목차

BS2_WriteQRCode

[+ 2.8] X-Station 2 CSN 카드 구조체에 QR코드로 사용할 문자열을 기록합니다.
QR코드로 사용할 데이터는, ASCII코드 32(' ')에서 126('~')까지의 문자들로 구성되며,최대 길이는 32 자입니다.

함수

```
#include "BS_API.h"

int BS2_WriteQRCode(const char* qrText, BS2CSNCard* card);
```

> BS2CSNCard 구조체 보기

파라미터

- [In] *qrText* : QR코드로 사용할 문자열
- [Out] *card* : QR코드가 적용된 CSN 카드 구조체 포인터

반환값

성공적으로 수행될 경우 BS_SDK_SUCCESS를 반환하고, 적절하지 않는 문자열이 전달되는 경우 BS_SDK_ERROR_INVALID_PARAM 오류 코드를 반환합니다.

샘플코드

C++

```
if (qrSupported)
    {
        if (Utility::isYes("Would you like to register the QR code string to be used for authentication?"))
        {
            stringstream msg;
            msg << "Enter the ASCII QR code." << endl;
            msg << "  [ASCII code consisting of values between 32 and 126].";
            string qrCode = Utility::getInput<string>(msg.str());
```

```
            BS2CSNCard qrCard = { , };
            sdkResult = BS2_WriteQRCode(qrCode.c_str(), &qrCard);
            if (BS_SDK_SUCCESS != sdkResult)
            {
                TRACE("BS2_WriteQRCode call failed: %d", sdkResult);
            }
            else
            {
                size_t numOfRealloc = numOfCards + 1;
                BS2CSNCard* ptrNewCard = new BS2CSNCard[numOfRealloc];
                memset(ptrNewCard, 0x0, sizeof(BS2CSNCard) * numOfRealloc);

                if ( < numOfCards && *cardObjs)
                {
                    memcpy(ptrNewCard, *cardObjs, sizeof(BS2CSNCard) *
numOfCards);
                    delete[] * cardObjs;
                    *cardObjs = NULL;
                }

                memcpy(ptrNewCard + numOfCards, &qrCard,
sizeof(BS2CSNCard));
                *cardObjs = ptrNewCard;
                numOfCards++;
            }
        }
    }

BS2_ReleaseObject(uidObj);
```

C#

```
 if (qrSupported)
            {
                Console.WriteLine("Would you like to register the QR code
string to be used for authentication? [y/n]");
                Console.Write(">>>> ");
                if (Util.IsYes())
                {
                    Console.WriteLine("Enter the ASCII QR code.");
                    Console.WriteLine("  [ASCII code consisting of values
between 32 and 126].");
                    Console.Write(">>>> ");
                    string qrCode = Console.ReadLine();

                    IntPtr qrCodePtr = Marshal.StringToHGlobalAnsi(qrCode);
                    BS2CSNCard qrCard =
Util.AllocateStructure<BS2CSNCard>();
                    result = (BS2ErrorCode)API.BS2_WriteQRCode(qrCodePtr,
ref qrCard);
                    if (BS2ErrorCode.BS_SDK_SUCCESS != result)
```

```
                    {
                        Console.WriteLine("Got error({0}).", result);
                    }
                    else
                    {
                        int numOfRealloc = userBlob.user.numCards + 1;
                        int structSize = Marshal.SizeOf(typeof(BS2CSNCard));
                        byte[] tempCard = new byte[structSize *
userBlob.user.numCards];

                        if ( < userBlob.user.numCards && IntPtr.Zero !=
userBlob.cardObjs)
                        {
                            Marshal.Copy(userBlob.cardObjs, tempCard, ,
structSize * userBlob.user.numCards);
                            Marshal.FreeHGlobal(userBlob.cardObjs);
                        }

                        userBlob.cardObjs = Marshal.AllocHGlobal(structSize
* numOfRealloc);
                        if ( < userBlob.user.numCards)
                        {
                            Marshal.Copy(tempCard, , userBlob.cardObjs,
structSize * userBlob.user.numCards);
                        }

                        IntPtr curCardObjs = userBlob.cardObjs + structSize
* userBlob.user.numCards;

                        byte[] qrArray = Util.StructToBytes<BS2CSNCard>(ref
qrCard);
                        Marshal.Copy(qrArray, , curCardObjs, structSize);
                        userBlob.user.numCards++;

                        Marshal.FreeHGlobal(qrCodePtr);
                    }
                }
            }
Marshal.FreeHGlobal(authGroupIDObj);
BS2_ReleaseObject(uidObj);
```