SAP Business One
How-To Guide

PUBLIC

Barcode DLL Development Guide

Applicable Releases:

SAP Business One 2005 B

SAP Business One 2007 B

SAP Business One 8.8

Brazil

English

July 2009

Table of Contents

Introduction

This document describes how to create a dynamic link library (DLL) for calculating Brazilian barcode numbers. One Boleto algorithm can be compiled as a DLL to calculate the barcode number correlated with the algorithm.

The target audience of this document is developers with basic C++ development experience.

If you are not familiar with the basic concepts of DLL, familiarize yourself with the basic concepts before you start to develop a DLL.

Hardware and Software Requirements

SAP Software/Libraries

To develop a DLL for the barcode calculation, you need the following software components provided by SAP:

- SAP Business One 2005 B PL25 or higher (we recommend that you use the latest version)
- Business Object Headers (C++)

Third-Party Software

To develop a DLL for the barcode algorithm, you need one of the following software components provided by a third party:

- Microsoft Visual C++ Version 6.0
- Microsoft Visual Studio 2005

Hardware

As a minimum, we recommend that your system meet the following requirements:

- Intel Pentium III or equivalent processor
- 256 MB RAM
- 1 GB free hard disk space (as well as another 1 GB for Visual Studio)

Note

For more specific hardware requirements, see the documentation of SAP Business One and Microsoft Visual C++.

“Barcode DLL” Interface Specification

The header file `Boleto_BarcodeInterface.h`, which contains Boleto barcode interface and general definitions, must be included into your project. You can use the three callback procedures defined in `PBarcodeFuncs` to develop your own barcode algorithm. Your DLL must declare and implement the `DllGetAlgorithmName` and `DllGetBarcodeNumber` methods, which will be called by the SAP Business One.

The following code lines show the whole picture of `Boleto_BarcodeInterface.h`:

```
#ifndef BOLETO_BARCODEINTERFACE_H
   #define BOLETO_BARCODEINTERFACE_H

////////////////////////////////////////////////////////////////////////////
//General Settings
typedef long ErrCode;
typedef unsigned long BarcodeHandler;


#define ALGORITHM_LENGTH                50 // Length of the algorithm name.
#define BOLETO_BARCODE_LEN              101
#define BOLETO_OURNUMCHECKDIGIT_LEN     2

////////////////////////////////////////////////////////////////////////////
//Boleto Barcode accessable Tables
#define Boleto_OBOE     _T("OBOE")//Bill of Exchange
#define Boleto_DSC1     _T("DSC1")//House Bank Definition
#define Boleto_OPYM     _T("OPYM")//Payment Method
#define Boleto_OPTF     _T("OPTF")//Portfolio Definition

////////////////////////////////////////////////////////////////////////////
//Error Message Definition
#define noErr       0
#define errNoMsg        -10
#define barcodeDataInvalid    -11
#define algorithmNameInvalid  -12
#define bankCodeInvalid    -20
#define branchCodeInvalid     -21
#define ourNumberInvalid      -22
#define agreementNumberInvalid    -23
#define accountNumberInvalid      -24
#define accountCheckNumberInvalid -25
#define portfolioNumberInvalid    -26
#define currencyCodeInvalid       -27
#define dueDateInvalid     -28
#define amountInvalid      -29
#define createDateInvalid     -30

#define barcodeNumberInvalid  -31

////////////////////////////////////////////////////////////////////////////
//Boleto Barcode Callback Procs Struct
```

```
typedef struct _BarcodeFuncs
{
   ErrCode (*GetFieldValue)(BarcodeHandler handle, const TCHAR *tableName, const
TCHAR *fieldName, TCHAR *value);
   ErrCode (*GetFieldLength) (BarcodeHandler handle, const TCHAR *tableName, const
TCHAR *fieldName, long *pLength);
   ErrCode (*SetBarcodeNumber)(BarcodeHandler handle, const TCHAR *barcodeNumber,
const TCHAR *ourNumberCheckDigit);

}BarcodeFuncs,*PBarcodeFuncs;

#endif //BOLETO_BARCODEINTERFACE_H
```

Boleto Barcode Accessible Tables

To construct your barcode algorithm, you can access fields only in the following four tables:

```
#define Boleto_OBOE     _T("OBOE")//Bill of Exchange
#define Boleto_DSC1     _T("DSC1")//House Bank Definition
#define Boleto_OPYM     _T("OPYM")//Payment Method
#define Boleto_OPTF     _T("OPTF")//Portfolio Definition
```

To display concrete column information in these four tables, you can use the SAP Business One *Query Generator:*

1. To access the *Query Generator* from the SAP Business One *Main Menu*, choose *Tools* → *Queries* → *Query Generator*.
2. In the *Query Generator* window, enter the table name you want to access and press TAB.

