

# ZKFinger SDK

## for Java

**Version: 2.0**

---

**Date: Sep, 2016**



## 修订记录

日期	版本	描述	作者
2016-05-21	1.0.0	基础版本	陈建兴
2016-06-01	1.0.1	增加外部图像接口	陈建兴
2016-09-18	2.0.0	统一接口，保留 1.0 接口	陈建兴

## 目录

1 ZKFinger SDK 概述 .....	1
2 开发环境搭建 .....	1
2.1 导入 ZKFingerReader.jar .....	1
2.2 SDK 部署 .....	1
3 ZKFinger SDK .....	2
3.1 FingerprintSensorEx.class .....	2
3.1.1 Init .....	2
3.1.2 Terminate .....	3
3.1.3 OpenDevice .....	3
3.1.4 CloseDevice .....	3
3.1.5 SetParameters .....	4
3.1.6 GetParameters .....	5
3.1.7 AcquireFingerprint .....	5
3.1.8 AcquireFingerprintImage .....	6
3.1.9 DBInit .....	7
3.1.10 DBFree .....	7
3.1.11 DBAdd .....	7
3.1.12 DBDel .....	8
3.1.13 DBCount .....	8
3.1.14 DBMatch .....	9
3.1.15 DBIdentify .....	9
3.1.16 DBMerge .....	10
3.1.17 ExtractFromImage .....	11
3.1.18 BlobToBase64 .....	11
3.1.19 Base64ToBlob .....	12
4 附录 .....	12
4.1 参数代码 .....	12
4.2 错误代码 .....	13

感谢您使用中控的ZKFinger SDK，在使用前请仔细阅读ZKFinger SDK概述，以便您能更快地掌握并使用ZKFinger SDK。

## 文档隐私权说明

非经过本公司书面同意，任何单位和个人不得擅自摘抄、复制本手册内容的部分或全部，并不得以任何形式传播。本手册中描述的产品中，可能包含我司及其可能存在的许可人享有版权的软件，除非获得相关权利人的许可，否则，任何人不得以任何形式对前述软件进行复制、分发、修改、摘录、反编译、反汇编、解密、反向工程、出租、转让等侵犯软件版权的行为。

## 文档使用说明

由于ZKFinger SDK软件功能不断扩充，ZKFinger SDK文档版本也会不断地升级，所以在使用ZKFinger SDK软件时，请详细阅读ZKFinger SDK文档内容。如有上诉原因给您造成的不便，敬请谅解，您也可以联系我们文档编写人，联系信息如下，谢谢！

公司：厦门中控智慧信息技术有限公司

地址：厦门市集美区诚毅北大街软件园三期8号2002（B02 21楼）

电话：0592-7791134-3168

网站：[www.zkteco.com](http://www.zkteco.com)

邮箱：[sdksupport@zkteco.com](mailto:sdksupport@zkteco.com)

## 1 ZKFinger SDK 概述

ZKFinger SDK是中控提供给开发者的一套应用程序接口，具有统一管理中控指纹采集器设备模块的功能。开发者可以使用各个类中函数，开发操作Java的应用。

ZKFinger SDK包括以下功能：

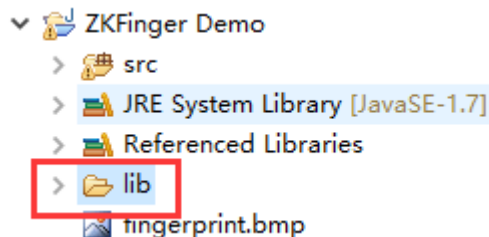
**指纹采集器设备：**主要是操作指纹采集,算法操作，如初始化设备、打开设备，关闭设备，1:1,1:N等；

