# 목차

# BS2_SetCustomCardConfig

[+ 2.9.4] Custom smart card 설정을 저장합니다.

> **주의 사항**
> Custom smart card 기능을 사용하려면, BS2SystemConfig의 useCardOperationMask에서, suprema smart card 모드들을 비활성화 시키고, custom smart card 모드들을 활성화 시켜야 합니다.

## 함수

```
#include "BS_API.h"

int BS2_SetCustomCardConfig(void* context, uint32_t deviceId, const
BS2CustomCardConfig* config);
```

> BS2CustomCardConfig 구조체 보기

## 파라미터

- [In] *context* : Context
- [In] *deviceId* : 장치 식별자
- [Out] *config* : Custom smart card 정보를 설정할 포인터

## 반환값

성공적으로 수행될 경우 BS_SDK_SUCCESS를 반환하고, 에러가 발생할 경우 상응하는 에러 코드를 반환합니다.

## 함께 보기

BS2_GetCustomCardConfig

## 샘플코드(C++)

sample_setcustomcardconfig.cpp

```
    ConfigControl cc(context);
    DeviceControl dc(context);
    BS2DeviceCapabilities capabilies = { , };
    BS2CustomCardConfig config = { , };

    BS2_DEVICE_ID id = Utility::getSelectedDeviceID(device);

    int sdkResult = dc.getDeviceCapabilities(id, capabilies);
    if (BS_SDK_SUCCESS != sdkResult)
        return sdkResult;

    if (!capabilies.customSmartCardSupported)
    {
        TRACE("Not supported function.");
        return BS_SDK_ERROR_NOT_SUPPORTED;
    }

    sdkResult = cc.getCustomCardConfig(id, config);
    if (BS_SDK_SUCCESS != sdkResult)
        return sdkResult;

    string msg = "Please enter a data type of cards. (0: Binary, 1:
ASCII, 2: UTF16, 3: BCS)";
    config.dataType =
(BS2_CARD_DATA_TYPE)Utility::getInput<uint32_t>(msg);
    config.useSecondaryKey = Utility::isYes("Do you want to use
secondary key?");

    ostringstream oss;

    if (Utility::isYes("Do you want to change mifare custom card
settings?"))
    {
        memset(&config.mifare.primaryKey, 0x0,
sizeof(config.mifare.primaryKey));
        oss << "Please enter the hexadecimal " <<
sizeof(config.mifare.primaryKey) << "-bytes primary key for mifare
card." << endl;
        oss << " [Like 12 34 56 ... EF]" << endl;
        Utility::getLineHexaString<uint8_t>(oss.str(),
config.mifare.primaryKey, sizeof(config.mifare.primaryKey));

        if (config.useSecondaryKey)
        {
            memset(&config.mifare.secondaryKey, 0x0,
sizeof(config.mifare.secondaryKey));
            oss.str("");
            oss << "Please enter the hexadecimal " <<
sizeof(config.mifare.secondaryKey) << "-bytes secondary key for mifare
```

```
card." << endl;
            oss << " [Like 12 34 56 ... EF]" << endl;
            Utility::getLineHexaString<uint8_t>(oss.str(),
config.mifare.secondaryKey, sizeof(config.mifare.secondaryKey));
        }

        msg = "Please enter the start block index of mifare card.";
        config.mifare.startBlockIndex =
(uint16_t)Utility::getInput<uint32_t>(msg);
        msg = "Please enter the card data size of mifare card.";
        config.mifare.dataSize =
(uint8_t)Utility::getInput<uint32_t>(msg);
        msg = "Please enter the skip bytes of mifare card.";
        config.mifare.skipBytes =
(uint8_t)Utility::getInput<uint32_t>(msg);
    }

    if (Utility::isYes("Do you want to change desfire custom card
settings?"))
    {
        msg = "Please enter a operation mode for desfire card. (0:
Legacy, 1: Advanced(AppLevelKey))";
        config.desfire.operationMode =
(uint8_t)Utility::getInput<uint32_t>(msg);

        if (DESFIRECARD_OPERATION_MODE_LEGACY ==
config.desfire.operationMode)
        {
            memset(&config.desfire.primaryKey, 0x0,
sizeof(config.desfire.primaryKey));
            oss.str("");
            oss << "Please enter the hexadecimal " <<
sizeof(config.desfire.primaryKey) << "-bytes primary key for desfire
card." << endl;
            oss << " [Like 12 34 56 ... EF]" << endl;
            Utility::getLineHexaString<uint8_t>(oss.str(),
config.desfire.primaryKey, sizeof(config.desfire.primaryKey));

            if (config.useSecondaryKey)
            {
                memset(&config.desfire.secondaryKey, 0x0,
sizeof(config.desfire.secondaryKey));
                oss.str("");
                oss << "Please enter the hexadecimal " <<
sizeof(config.desfire.secondaryKey) << "-bytes secondary key for
desfire card." << endl;
                oss << " [Like 12 34 56 ... EF]" << endl;
                Utility::getLineHexaString<uint8_t>(oss.str(),
