

BS2_ScanFaceEx 1
..... 1
..... 1
..... 1
..... 1

Face API > BS2_ScanFaceEx

BS2_ScanFaceEx

[+ 2.7.1] Visual Face

```
#include "BS_API.h"
```

```
int BS2_ScanFaceEx(void* context, uint32_t deviceId, BS2FaceEx* faceEx,
uint8_t enrollmentThreshold, OnReadyToScan ptrReadyToScan);
```

BS2FaceEx

- [In] *context* : Context
- [In] *deviceId* :
- [Out] *faceEx* :
- [In] *enrollmentThreshold* : threshold .
BS2FaceConfig::enrollThreshold
- [Out] *ptrReadyToScan* : 가

BS_SDK_SUCCESS , 가

C++

```
int UserControl::scanFaceEx(BS2_DEVICE_ID id, BS2FaceEx* ptrFace, uint8_t&
numOfFace)
{
    if (!ptrFace)
        return BS_SDK_ERROR_INVALID_PARAM;

    const int MAX_RETRY = 2;
    int sdkResult = BS_SDK_SUCCESS;
```

```
int retryCount = ;

while (retryCount < MAX_RETRY)
{
    sdkResult = BS2_ScanFaceEx(context_, id, ptrFace,
BS2_FACE_ENROLL_THRESHOLD_DEFAULT, onReadyToScanFace);
    if (BS_SDK_SUCCESS != sdkResult)
    {
        TRACE("BS2_ScanFaceEx call failed: %d", sdkResult);
        retryCount++;
    }
    else
    {
        numOfFace++;
        break;
    }
}

return sdkResult;
}

BS2_ReleaseObject(uidObj);
```

C#

```
if (Util.IsYes())
{
    Console.WriteLine("How many faceEx would you like to register?");
    Console.Write(">> ");
    int numOfFace = Util.GetInput(1);
    if (< numOfFace)
    {
        int structSize = Marshal.SizeOf(typeof(BS2FaceExWarped));
        BS2FaceExWarped[] faceEx =
Util.AllocateStructureArray<BS2FaceExWarped>(1);
        userBlob[].faceExObjs = Marshal.AllocHGlobal(structSize *
numOfFace);
        IntPtr curFaceExObjs = userBlob[].faceExObjs;
        cbFaceOnReadyToScan = new API.OnReadyToScan(ReadyToScanForFace);

        for (int index = ; index < numOfFace;)
        {
            sdkResult = (BS2ErrorCode)API.BS2_ScanFaceEx(sdkContext,
deviceID, faceEx, (byte)BS2FaceEnrollThreshold.THRESHOLD_DEFAULT,
cbFaceOnReadyToScan);
            if (BS2ErrorCode.BS_SDK_SUCCESS != sdkResult)
                Console.WriteLine("BS2_ScanFaceEx call failed: %d",
sdkResult);
            else
            {
                userBlob[].user.numFaces++;
            }
        }
    }
}
```

```
        index++;
        faceEx[].faceIndex = (byte)index;
        Marshal.StructureToPtr(faceEx[], curFaceExObjs, false);
        curFaceExObjs += structSize;

        Thread.Sleep(100);
    }
}

cbFaceOnReadyToScan = null;
}
```

From:

<https://kb.supremainc.com/bs2sdk/> - **BioStar 2 Device SDK**

Permanent link:

https://kb.supremainc.com/bs2sdk./doku.php?id=ko:bs2_scanfaceex

Last update: **2024/05/28 19:48**