   For example, input **OBOE**, and then press TAB.

In the middle area, you can get all the column names and descriptions of the table.

BarcodeFuncs Interface

`BarcodeFuncs` is a structure and consists of the following callback procedures implemented in SAP Business One. Use these procedures to construct your Boleto algorithm:

Callback Procedure	Description
`GetFieldValue`	Enables you to obtain any field value used for barcode calculation
`GetFieldLength`	Enables you to determine the string length of the field, so that you can allocate appropriate space for the field
`SetBarcodeNumber`	Enables you to return the value of *Barcode Number* and *Our Number Check Digit* back to SAP Business One

```
typedef struct _BarcodeFuncs
{
   ErrCode (*GetFieldValue)(BarcodeHandler handle, const TCHAR *tableName, const
TCHAR *fieldName, TCHAR *value);
   ErrCode (*GetFieldLength) (BarcodeHandler handle, const TCHAR *tableName, const
TCHAR *fieldName, long *pLength);
```

```
   ErrCode (*SetBarcodeNumber)(BarcodeHandler handle, const TCHAR *barcodeNumber,
const TCHAR *ourNumberCheckDigit);

}BarcodeFuncs,*PBarcodeFuncs;

#endif //BOLETO_BARCODEINTERFACE_H
```

BarcodeFuncs		
Callback Function	**Parameter**	**Description**
`GetFieldValue`	*handle*	Provided by SAP Business One to access an internal system object
	tableName	Four Boleto barcode accessible tables defined as Macro in the interface header file (`Boleto_BarcodeInterface.h`)
	fieldName	Column name in one of the accessible tables You can display exact table structure and column names with the SAP Business One *Query Generator*.
	value	Output parameter used to retrieve the field value
`GetFieldLength`	*handle*	Provided by SAP Business One to access internal system objects
	tableName	Four Boleto barcode accessible tables defined as Macro in the interface header file.
	fieldName	Column name in one of the accessible tables You can display exact table structure and column names with the SAP Business One *Query Generator*.
	pLength	Output parameter used to retrieve the field length
`SetBarcodeNumber`	*handle*	Provided by SAP Business One to access an internal system object
	barcodeNumber	With this parameter, you can send the barcode number generated in your DLL back to SAP Business One.
	ourNumberCheckDigit	With this parameter, you can send *Our Number Check Digit* generated in your DLL back to SAP Business One.

ErrCode Interface

ErrCode is a collection of macro definitions. When barcode calculation cannot be processed, the DLL returns the errCode to SAP Business One, which then displays a warning message.

```
//Error Message Definition
#define noErr       0
#define errNoMsg        -10
```

```
#define barcodeDataInvalid    -11
#define algorithmNameInvalid  -12
#define bankCodeInvalid    -20
#define branchCodeInvalid     -21
#define ourNumberInvalid      -22
#define agreementNumberInvalid    -23
#define accountNumberInvalid      -24
#define accountCheckNumberInvalid -25
#define portfolioNumberInvalid    -26
#define currencyCodeInvalid       -27
#define dueDateInvalid     -28
#define amountInvalid      -29
#define createDateInvalid     -30

#define barcodeNumberInvalid  -31
```

ErrCode		
Parameter	**Value**	**Description/Message**
noErr	0	No error if successful
errNoMsg	-10	The DLL algorithm returns an error
bankCodeInvalid	-20	The house bank code is not numeric or is invalid
branchCodeInvalid	-21	The house bank branch is not numeric or is invalid
ourNumberInvalid	-22	The our number in house bank is not numeric or is invalid
agreementNumberInvalid	-23	The agreement number in house bank is not numeric or is invalid
accountNumberInvalid	-24	The account in house bank is not numeric or is invalid
accountCheckNumberInvalid	-25	The account check digit in house bank is not numeric or is invalid
portfolioNumberInvalid	-26	The portfolio number in house bank is not numeric or is invalid
currencyCodeInvalid	-27	The currency code in house bank is not numeric or is invalid
dueDateInvalid	-28	Bill of exchange due date is invalid
amountInvalid	-29	Bill of exchange amount is invalid or run over
createDateInvalid	-30	Bill of exchange create date is invalid

DllGetAlgorithmName Interface

SAP Business One calls this method when uploading the DLL and uses an empty array to retrieve the algorithm name.