config.desfire.secondaryKey, sizeof(config.desfire.secondaryKey));
            }
        }
```

```
        else    // DESFIRECARD_OPERATION_MODE_APPLEVELKEY
        {
            memset(&config.desfire.desfireAppKey.appMasterKey, 0x0,
sizeof(config.desfire.desfireAppKey.appMasterKey));
            memset(&config.desfire.desfireAppKey.fileReadKey, 0x0,
sizeof(config.desfire.desfireAppKey.fileReadKey));
            memset(&config.desfire.desfireAppKey.fileWriteKey, 0x0,
sizeof(config.desfire.desfireAppKey.fileWriteKey));

            oss.str("");
            oss << "Please enter the hexadecimal " <<
sizeof(config.desfire.desfireAppKey.appMasterKey) << "-bytes
appMasterKey for desfire card." << endl;
            oss << " [Like 12 34 56 ... EF]" << endl;
            Utility::getLineHexaString<uint8_t>(oss.str(),
config.desfire.desfireAppKey.appMasterKey,
sizeof(config.desfire.desfireAppKey.appMasterKey));

            oss.str("");
            oss << "Please enter the hexadecimal " <<
sizeof(config.desfire.desfireAppKey.fileReadKey) << "-bytes fileReadKey
for desfire card." << endl;
            oss << " [Like 12 34 56 ... EF]" << endl;
            Utility::getLineHexaString<uint8_t>(oss.str(),
config.desfire.desfireAppKey.fileReadKey,
sizeof(config.desfire.desfireAppKey.fileReadKey));

            msg = "Please enter the fileReadKeyNumber of desfire
card.";
            config.desfire.desfireAppKey.fileReadKeyNumber =
(uint8_t)Utility::getInput<uint32_t>(msg);
        }

        oss.str("");
        oss << "Please enter the hexadecimal " <<
sizeof(config.desfire.appID) << "-bytes appID for desfire card." <<
endl;
        oss << " [Like 12 34 EF]" << endl;
        Utility::getLineHexaString<uint8_t>(oss.str(),
config.desfire.appID, sizeof(config.desfire.appID));

        msg = "Please enter the fileID for desfire card.";
        config.desfire.fileID =
(uint8_t)Utility::getInput<uint32_t>(msg);
        msg = "Please enter a encryption type for desfire card. (0:
DES/3DES, 1: AES)";
        config.desfire.encryptionType =
(uint8_t)Utility::getInput<uint32_t>(msg);
        msg = "Please enter the card data size of desfire card.";
        config.desfire.dataSize =
(uint8_t)Utility::getInput<uint32_t>(msg);
```

```
        msg = "Please enter the skip bytes of desfire card.";
        config.desfire.skipBytes =
(uint8_t)Utility::getInput<uint32_t>(msg);
    }

    msg = "Please enter a smart card byte order. (0: MSB, 1: LSB)";
    config.smartCardByteOrder =
(BS2_CARD_BYTE_ORDER)Utility::getInput<uint32_t>(msg);
    msg = "Please enter a formatID.";
    config.formatID = (BS2_UID)Utility::getInput<uint32_t>(msg);

    sdkResult = cc.setCustomCardConfig(id, config);
    if (BS_SDK_SUCCESS != sdkResult)
        return sdkResult;

    oss.str("");
    oss << "To use the custom smart card function, you must turn off
the Suprema smart card function." << endl;
    oss << "Do you want to change the card operation mode?";
    if (Utility::isYes(oss.str()))
    {
        BS2SystemConfig sysConfig = { , };
        sdkResult = cc.getSystemConfig(id, sysConfig);
        if (BS_SDK_SUCCESS != sdkResult)
            return sdkResult;

        uint32_t preMask = sysConfig.useCardOperationMask;

        // Turn off Suprema smart card
        sysConfig.useCardOperationMask &=
~(uint32_t)CARD_OPERATION_MASK_CLASSIC_PLUS;
        sysConfig.useCardOperationMask &=
~(uint32_t)CARD_OPERATION_MASK_DESFIRE_EV1;
        sysConfig.useCardOperationMask &=
~(uint32_t)CARD_OPERATION_MASK_SR_SE;
        sysConfig.useCardOperationMask &=
~(uint32_t)CARD_OPERATION_MASK_SEOS;

        // Turn on Custom smart card
        sysConfig.useCardOperationMask |=
(uint32_t)CARD_OPERATION_MASK_CUSTOM_CLASSIC_PLUS;
        sysConfig.useCardOperationMask |=
(uint32_t)CARD_OPERATION_MASK_CUSTOM_DESFIRE_EV1;

        // Apply
        sysConfig.useCardOperationMask |= (uint32_t)CARD_OPERATION_USE;

        sdkResult = cc.setSystemConfig(id, sysConfig);
        if (BS_SDK_SUCCESS == sdkResult)
            TRACE("Card operation mode was changed 0x08d => 0x08d",
preMask, sysConfig.useCardOperationMask);
```

```
    }

    return sdkResult;
```