## 2 开发环境搭建

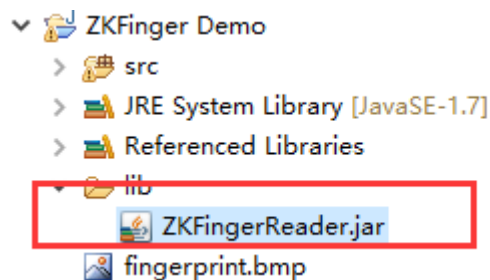
### 2.1 导入 ZKFingerReader.jar

打开 SDK文件夹，将java/lib目录中的ZKFingerReader.jar导入到应用程序开发工具中（以eclipse为例）

步骤 1：在工程目录添加lib目录；



步骤 2：复制 ZKFingerReader.jar，鼠标右键单击工程 lib 目录，粘贴，即可实现。



### 2.2 SDK 部署

安装ZKFinger SDK 5.x/ZKOnline SDK 5.x。。

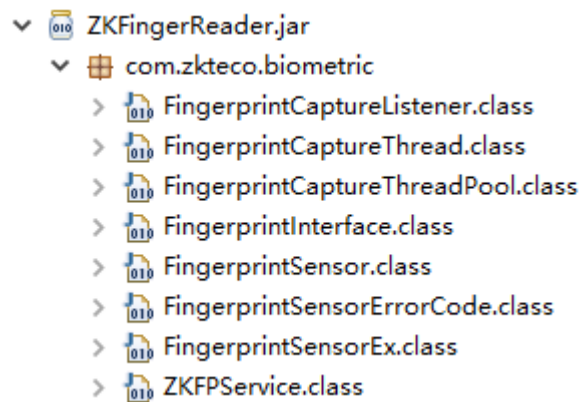
## 3 ZKFinger SDK

ZKFinger SDK 将各个功能模块抽象成类，用户通过调用类中方法完成对底层硬件设备的操作，以及对指纹算法的处理。

ZKFinger SDK包括指纹采集、算法处理类等。关键类对应的类型如下图所示：

类名	类型
<code>com.zkteco.biometric.FingerprintSensorEx</code>	指静采集器设备类，算法处理等

SDK包结构如下：



### 3.1 FingerprintSensorEx.class

FingerprintSensorEx.class 操作指指纹仪设备类。如打开设备、关闭设备、采集指纹，比对指纹等。

#### 3.1.1 Init

[函数]

**public static int Init ()**

[功能]

初始化资源

[参数]

[返回值]

0 表示成功，其他参考错误码说明

[注意]

### 3.1.2 Terminate

[函数]

**public static int Terminate ()**

[功能]

释放资源

[参数]

[返回值]

0 表示成功，其他参考错误码说明

[注意]

### 3.1.3 OpenDevice

[函数]

**public static long OpenDevice (int index)**

[功能]

连接设备

[参数]

**index**

设备索引号，该值是接入采集器总数决定的。

例如：

当采集器总数为 1 时，则index的值为 0；

当采集器总数为 2 时，index的值为 0 或 1；

.....

[返回值]

设备句柄，值为 0 时打开失败

[注意]

### 3.1.4 CloseDevice

[函数]

**public static int CloseDevice(long devHandle)**

[功能]

关闭设备

[参数]

**devHandle**

设备句柄

[返回值]

0 表示成功，其他见错误代码说明

[注意]

### 3.1.5 SetParameters

[函数]

```
public static int SetParameters(long devHandle, int code, byte[] paramValue, int size)
```

[功能]

设置参数

[参数]

**devHandle**

设备句柄

**code**

参数代码

**paramValue**

参数值

**size**

参数数据长度

[返回值]

0 表示成功，其他见错误代码说明

[注意]

[示例]

```
byte[] value = new byte[4];
in len = 4;    //sizeof int
int FakeFunOn = 1;
value[0] = FakeFunOn & 0xFF;
value[1] = (FakeFunOn & 0xFF00) >> 8;
value[2] = (FakeFunOn & 0xFF0000) >> 16;
value[3] = (FakeFunOn & 0xFF000000) >> 24;
```



```
int ret = SetParameter(2002, value, len); //set FakeFunOn
```

