

**BS2\_PartialUpdateUserFaceEx** ..... 1

..... 1

..... 1

..... 2

..... 2

(C++) ..... 2

(C#) ..... 3

# BS2\_PartialUpdateUserFaceEx

[+ 2.8.3]

mask  
, BS2User infoMask

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

```
int BS2_PartialUpdateUserFaceEx(void* context, uint32_t deviceId,
BS2_USER_MASK mask, BS2UserFaceExBlob* userBlob, uint32_t userCount);
```

## BS2UserFaceExBlob

- [In] *context* : Context
- [In] *deviceId* :
- [In] *mask* : mask

0x0002	( , )
0x0004	
0x0008	
0x0010	PIN
0x0020	
0x0040	
0x0080	
0x0100	
0x0200	
0x0400	
0x0800	(FSF2, BS3)
0x1000	(FSF2, BS3)

- [In] *userBlob* :
- [In] *userCount* :

BS\_SDK\_SUCCESS , 가 .  
BS2\_EVENT\_USER\_UPDATE\_PARTIAL\_SUCCESS , 가 .  
BS2\_EVENT\_USER\_UPDATE\_PARTIAL\_FAIL , 가 .

[BS2\\_PartialUpdateUser](#)  
[BS2\\_PartialUpdateUserEx](#)  
[BS2\\_PartialUpdateUserSmall](#)  
[BS2\\_PartialUpdateUserSmallEx](#)  
[BS2\\_PartialUpdateUserFaceEx](#)

## (C++)

[sample\\_partialupdate.cpp](#)

```
BS2_USER_MASK maskWantUpdate = BS2_USER_MASK_SETTING |  
BS2_USER_MASK_SETTING_EX | BS2_USER_MASK_JOB;  
int sdkResult = BS_SDK_SUCCESS;  
  
BS2UserFaceExBlob userBlob = { , };  
BS2User& user = userBlob.user;  
BS2UserSetting& setting = userBlob.setting;  
BS2UserSettingEx& settingEx = userBlob.settingEx;  
  
setting.fingerAuthMode = BS2_AUTH_MODE_NONE;  
setting.cardAuthMode = BS2_AUTH_MODE_NONE;  
setting.idAuthMode = BS2_AUTH_MODE_NONE;  
  
settingEx.faceAuthMode = BS2_AUTH_MODE_NONE;  
settingEx.fingerprintAuthMode = BS2_AUTH_MODE_NONE;  
settingEx.cardAuthMode = BS2_AUTH_MODE_NONE;  
settingEx.idAuthMode = BS2_AUTH_MODE_NONE;  
  
if (BS_SDK_SUCCESS != (sdkResult = uc.getUserBlobUserID(user)))  
    return sdkResult;  
  
if ((maskWantUpdate & BS2_USER_MASK_SETTING) == BS2_USER_MASK_SETTING)  
{  
    if (BS_SDK_SUCCESS != (sdkResult =  
uc.getUserBlobExpiryDate(setting)))  
        return sdkResult;  
  
    if (BS_SDK_SUCCESS != (sdkResult =  
uc.getUserBlobPrivateAuthMode(setting, deviceInfo, deviceInfoEx)))
```

```
        return sdkResult;

        if (BS_SDK_SUCCESS != (sdkResult =
uc.getUserBlobSecurityLevel(setting)))
            return sdkResult;
    }

    if ((maskWantUpdate & BS2_USER_MASK_SETTING_EX) ==
BS2_USER_MASK_SETTING_EX)
    {
        if (BS_SDK_SUCCESS != (sdkResult =
uc.getUserBlobPrivateAuthModeEx(settingEx, deviceInfo, deviceInfoEx)))
            return sdkResult;
    }

    // ...

    if ((maskWantUpdate & BS2_USER_MASK_JOB) == BS2_USER_MASK_JOB)
    {
        if (BS_SDK_SUCCESS != (sdkResult =
uc.getUserBlobJobCode(userBlob.job)))
            return sdkResult;

        user.infoMask |= BS2_USER_INFO_MASK_JOB_CODE;
    }

    // ...

    user.numCards = ;
    if ((maskWantUpdate & BS2_USER_MASK_CARD) == BS2_USER_MASK_CARD)
    {
        // ...
    }

    int sdkResult = BS2_PartialUpdateUserFaceEx(context, id,
maskWantUpdate, &userBlob, 1);
    if (BS_SDK_SUCCESS != sdkResult)
    {
        TRACE("BS2_PartialUpdateUserFaceEx call failed: %d", sdkResult);
        return sdkResult;
    }
}
```