## 샘플코드(C#)

sample_setcustomcardconfig.cs

```
            BS2DeviceCapabilities capa;
            if (!CommonControl.getDeviceCapabilities(sdkContext,
deviceID, out capa))
                return;

            if (!Convert.ToBoolean(capa.functionExSupported &
(byte)BS2CapabilityFunctionExSupport.FUNCTIONEX_SUPPORT_CUSTOMSMARTCARD
))
            {
                Console.WriteLine("Not supported function.");
                return;
            }

            BS2CustomCardConfig config;
            Console.WriteLine("Try to get CustomCardConfig");
            BS2ErrorCode result =
(BS2ErrorCode)API.BS2_GetCustomCardConfig(sdkContext, deviceID, out
config);
            if (BS2ErrorCode.BS_SDK_SUCCESS != result)
                return;

            Util.HighlightLine("Please enter a data type of cards. (0:
Binary, 1: ASCII, 2: UTF16, 3: BCS)", "data type");
            Console.Write(">>>> ");
            config.dataType = Util.GetInput((byte));

            Util.HighlightLine("Do you want to use secondary key?",
"use secondary key");
            Console.Write(">>>> ");
            bool useSecondaryKey = Util.IsYes();
            config.useSecondaryKey = Convert.ToByte(useSecondaryKey);

            Util.HighlightLine("Do you want to change mifare custom
card settings? [Y/n]", "mifare custom card");
            Console.Write(">>>> ");
            if (Util.IsYes())
            {
                int sizeOfKey = config.mifare.primaryKey.Length;
                Array.Clear(config.mifare.primaryKey, , sizeOfKey);
                string tempStr = String.Format("Please enter the
```

```
hexadecimal {0}-bytes primary key for mifare card. [KEY1-KEY2-...-
KEY6]", sizeOfKey);
                Util.HighlightLineMulti(tempStr, "primary key", "mifare
card");
                Console.Write(">>>> ");
                enterSmartcardKey(config.mifare.primaryKey);

                if (useSecondaryKey)
                {
                    sizeOfKey = config.mifare.secondaryKey.Length;
                    Array.Clear(config.mifare.secondaryKey, ,
sizeOfKey);
                    tempStr = String.Format("Please enter the
hexadecimal {0}-bytes secondary key for mifare card. [KEY1-KEY2-...-
KEY6]", sizeOfKey);
                    Util.HighlightLineMulti(tempStr, "secondary key",
"mifare card");
                    Console.Write(">>>> ");
                    enterSmartcardKey(config.mifare.secondaryKey);
                }

                Util.HighlightLineMulti("Please enter the start block
index of mifare card.", "start block index", "mifare card");
                Console.Write(">>>> ");
                config.mifare.startBlockIndex =
Util.GetInput((UInt16));

                Util.HighlightLineMulti("Please enter the card data
size of mifare card.", "card data size", "mifare card");
                Console.Write(">>>> ");
                config.mifare.dataSize = Util.GetInput((byte));

                Util.HighlightLineMulti("Please enter the skip bytes of
mifare card.", "skip bytes", "mifare card");
                Console.Write(">>>> ");
                config.mifare.skipBytes = Util.GetInput((byte));
            }

            Util.HighlightLine("Do you want to change desfire custom
card settings? [Y/n]", "desfire custom card");
            Console.Write(">>>> ");
            if (Util.IsYes())
            {
                Util.HighlightLineMulti("Please enter a operationMode
of desfire card. (0: Legacy, 1: Advanced(AppLevelKey))",
"operationMode", "desfire card");
                Console.Write(">>>> ");
                config.desfire.operationMode = Util.GetInput((byte));