### 3.1.6 GetParameters

[函数]

```
public static int GetParameters(long devHandle, int code, byte[] paramValue, int[] size)
```

[功能]

获取参数

[参数]

**devHandle**

设备句柄

**code**

参数代码

**paramValue**

参数值

**size**

参数数据长度

[返回值]

0 表示成功，其他见错误代码说明

[注意]

[示例]

```
byte[] value = new byte[4];
int[] len = new int[1];
len[0] = 4;
int ret = GetParameter(1, value, len); //image width
if (0 == ret)
{
    //convert byte array to int
}
```

### 3.1.7 AcquireFingerprint

[函数]

```
public static int AcquireFingerprint(long devHandle, byte[] imgBuffer,  
byte[] template, int[] size)
```

[功能]

采集指纹图像，指纹模板

[参数]

**devHandle**

设备句柄

**imgBuffer**

图像数据(预分配 width\*height Bytes)

**template**

模板数据(预分配 2048 Bytes)

**size**

返回模板数据长度

[返回值]

0 表示成功，其他见错误代码说明

[注意]

### 3.1.8 AcquireFingerprintImage

[函数]

```
public static int AcquireFingerprintImage(long devHandle, byte[]  
imgBuffer)
```

[功能]

采集指纹图像

[参数]

**devHandle**

设备句柄

**imgBuffer**

图像数据(预分配 width\*height Bytes)

[返回值]

0 表示成功，其他见错误代码说明

[注意]

### 3.1.9 DBInit

**[函数]**

**public static long DBInit()**

**[功能]**

初始化算法库。

**[参数说明]**

**[返回值]**

算法句柄

**[注意]**

### 3.1.10 DBFree

**[函数]**

**public static int DBFree(long dbHandle)**

**[功能]**

释放算法库。

**[参数说明]**

**dbHandle**

算法句柄

**[返回值]**

算法句柄，0 表示失败

**[注意]**

### 3.1.11 DBAdd

**[函数]**

**public int DBAdd(long dbHandle , int fid, byte[] regTemplate)**

**[功能]**

添加登记模板到内存。

**[参数说明]**

**dbHandle**

算法句柄

**Fid**

指纹 ID

**regTemplate**

登记模板

**[返回值]**

0 表示成功，其他见错误代码说明

**[注意]**

### 3.1.12 DBDel

**[函数]**

**public int DBDel (long dbHandle , int fid)**

**[功能]**

从内存中删除一枚登记模板。

**[参数说明]**

**dbHandle**

算法句柄

**Fid**

指纹 ID

**[返回值]**

0 表示成功，其他见错误代码说明

**[注意]**

### 3.1.13 DBCount

**[函数]**

**public int DBCount (long dbHandle )**

**[功能]**

获取内存中指纹数。

**[参数说明]**

**dbHandle**

算法句柄

**[返回值]**

$\geq 0$  表示指纹模板数,  $< 0$  见错误代码说明

**[注意]**

### 3.1.14 DBMatch

**[函数]**

**public int DBMatch(long dbHandle , byte[] temp1, byte[] temp2)**

**[功能]**

比对两枚指纹模板。

**[参数说明]**

**dbHandle**

算法句柄

**temp1**

指纹模板 1

**temp2**

指纹模板 2

**[返回值]**

返回比对分数( $< 0$  见错误代码说明)

**[注意]**

### 3.1.15 DBIdentify

**[函数]**

**public int DBIdentify(long dbHandle , byte[] template, int[] fid, int[] socre)**

**[功能]**

1:N识别。

**[参数说明]**

**dbHandle**

算法句柄

**template**

指纹模板

**Fid**

返回指纹 ID

**Score**

返回比对分数

**[返回值]**

0 表示成功，其他见错误代码说明

**[注意]**

### 3.1.16 DBMerge

**[函数]**

```
public int DBMerge(long dbHandle , byte[] temp1, byte[] temp2,  
byte[] temp3, byte[] regTemp, int[] regTempLen)
```