## (C#)

[sample\\_partialupdate.cs](#)

```
BS2_USER_MASK mask = (BS2_USER_MASK)BS2UserMaskEnum.SETTING |
(BS2_USER_MASK)BS2UserMaskEnum.SETTING_EX |
```

```
(BS2_USER_MASK)BS2UserMaskEnum.JOB;

BS2ErrorCode sdkResult = BS2ErrorCode.BS_SDK_SUCCESS;
BS2UserFaceExBlob[] userBlob =
Util.AllocateStructureArray<BS2UserFaceExBlob>(1);

userBlob[].cardObjs = IntPtr.Zero;
userBlob[].fingerObjs = IntPtr.Zero;
userBlob[].faceObjs = IntPtr.Zero;
userBlob[].user_photo_obj = IntPtr.Zero;
userBlob[].faceExObjs = IntPtr.Zero;

userBlob[].setting.fingerAuthMode = (byte)BS2FingerAuthModeEnum.NONE;
userBlob[].setting.cardAuthMode = (byte)BS2CardAuthModeEnum.NONE;
userBlob[].setting.idAuthMode = (byte)BS2IDAuthModeEnum.NONE;

userBlob[].settingEx.faceAuthMode = (byte)BS2ExtFaceAuthModeEnum.NONE;
userBlob[].settingEx.fingerprintAuthMode =
(byte)BS2ExtFingerprintAuthModeEnum.NONE;
userBlob[].settingEx.cardAuthMode = (byte)BS2ExtCardAuthModeEnum.NONE;
userBlob[].settingEx.idAuthMode = (byte)BS2ExtIDAuthModeEnum.NONE;

string userID;
if (BS2ErrorCode.BS_SDK_SUCCESS != (sdkResult = getUserBlobUserID(ref
userBlob[].user, out userID)))
    return;

if ((mask & (BS2_USER_MASK)BS2UserMaskEnum.SETTING) ==
(BS2_USER_MASK)BS2UserMaskEnum.SETTING)
{
    if (BS2ErrorCode.BS_SDK_SUCCESS != (sdkResult =
getUserBlobExpiryDate(ref userBlob[].setting)))
        return;

    if (BS2ErrorCode.BS_SDK_SUCCESS != (sdkResult =
getUserBlobPrivateAuthMode(ref userBlob[].setting)))
        return;

    if (BS2ErrorCode.BS_SDK_SUCCESS != (sdkResult =
getUserBlobSecurityLevel(ref userBlob[].setting)))
        return;
}

if ((mask & (BS2_USER_MASK)BS2UserMaskEnum.SETTING_EX) ==
(BS2_USER_MASK)BS2UserMaskEnum.SETTING_EX)
{
    if (BS2ErrorCode.BS_SDK_SUCCESS != (sdkResult =
getUserBlobPrivateAuthModeEx(ref userBlob[].settingEx)))
        return;
}
```

```
// ...

if ((mask & (BS2_USER_MASK)BS2UserMaskEnum.JOB) ==
    (BS2_USER_MASK)BS2UserMaskEnum.JOB)
{
    if (BS2ErrorCode.BS_SDK_SUCCESS != (sdkResult =
    getUserBlobJobCode(ref userBlob[].job)))
        return;

    userBlob[].user.infoMask |= (byte)BS2UserInfoMaskEnum.JOB_CODE;
}

// ...

userBlob[].user.numCards = ;
if ((mask & (BS2_USER_MASK)BS2UserMaskEnum.CARD) ==
    (BS2_USER_MASK)BS2UserMaskEnum.CARD)
{
    // ...
}

sdkResult = (BS2ErrorCode)API.BS2_PartialUpdateUserFaceEx(sdkContext,
deviceID, mask, userBlob, (UInt32)1);
if (BS2ErrorCode.BS_SDK_SUCCESS != sdkResult)
    Console.WriteLine("BS2_PartialUpdateUserFaceEx call failed {0}",
sdkResult);
```

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

Permanent link:  
[https://kb.supremainc.com/bs2sdk./doku.php?id=ko:bs2\\_partialupdateuserfaceex&rev=1661391734](https://kb.supremainc.com/bs2sdk./doku.php?id=ko:bs2_partialupdateuserfaceex&rev=1661391734)

Last update: **2022/08/25 10:42**