                string tempStr;
                int sizeOfKey;
```

```
                if (config.desfire.operationMode ==
Convert.ToByte(BS2DesfireCardOperation.BS2_DESFIRECARD_OPERATION_MODE_L
EGACY))
                {
                    sizeOfKey = config.desfire.primaryKey.Length;
                    Array.Clear(config.desfire.primaryKey, ,
sizeOfKey);
                    tempStr = String.Format("Please enter the
hexadecimal {0}-bytes primary key for desfire card. [KEY1-KEY2-...-
KEY6]", sizeOfKey);
                    Util.HighlightLineMulti(tempStr, "primary key",
"desfire card");
                    Console.Write(">>>> ");
                    enterSmartcardKey(config.desfire.primaryKey);

                    if (useSecondaryKey)
                    {
                        sizeOfKey = config.desfire.secondaryKey.Length;
                        Array.Clear(config.desfire.secondaryKey, ,
sizeOfKey);
                        tempStr = String.Format("Please enter the
hexadecimal {0}-bytes secondary key for desfire card. [KEY1-KEY2-...-
KEY6]", sizeOfKey);
                        Util.HighlightLineMulti(tempStr, "secondary
key", "desfire card");
                        Console.Write(">>>> ");
                        enterSmartcardKey(config.desfire.secondaryKey);
                    }
                }
                else  //
BS2DesfireCardOperation.BS2_DESFIRECARD_OPERATION_MODE_APPLEVELKEY
                {
                    int sizeOfAppMasterKey =
config.desfire.desfireAppKey.appMasterKey.Length;
                    int sizeOfFileReadKey =
config.desfire.desfireAppKey.fileReadKey.Length;
                    int sizeOfFileWriteKey =
config.desfire.desfireAppKey.fileWriteKey.Length;
Array.Clear(config.desfire.desfireAppKey.appMasterKey, ,
sizeOfAppMasterKey);
Array.Clear(config.desfire.desfireAppKey.fileReadKey, ,
sizeOfFileReadKey);
Array.Clear(config.desfire.desfireAppKey.fileWriteKey, ,
sizeOfFileWriteKey);

                    tempStr = String.Format("Please enter the
hexadecimal {0}-bytes appMasterKey for desfire card. [KEY1-KEY2-...-
KEY16]", sizeOfAppMasterKey);
                    Util.HighlightLineMulti(tempStr, "appMasterKey",
"desfire card");
                    Console.Write(">>>> ");
```

```
enterSmartcardKey(config.desfire.desfireAppKey.appMasterKey);

                    tempStr = String.Format("Please enter the
hexadecimal {0}-bytes fileReadKey for desfire card. [KEY1-KEY2-...-
KEY16]", sizeOfFileReadKey);
                    Util.HighlightLineMulti(tempStr, "fileReadKey",
"desfire card");
                    Console.Write(">>>> ");
enterSmartcardKey(config.desfire.desfireAppKey.fileReadKey);

                    Util.HighlightLineMulti("Please enter the
fileReadKeyNumber of desfire card.", "fileReadKeyNumber", "desfire
card");
                    Console.Write(">>>> ");
                    config.desfire.desfireAppKey.fileReadKeyNumber =
Util.GetInput((byte));
                }

                sizeOfKey = config.desfire.appID.Length;
                Array.Clear(config.desfire.appID, , sizeOfKey);
                tempStr = String.Format("Please enter the hexadecimal
{0}-bytes appID for desfire card. [KEY1-KEY2-...-KEY6]", sizeOfKey);
                Util.HighlightLineMulti(tempStr, "appID", "desfire
card");
                Console.Write(">>>> ");
                enterSmartcardKey(config.desfire.appID);

                Util.HighlightLineMulti("Please enter the fileID of
desfire card.", "fileID", "desfire card");
                Console.Write(">>>> ");
                config.desfire.fileID = Util.GetInput((byte));

