim();
    String sBTLD = bcr.getStrProperty(BarcodeReader.PropertyNum.BTLD_FW_VER).trim();

    int buf = bcr.getNumProperty(BarcodeReader.PropertyNum.MAX_FRAME_BUFFER_SIZE);
    int hRes = bcr.getNumProperty(BarcodeReader.PropertyNum.HORIZONTAL_RES);
    int vRes = bcr.getNumProperty(BarcodeReader.PropertyNum.VERTICAL_RES);

    String s = "Model:\t\t" + sMod + "\n";
    s += "Serial:\t\t" + sSer + "\n";
    s += "Bytes:\t\t" + buf + "\n";
    s += "V-Res:\t\t" + vRes + "\n";
    s += "H-Res:\t\t" + hRes + "\n";
    s += "ImgKit:\t\t" + sImg + "\n";
    s += "Engine:\t\t" + sEng + "\n";
    s += "FW BTLD:\t\t" + sBTLD + "\n";

    AlertDialog.Builder dlg = new AlertDialog.Builder(this);
    if (dlg != null)
    {
        dlg.setTitle("SDL Properties");
        dlg.setMessage(s);
        dlg.setPositiveButton("ok", null);
        dlg.show();
    }
}
```


3.9 Disable all Code Types

```
dspStat("All Paramters Disabled");
bcr.disableAllCodeTypes();
```

3.10 Recover the scanner to default setting

```
private void doDefaultParams()
{
    setIdle();
    bcr.setDefaultParameters();
    dspStat("Parameters Defaulted");

    // reset modes
    snapPreview = false;
    int val = bcr.getNumParameter(BarCodeReader.ParamNum.PRIM_TRIG_MODE);
    if (val != BarCodeReader.BCR_ERROR)
        trigMode = val;
}
```

3.11 Scanner Key Broadcast Value

Pressing scanner key broadcast value

: shmaker.android.intent.action.SCANER_KEYEVENT_DOWN

Scanner key lifts broadcast value

: shmaker.android.intent.action.SCANER_KEYEVENT_UP

Long Pressing scanner key broadcast value

shmaker.android.intent.action.SCANER_KEYEVENT_LONG

The above three broadcasts can only be received as third-party apps, but, third party app cannot send these three broadcasts.

remark

Development can be done through install directly.

4 Revision History

2019-06-29	2.0.0	Initial release.