



# **ZKFinger SDK**

## **for Java**

**Version: 2.0**

---

**Date: Sep 2016**



### Revision Records

Date	Version	Description	Author
2016-05-21	1.0.0	Basic version	Chen Jianxing
2016-06-01	1.0.1	Added external image interfaces.	Chen Jianxing
2016-09-18	2.0.0	<b>Added 2.0 interface, keep old interface</b>	Chen Jianxing

## Contents

1. Overview of ZKFinger SDK .....	1
2. Development Environment Setup.....	1
2.1 Importing ZKFingerReader.jar .....	1
2.2 Deploying SDK .....	1
3 ZKFinger SDK .....	2
3.1 <b>FingerprintSensor.class</b> .....	2
3.1.1 Init.....	2
3.1.2 Terminate .....	3
3.1.3 OpenDevice.....	3
3.1.4 CloseDevice .....	4
3.1.5 SetParameters .....	4
3.1.6 GetParameters .....	5
3.1.7 AcquireFingerprint .....	6
3.1.8 AcquireFingerprintImage.....	6
3.1.9 DBInit .....	7
3.1.10 DBFree.....	7
3.1.11 DBAdd.....	8
3.1.12 DBDel .....	8
3.1.13 DBCount .....	9
3.1.14 DBMatch.....	9
3.1.15 DBIdentify .....	10
3.1.16 DBMerge.....	10
3.1.17 ExtractFromImage .....	11
3.1.18 BlobToBase64 .....	12
3.1.19 Base64ToBlob .....	12
4 Appendixes.....	13
4.1 Parameter Codes .....	13
4.2 Error Code .....	14



Thank you for using ZKTeco ZKFinger SDK. Please read this document carefully before use to fast learn how to use ZKFinger SDK.

### **Privacy Policy**

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of ZKTeco. The product described in this manual may include copyrighted software of ZKTeco and possible licensors. Customers shall not reproduce, distribute, modify, decompile, disassemble, decrypt, extract, reverse engineer, lease, assign, or sublicense the said software in any manner, unless such restrictions are prohibited by applicable laws or such actions are approved by respective copyright holders under license.

### **Usage Description**

As functions of the ZKFinger SDK software are constantly expanded, ZKFinger SDK documentations will be upgrading. Therefore, please read ZKFinger SDK documents carefully when using the ZKFinger SDK software. We apologize for any inconvenience caused by the preceding reasons. You can also contact the authors of the documentations. Thank you.

Company: ZKTech (Xiamen) Software

Address: 22 nd Floor, No.8, Cheng Yi North Street of Software Park, Xiamen City, Fujian Province, China.

Telephone: 0592-7791134-3168

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

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

# 1. Overview of ZKFinger SDK

ZKFinger SDK is a set of application programming interfaces (APIs) developed by ZKTeco for development engineers. It is capable of managing ZKTeco fingerprint readers in a unified manner. Development engineers can use functions in different classes to develop Java-based applications.

ZKFinger SDK supports the following functions:

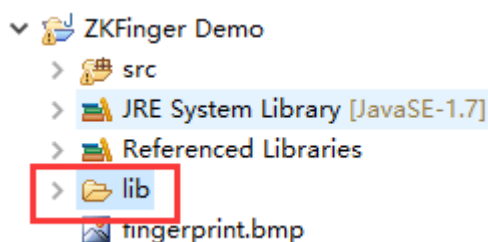
Fingerprint readers: ZKFinger SDK supports fingerprint capture and algorithm operations, including device initialization, device startup, device shutdown, 1:1 comparison, and 1:N comparison.

## 2. Development Environment Setup

### 2.1 Importing ZKFingerReader.jar

Open the **SDK** folder and import **ZKFingerReader.jar** in the **java/lib** directory to the application development tool (the following uses Eclipse as an example).

Step 1: Create the **lib** directory in the directory of a project.



Step 2: Copy **ZKFingerReader.jar**, right-click the **lib** directory and choose **Paste** to copy **ZKFingerReader.jar** into the **lib** directory.



### 2.2 Deploying SDK

Install ZKFinger SDK 5.x/ZKOnline SDK 5.x.