Note

The maximum length of the algorithm name is 50 characters.

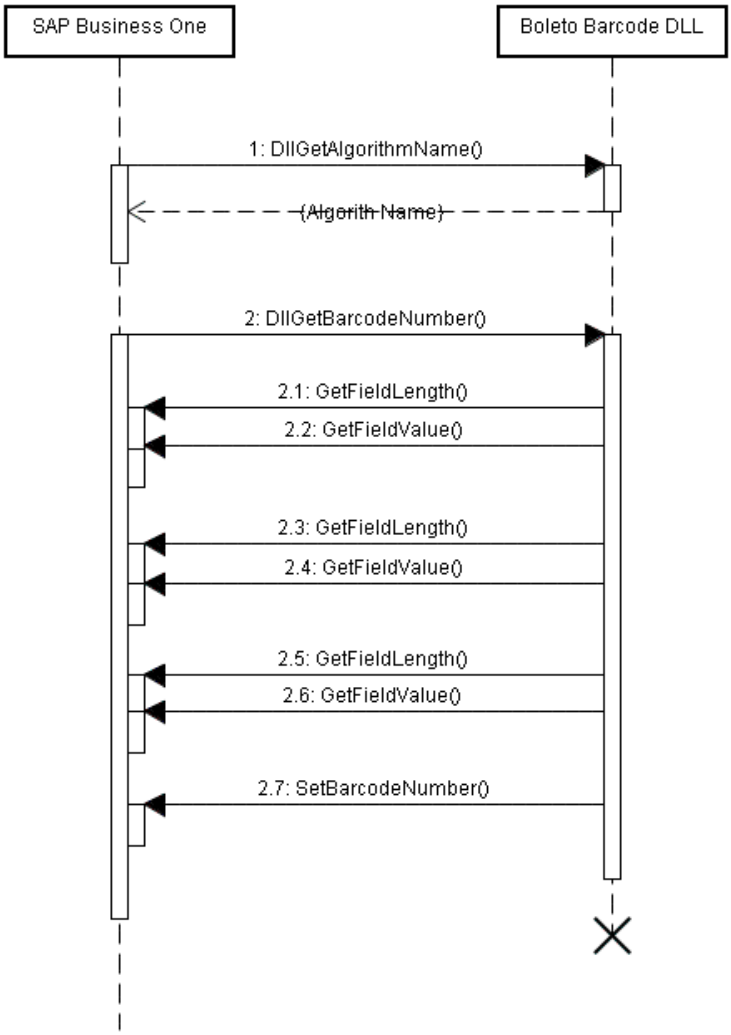
DllGetAlgorithmName		
Parameter	**Type**	**<Header>**
Return Value	`ErrCode`	noErr if successful, or certain errCode defined
pAlgorithmName	`TCHAR *`	Pointer to an array which will contain the barcode algorithm name, DLL should copy a 0-terminated string to this array
length	`long`	Length of the pAlgorithmName array, a reference value to DL

DllGetBarcodeNumber Interface

SAP Business One calls this method to receive the value of the barcode number and the `Our Number Check Digit`. You should implement your Boleto barcode algorithm in this function.

DllGetBarcodeNumber		
Parameter	**Type**	**Description**
Return Value	`ErrCode`	noErr if successful, or certain errCode defined
pFuncs	`PBarcodeFuncs`	A structure consists of three callback procedures, which are implemented by SAP Business One. These are used to construct your barcode algorithm.
handle	`BarcodeHandler`	Used by the DLL to access objects in SAP Business One

The following diagram shows the sequence that takes place when generating a barcode number. With this diagram, you can understand the main idea of how SAP Business One and your barcode DLL interact with each other and how to use the three callback procedures in the `BarcodeFuncs` structure to construct your barcode Algorithm.



Creating a DLL with Microsoft Visual C++ 6.0

The following steps describe how to create a DLL using Microsoft Visual C++ 6.0.