**[功能]**

合并登记模板。

**[参数说明]**

**dbHandle**

算法句柄

**temp1**

预登记模板 1

**temp2**

预登记模板 2

**temp3**

预登记模板 3

**regTemp**

返回登记模板

**regTempLen**

返回登记模板长度

**[返回值]**

0 表示成功，其他见错误代码说明

**[注意]**

### 3.1.17 ExtractFromImage

[函数]

```
public int ExtractFromImage(long dbHandle , String filePath, int  
DPI, byte[] template, int[] size)
```

[功能]

从 BMP 或 JPG 提取指纹模板

[参数说明]

**dbHandle**

算法句柄

**FilePath**

图片全路径

**DPI**

图像 DPI

**Template**

返回指纹模板

**Size**

返回指纹模板长度

[返回值]

0 表示成功，其他见错误代码说明

[注意]

仅标准版支持该功能

### 3.1.18 BlobToBase64

[函数]

```
public static String BlobToBase64(byte[] buf, int cbBuf)
```

[功能]

byte 数组转 Base64 字符串

[参数说明]

**buf**

二进制数据

**cbBuf**



数据长度

[返回值]

Base64 格式字符串

### 3.1.19 Base64ToBlob

[函数]

```
public static int Base64ToBlob(String strBase64, byte[] buf, int  
cbBuf)
```

[功能]

Base64 字符串转二进制数组

[参数说明]

**strBase64**

Base64 格式字符串

**buf**

返回二进制数组

**cbBuf**

buf 数组长度

[返回值]

返回二进制数据长度，0 表示失败

## 4 附录

### 4.1 参数代码

参数代码	属性	数据类型	描述
1	只读	Int	图像宽
2	只读	Int	图像高
3	读写 (目前只有 LIVEID20R 可写)	Int	图像 DPI(儿童建议 设置 750/1000)
106	只读	Int	图像数据大小
1015	只读	4 字节 Byte 数组	VID&PID(前 2 字



			节 VID,后 2 字节 PID)
<b>2002</b>	读 写 ( 目 前 只 有 LIVEID20R 支持)	Int	防假开关(1 打开 0 关闭)
<b>2004</b>	只读	Int	低五位全为 1 表示 真 手 指 (value&31==31)
<b>1101</b>	只读	String	厂商信息
<b>1102</b>	只读	String	产品名
<b>1103</b>	只读	String	设备序列号
<b>101</b>	只写(非 LIVE20R 需调用关闭)	Int	1 表示 闪 白 灯 ;0 表示关闭
<b>102</b>	只写(非 LIVE20R 需调用关闭)	Int	1 表示 闪 绿 灯 ;0 表示关闭
<b>103</b>	只写(非 LIVE20R 需调用关闭)	Int	1 表示 闪 红 灯 ;0 表示关闭
<b>104</b>	只写(LIVE20R 不支持)	Int	1 表示开启蜂鸣;0 表示关闭
10001	读写(仅 ISO/ANSI 版本支持)	Int	0 表示 ANSI;1 表示 ISO

## 4.2 错误代码

错误码	备注
0	成功
1	已经初始化
-1001	失败
-1002	连接设备失败
-1003	设备未连接
-1	初始化算法库失败
-2	初始化采集库失败

-3	无设备连接
-4	接口暂不支持
-5	无效参数
-6	打开设备失败
-7	无效句柄
-8	取像失败
-9	提取指纹模板失败
-10	中断操作
-11	内存不足
-12	当前正在采集指纹
-13	添加指纹模板到内存失败
-14	添加指纹模板失败
-17	操作失败
-18	取消采集
-20	比对指纹失败
-22	合并登记指纹模板失败
-24	处理图像失败