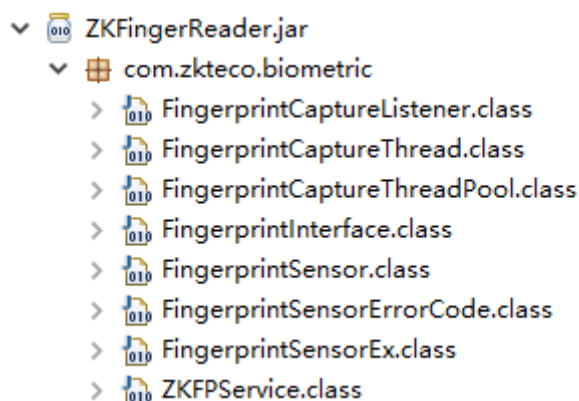
## 3 ZKFinger SDK

ZKFinger SDK abstracts function modules as classes. Users can call methods in classes to complete underlying hardware operations and processing of the fingerprint algorithm.

ZKFinger SDK includes the fingerprint reader class and algorithm handling class. The following table lists the types of the classes.

Class Name	Type
<b>com.zkteco.biometric.FingerprintSensorEx</b>	Fingerprint reader class, algorithm handling class

The following figure shows the structure of the SDK package.



### 3.1 FingerprintSensor.class

FingerprintSensor.class is a class for controlling fingerprint readers, which can be used to start and shut down a fingerprint reader, verify and identify.

#### 3.1.1 Init

[Function]

**public static int Init ()**

[Purpose]

This function is used to Initialize resource.

[Parameter Description]

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.2 Terminate

[Function]

**public static int Terminate ()**

[Purpose]

This function is used to release resource.

[Parameter Description]

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.3 OpenDevice

[Function]

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

[Purpose]

This function is used to connect to a device.

[Parameter Description]

**index**

Device index number. The value is determined based on the total number of connected fingerprint readers.

Example:

When a total of one fingerprint reader is connected, the index is 0.

When a total of two fingerprint readers are connected, the index is 0 or 1.

...

[Return Value]

0 fail

Device handle success

[Note]

### 3.1.4 CloseDevice

[Function]

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

[Purpose]

This function is used to connect to a device.

[Parameter Description]

**devHandle**

Device Handle

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.5 SetParameters

[Function]

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

[Purpose]

This function is used to set a parameter.

[Parameter Description]

**devHandle**

Device handle

**code**

Parameter code (See the Appendixes.)

**paramValue**

Parameter value

**size**

Parameter data length

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

[Example]

```
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

[Function]

```

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

```

[Purpose]

This function is used to acquire a parameter.

[Parameter Description]

**devHandle**

Device handle

**code**

Parameter code (See the Appendixes.)

**paramValue**

Parameter value

**size**

Parameter data length

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

[Example]

```

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

[Function]

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

[Purpose]

This function is used to extract a fingerprint image and template .

[Parameter Description]

**devHandle**

Device handle

**imgBuffer**

Image data(width \* height bytes)

**template**

Template data(2048 Bytes)

**size**

Length of the returned fingerprint template

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.8 AcquireFingerprintImage

[Function]

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

[Purpose]

This function is used to extract a fingerprint image

[Parameter Description]

**devHandle**

Device handle

**imgBuffer**

Image data (width\*height Bytes)

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.9 DBInit

[Function]

**public static long DBInit()**

[Purpose]

This function is used to Initialize algorithm library.

[Parameter Description]

[Return Value]

Catch handle

[Note]

### 3.1.10 DBFree

[Function]

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

[Purpose]

This function is used to release algorithm library.

[Parameter Description]

**dbHandle**

Catch handle

[Return Value]

Catch handle success

0 fail

[Note]

### 3.1.11 DBAdd

[Function]

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

[Purpose]

This function is used to add a registered template to the memory.

[Parameter Description]

**dbHandle**

Catch handle

**Fid**

Fingerprint ID

**regTemplate**

Registered template

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.12 DBDel

[Function]

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

[Purpose]

This function is used to delete a registered template from the memory.

[Parameter Description]

**dbHandle**

Catch handle

**Fid**

Fingerprint ID

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.13 DBCount

[Function]

**public int DBCount (long dbHandle )**

[Purpose]

This function is used to acquire the number of fingerprint images in the memory.

[Parameter Description]

**dbHandle**

Catch handle

[Return Value]

>=0 Fingerprint template count

<0 See the error code description.

[Note]

### 3.1.14 DBMatch

[Function]

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

[Purpose]

This function is used to compare two fingerprint templates.