1. Creating a Project

1. Run Microsoft Visual C++.
2. On the *Start* page, choose *New Project*.

   Alternatively, you can choose from the menu *File* → *New* → *Project* or press **`Ctrl+N`**.
3. On the *Project* tab, choose *Win32 Dynamic-Link Library*.
   a. Enter the project name.

    Note

   The maximum length of DLL name is 45(char).

   b. Specify the working folder.
   c. Press *OK* to go the next window.
4. Choose the kind of DLL you want to create and click *Finish*.

   For example, you could choose *A simple DLL project*.

2. Adding an Additional Header File into the Project

1. Copy the header file `Boleto_BarcodeInterface.h` into your project folder.
2. Under your project workspace, right-click the folder *Header Files* and choose *Add Files to Folder…*.
3. In the *Insert Files into Project* window, choose the header file *Boleto_BarcodeInterface.h*, and click *OK*.

3. Creating the C++ Header File

1. Declare the methods to export to the DLL:

   ```
   ErrCode DllGetBarcodeNumber(PBarcodeFuncs pFuncs, BarcodeHandler
   &handle);
   ```

   ```
   ErrCode DllGetAlgorithmName(TCHAR *pAlgorithmName, long length);
   ```
2. Include the header file `Boleto_BarcodeInterface.h`.

   ```
   #include "Boleto_BarcodeInterface.h"
   ```

    Note

   For detailed information, see the sample code attached in the section *Implementation Examples*.
3. Declare Macro and internal functions used in your algorithm.

4. Creating a DEF File

Create a def file, for example, `BankOfBrazil.def`, and add it to the DLL project:

 Example

```
LIBRARY  BankOfBrazil
EXPORTS
    DllGetBarcodeNumber @1
    DllGetAlgorithmName @2
```

5. Implementing the C++ File

For more information, see the sample code attached in the section *Implementation Examples*.

Creating a DLL with Microsoft Visual Studio.Net 2005

The following steps describe how to create a DLL using Microsoft Visual Studio.Net 2005

1. Creating a Project

1. Run Microsoft Visual Studio 2005.
2. On the *Start* page, choose *New Project*.

   Alternatively, you can choose from the menu *File* → *New* → *Project* or press **`Ctrl+Shift+N`**.
3. On the *New Project* window, choose the template *Win32 Project*.
   a. Enter the project name.

    Note

   The maximum length of the DLL name is 45 characters.

   b. Specify the location where you want store your project.
   c. Choose *OK*.
4. Follow the steps in the *Win32 Application Wizard*:
   a. In the *Welcome* window, choose *Next* to continue.
   b. In the *Application Settings* window, make the following settings:

      Application type: *DLL*

      Additional options: *Export symbols*
   c. Choose *Finish* to create the DLL project.

2. Adding an Additional Header File into the Project

1. Copy the header file `Boleto_BarcodeInterface.h` into your project folder.
2. Under *Solution Explorer*, right-click the folder *Header Files* and choose *Add* → *Existing Item…*.
3. In the *Add Existing Item* window, choose the header file *Boleto_BarcodeInterface.h* and click *Add*.

3. Modifying the C++ Header File

1. Include the header file `Boleto_BarcodeInterface.h`.

   ```
   #include "Boleto_BarcodeInterface.h"
   ```

    Note

   For detailed information, see the sample code attached in the section *Implementation Examples*.
2. Declare the methods whose function is to export the DLL file:

   ```
   ErrCode DllGetBarcodeNumber(PBarcodeFuncs pFuncs, BarcodeHandler
   &handle);

   ErrCode DllGetAlgorithmName(TCHAR *pAlgorithmName, long length);
   ```

3. Declare the macro and internal functions that will be used in your algorithm.

4. Creating a DEF File

Create a def file (for example, `BankOfBrazil.def`) and add it to the DLL project:

 Example

```
LIBRARY  BankOfBrazil
EXPORTS
    DllGetBarcodeNumber @1
    DllGetAlgorithmName @2
```

5. Implementing the C++ File

For more information, see the sample code attached in the section *Implementation Examples*.

Implementation Examples

To see what a project implementation might look like, refer to the sample code file `BarcodeAlgorithm_Sample.zip` generated by SAP Business One.