                Util.HighlightLineMulti("Please enter a encryptionType
of desfire card. (0: DES/3DES, 1: AES)", "encryptionType", "desfire
card");
                Console.Write(">>>> ");
                config.desfire.encryptionType = Util.GetInput((byte));

                Util.HighlightLineMulti("Please enter the card data
size of desfire card.", "card data size", "desfire card");
                Console.Write(">>>> ");
                config.desfire.dataSize = Util.GetInput((byte));

                Util.HighlightLineMulti("Please enter the skip bytes of
desfire card.", "skip bytes", "desfire card");
                Console.Write(">>>> ");
                config.desfire.skipBytes = Util.GetInput((byte));
            }

            Util.HighlightLine("Please enter a smart card byte order.
(0: MSB, 1: LSB)", "smart card byte order");
```

```
            Console.Write(">>>> ");
            config.smartCardByteOrder = Util.GetInput((byte));

            Util.HighlightLine("Please enter a formatID.", "formatID");
            Console.Write(">>>> ");
            config.formatID = Util.GetInput((UInt32));

            Console.WriteLine("Trying to set CustomCardConfig.");
            result =
(BS2ErrorCode)API.BS2_SetCustomCardConfig(sdkContext, deviceID, ref
config);
            if (result != BS2ErrorCode.BS_SDK_SUCCESS)
                Console.WriteLine("Got error({0}).", result);
            else
                Console.WriteLine("Card config set success");

            Util.HighlightLineMulti("To use the custom smart card
function, you must turn off the Suprema smart card function. Do you
want to change the card operation mode? [Y/n]",
                "turn off the Suprema smart card function", "change the
card operation mode?");
            Console.Write(">>>> ");
            if (Util.IsYes())
            {
                BS2SystemConfig sysConfig;
                result =
(BS2ErrorCode)API.BS2_GetSystemConfig(sdkContext, deviceID, out
sysConfig);
                if (result != BS2ErrorCode.BS_SDK_SUCCESS)
                {
                    Console.WriteLine("Got error({0}).", result);
                }

                UInt32 preMask = sysConfig.useCardOperationMask;

                // Turn off Suprema smart card
                sysConfig.useCardOperationMask &=
~(UInt32)BS2SystemConfigCardOperationMask.CARD_OPERATION_MASK_CLASSIC_P
LUS;
                sysConfig.useCardOperationMask &=
~(UInt32)BS2SystemConfigCardOperationMask.CARD_OPERATION_MASK_DESFIRE_E
V1;
                sysConfig.useCardOperationMask &=
~(UInt32)BS2SystemConfigCardOperationMask.CARD_OPERATION_MASK_SR_SE;
                sysConfig.useCardOperationMask &=
~(UInt32)BS2SystemConfigCardOperationMask.CARD_OPERATION_MASK_SEOS;

                // Turn on Custom smart card
                sysConfig.useCardOperationMask |=
(UInt32)BS2SystemConfigCardOperationMask.CARD_OPERATION_MASK_CUSTOM_CLA
SSIC_PLUS;
```

```
                sysConfig.useCardOperationMask |=
(UInt32)BS2SystemConfigCardOperationMask.CARD_OPERATION_MASK_CUSTOM_DES
FIRE_EV1;

                // Apply
                sysConfig.useCardOperationMask |=
(UInt32)BS2SystemConfigCardOperationMask.CARD_OPERATION_USE;

                result =
(BS2ErrorCode)API.BS2_SetSystemConfig(sdkContext, deviceID, ref
sysConfig);
                if (result != BS2ErrorCode.BS_SDK_SUCCESS)
                    Console.WriteLine("Card operation mode update
failed ({0}).", result);
                else
                    Console.WriteLine("Card operation mode was changed
0x{0:x8} => 0x{1:x8}", preMask, sysConfig.useCardOperationMask);
            }
```