[Parameter Description]

**dbHandle**

Catch handle

**temp1**

Fingerprint template 1

**temp2**

Fingerprint template 2

[Return Value]

The comparison score is returned.

<0 See the error code description.

[Note]

### 3.1.15 DBIdentify

[Function]

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

[Purpose]

This function is used to conduct 1:N comparison.

[Parameter Description]

**dbHandle**

Catch handle

**template**

Fingerprint template

**Fid**

Returned fingerprint ID

**Score**

Returned comparison score

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.16 DBMerge

[Function]

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

[Purpose]

This function is used to combine registered fingerprint templates.

[Parameter Description]

**dbHandle**

Catch handle

**temp1**

Preregistered template 1

**temp2**

Preregistered template 2

**temp3**

Preregistered template 3

**regTemp**

Returned registered template

**regTempLen**

Length of the returned registered template

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

### 3.1.17 ExtractFromImage

[Function]

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

[Purpose]

This function is used to extract a fingerprint template from a BMP or JPG file.

[Parameter Description]

**dbHandle**

Catch handle

**FilePath**

Full path of a picture file

**DPI**

Image DPI

**Template**

Returned fingerprint template

**Size**

Length of the returned fingerprint template

[Return Value]

0 Succeeded

Others See the error code description.

[Note]

**Only the SDK of the standard version supports this function.**

### 3.1.18 BlobToBase64

[Function]

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

[Purpose]

This function is used to convert byte[] to Base64 string.

[Parameter Description]

**buf**

Blob data

**cbBuf**

Data length

[Return Value]

Base64 string

[Note]

### 3.1.19 Base64ToBlob

[Function]

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

[Purpose]

This function is used to convert Base64 string to byte[]

[Parameter Description]

**strBase64**

Base64 string

**buf**

Returned blob data

**cbBuf**

The length of buf

[Return Value]

0 fail

the length of binary data success



## 4 Appendixes

### 4.1 Parameter Codes

Parameter Code	Property	Data Type	Description
<b>1</b>	Read-only	Int	Image width
<b>2</b>	Read-only	Int	Image height
<b>3</b>	Read-write (supported only by the LIVEID20R currently)	Int	Image DPI (750/1000 is recommended for children.)
<b>106</b>	Read-only	Int	Image data size
<b>1015</b>	Read-only	4-byte array	VID&PID (The former two bytes indicate VID and the latter two bytes indicate PID.)
<b>2002</b>	Read-write (supported only by the LIVEID20R currently)	Int	Anti-fake function (1: enable; 0: disable)
<b>2004</b>	Read-only	Int	A fingerprint image is true if the lower five bits are all 1's (value&31==31).
<b>1101</b>	Read-only	String	Vendor information
<b>1102</b>	Read-only	String	Product name
<b>1103</b>	Read-only	String	Device SN
<b>101</b>	Write-only (For devices other than the LIVE20R, a function needs to be called to turn off the light.)	Int	1 indicates that the white light blinks; 0 indicates that the white light is off.
<b>102</b>	Write-only (For devices other than the LIVE20R, a function needs to be called to turn off the light.)	Int	1 indicates that the green light blinks; 0 indicates that the green light is off.
<b>103</b>	Write-only (For devices other than the LIVE20R, a function needs to be called to turn off the light.)	Int	1 indicates that the red light blinks; 0 indicates that the red light is off.

Parameter Code	Property	Data Type	Description
<b>104</b>	Write-only (not supported by the LIVE20R)	Int	1 indicates that buzzing is started; 0 indicates that buzzing is turned off.
10001	Read-write (only supported by ISO/ANSI Version)	Int	0 ANSI378; 1 ISO 19794-2

## 4.2 Error Code

Error Code	Description
0	Succeeded
1	Initialized
-1001	Failed
-1002	Failed to connect to the device
-1003	Device not connected
-1	Failed to initialize the algorithm library
-2	Failed to initialize the capture library
-3	No device connected
-4	Not supported by the interface
-5	Invalid parameter
-6	Failed to start the device
-7	Invalid handle
-8	Failed to capture the image
-9	Failed to extract the fingerprint template
-10	Suspension operation
-11	Insufficient memory
-12	The fingerprint is being captured

Error Code	Description
-13	Failed to add the fingerprint template to the memory
-14	Failed to add the fingerprint template
-17	Operation failed
-18	Capture cancelled
-20	Fingerprint comparison failed
-22	Failed to combine registered fingerprint templates
-24	Image processing failed