Sample: C++ Header File

```
#ifndef BANKOFBRAZIL_H
#define BANKOFBRAZIL_H

////////////////////////////////////////////////////////////////////////////
#include "Boleto_BarcodeInterface.h"
#include "stdafx.h"
using namespace std;


// The following ifdef block is the standard way of creating macros which make
exporting
// from a DLL simpler. All files within this DLL are compiled with the
BANKOFBRAZIL_EXPORTS
// symbol defined on the command line. this symbol should not be defined on any
project
// that uses this DLL. This way any other project whose source files include this
file see
// BANKOFBRAZIL_API functions as being imported from a DLL, wheras this DLL sees
symbols
// defined with this macro as being exported.
#ifdef BANKOFBRAZIL_EXPORTS
#define BANKOFBRAZIL_API __declspec(dllexport)
#else
#define BANKOFBRAZIL_API __declspec(dllimport)
#endif

// Generic STL string class.
#ifdef _UNICODE
#define tstring wstring
#else
#define tstring string
#endif

//General Macro Define
#define START_DATE                  19971007 //Start date for calculating due date,
1997-Jul-07.
#define ArraySize(x)             (sizeof(x)/sizeof(x[0]))

////////////////////////////////////////////////////////////////////////////



//for every bank; one bank's Algorithm is compiled to one dll
extern "C" BANKOFBRAZIL_API ErrCode DllGetBarcodeNumber(PBarcodeFuncs pFuncs,
BarcodeHandler handle);

//get the Algorithm's Name displayed at drapdown list
```

```
//Parameter:
//  TCHAR pAlgorimName[51]:  out; the discription of algorithm
extern "C" BANKOFBRAZIL_API ErrCode DllGetAlgorithmName(TCHAR *pAlgorithmName, long
length);

////////////////////////////////////////////////////////////////////////////////
////
//define the length of fields used to construct Boleto Barcode
#define BOLETO_BANKOFBRAZIL_BANKCODE_LEN    4
#define BOLETO_BANKOFBRAZIL_CURRENCYCODE_LEN      2
#define BOLETO_BANKOFBRAZIL_DATEANDAMOUNT_LEN     15
#define BOLETO_BANKOFBRAZIL_OURNUMBER_LEN      12
#define BOLETO_BANKOFBRAZIL_BRANCHCODE_LEN     5
#define BOLETO_BANKOFBRAZIL_ACCOUNTNUMBER_LEN     9
#define BOLETO_BANKOFBRAZIL_PORTFOLIONUM_LEN      3
#define BOLETO_BANKOFBRAZIL_DUEDATE_LEN        5
#define BOLETO_BANKOFBRAZIL_AMOUNT_LEN         15
#define BOLETO_BANKOFBRAZIL_AGREENUM_LEN    5

// Declaration of internal functions.
namespace boleto_bankofbrazil
{
    // Declaration of internal functions.
    bool ValidateNumber(tstring str);
    void LeftPadding(TCHAR str[], size_t arraySize, size_t destLength);
    short days_of_year(short year, short mon, short day);
    long  days_num(long startrq,long endrq);
    int Modulus11(const TCHAR *origin,  const TCHAR *factor);
    ErrCode GetBankCode(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR
*bankCode);
    ErrCode GetCurrencyCode(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR
*currencyCode);
    ErrCode GetDueDate(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR *dueDate);
    ErrCode GetAmount(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR *amount);
    ErrCode GetDateAndAmount(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR
*dateAndAmount);
    ErrCode GetAgreeNumber(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR
*agreeNumber);
    ErrCode GetOurNumber(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR
*ourNumber);
    ErrCode GetBranchCode(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR
*branchCode);
    ErrCode GetAccountNumber(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR
*accountNumber);
    ErrCode GetPortfolioNumber(PBarcodeFuncs pFuncs, BarcodeHandler handle, TCHAR
*portfolioNumber);
    ErrCode InsertCheckDigit(TCHAR *barcodeNumber);
    ErrCode GetOurNumberCheckDigit(TCHAR *ourNum, TCHAR *ourNumberCheckDigit);
}

#endif // BANKOFBRAZIL_H
```

Sample: DllGetAlgorithmName

```
// Exported function which gets the Algorithm's Name displayed at dropdown list.
// Argument:
//     TCHAR *pAlgorimName:  in/out; the discription of algorithm.
//     long length: in; the length of array pAlgorithmName.
extern "C" BANKOFBRAZIL_API ErrCode DllGetAlgorithmName(TCHAR *pAlgorithmName,long
length)
{
   // Validate whether the passing-in pointer is NULL.
   if(!pAlgorithmName)
      return algorithmNameInvalid;

      // Bank name, no more than 50 characters.
   const TCHAR *algorithmName=_T("Banco do Brasil - Boleto");

   // The algorithm name should not be more than 50 characters.
   if(_tcslen(algorithmName)>=length)
      return algorithmNameInvalid;

   _tcscpy(pAlgorithmName,algorithmName);

   return noErr;
}
```

Sample: DllGetBarcodeNumber

```
// Exported function which gets the barcode number
// Argument:
//     PBarcodeFuncs:      structure consists of Callback Proc
//     BarcodeHandler:     handle used to access object in SAP Business one
extern "C" BANKOFBRAZIL_API ErrCode DllGetBarcodeNumber(PBarcodeFuncs pFuncs,
BarcodeHandler handle)
{
   using namespace boleto_bankofbrazil;

   ErrCode errCode=noErr;

   TCHAR barcodeNumber[BOLETO_BARCODE_LEN] = {0};
   TCHAR ourNumberCheckDigit[BOLETO_OURNUMCHECKDIGIT_LEN] = {0};

   // Definition for the barcode fields.
   // The size of TCHAR array is always 1 bigger than definition in order to store
//terminal NULL.
   TCHAR bankCode[BOLETO_BANKOFBRAZIL_BANKCODE_LEN] = {0};
   TCHAR currencyCode[BOLETO_BANKOFBRAZIL_CURRENCYCODE_LEN] = {0};
   TCHAR dateAndAmount[BOLETO_BANKOFBRAZIL_DATEANDAMOUNT_LEN] = {0};
   TCHAR ourNumber[BOLETO_BANKOFBRAZIL_OURNUMBER_LEN] = {0};
   TCHAR branchCode[BOLETO_BANKOFBRAZIL_BRANCHCODE_LEN] = {0};
   TCHAR accountNumber[BOLETO_BANKOFBRAZIL_ACCOUNTNUMBER_LEN] = {0};
   TCHAR portfolioNumber[BOLETO_BANKOFBRAZIL_PORTFOLIONUM_LEN] = {0};

   errCode=GetBankCode(pFuncs, handle, bankCode);
```

```
    if(errCode!=noErr)
        return errCode;
    _tcscpy(barcodeNumber,bankCode);

    errCode=GetCurrencyCode(pFuncs, handle, currencyCode);
    if(errCode!=noErr)
        return errCode;
    _tcscat(barcodeNumber,currencyCode);

    errCode=GetDateAndAmount(pFuncs, handle, dateAndAmount);
    if(errCode!=noErr)
        return errCode;
    _tcscat(barcodeNumber,dateAndAmount);

    errCode=GetOurNumber(pFuncs, handle, ourNumber);
    if(errCode!=noErr)
        return errCode;
    _tcscat(barcodeNumber,ourNumber);

    errCode=GetBranchCode(pFuncs, handle, branchCode);
    if(errCode!=noErr)
        return errCode;
    _tcscat(barcodeNumber,branchCode);

    errCode=GetAccountNumber(pFuncs, handle, accountNumber);
    if(errCode!=noErr)
        return errCode;
    _tcscat(barcodeNumber,accountNumber);

    errCode=GetPortfolioNumber(pFuncs, handle, portfolioNumber);
    if(errCode!=noErr)
        return errCode;
    _tcscat(barcodeNumber,portfolioNumber);

    errCode=InsertCheckDigit(barcodeNumber);
    if(errCode!=noErr)
        return errCode;

    errCode=GetOurNumberCheckDigit(ourNumber, ourNumberCheckDigit);
    if(errCode!=noErr)
        return errCode;

    //pass the Barcode Number and Our Number Check Digit back to SAP Business One
    pFuncs->SetBarcodeNumber(handle, barcodeNumber, ourNumberCheckDigit);

    return errCode;
}
```

Authorizations

For information about the authorizations required for developing a DLL, see the online help as well as the document *How to Define Authorizations,* which you can download from the documentation area of SAP Business One Customer Portal at http://service.sap.com/smb/sbocustomer/documentation.

Copyrights, Trademarks, and Disclaimers

The current version of the copyrights, trademarks, and disclaimers at http://service.sap.com/smb/sbocustomer/documentation is valid for this document.