

PRODUCT KEY FINDER DLL DOCUMENTATION

Author

Dave Hope (dave@davehope.co.uk)

Document History

28.07.2009 v1.0 Initial document created

TABLE OF CONTENTS

PURPOSE	1
LICENSING	1
Obtaining consent	1
EXPORTED METHODS	2
version()	2
Example	2
getDetectionMethodsCount()	3
Example	3
getProductKey(int detectionMethod, char* remoteHost)	4
Params	4
Example	4
FREQUENTLY ASKED QUESTIONS	6
How do I use your DLL from my .Net (c# / vb) application?	6
Program x isn't detected. Can you add support for it?	6
OBTAINING SUPPORT	6

PURPOSE

The DLL is intended to provide the functionality in the [Product Key Finder application](#) to developers, allowing them to implement the functionality into existing applications without having to reproduce someone else's work.

License detection functionality is likely to be useful in network reporting and inventory software.

LICENSING

The ProductKeyFinder DLL can be used in any free (gratis) program at zero cost. If you wish to use this software in commercial applications (shareware or otherwise) you may do so with prior written consent.

Any free program using the DLL must give credit in about dialogs and documentation. This should include my name and a link to <http://davehope.co.uk>

OBTAINING CONSENT

To obtain consent to use the ProductKeyFinder DLL in your commercial application simply e-mail dave@davehope.co.uk with a description of the application and the expected audience. I aim to respond within 48 working hours, usually much sooner.

EXPORTED METHODS

Methods in the DLL are exported using `__declspec(dllexport)`. The exported methods are as follows:

```
char* version( );
int getDetectionMethodsCount( );
std::vector< std::string > getProductKey( int detectionMethod, char*
remoteHost );
```

These methods should be sufficient to provide a good level of functionality.

VERSION()

This method returns a character array detailing the version of the DLL in use. This returns the full version number in the following format:

```
2.0.9.1000
```

The versioning scheme is:

```
Major . Minor . Maintenance . Build
```

Usually functionality will not change between maintenance builds, however will do between minor and major builds. The build number is of no significance is only used internally.

EXAMPLE

```
#include <iostream>
#include <windows.h> /* Required for LoadLibrary etc */

using namespace std;
typedef char* (*version)();

int main()
{
    version _version;

    HINSTANCE hInst = LoadLibraryW( L"ProductKeyFinder.dll" );
    if( hInst )
    {
        _version = (version)GetProcAddress( hInst, "version" );

        if( _version )
        {
            cout << "Version: " << _version() << ::endl;
        }
        FreeLibrary( hInst ); ///< Free up DLL.
    }
    else
    {
        cout << "Fail" << ::endl;
    }
    return ERROR_SUCCESS;
}
```

GETDETECTIONMETHODSCOUNT()

This method returns a 32bit integer with the number of detection methods currently available.

EXAMPLE

```
#include <iostream>
#include <windows.h> /* Required for LoadLibrary etc */

using namespace std;
typedef int (*getDetectionMethodsCount)();

int main()
{
    getDetectionMethodsCount _getDetectionMethodsCount;

    HINSTANCE hInst = LoadLibraryW( L"ProductKeyFinder.dll" );
    if( hInst )
    {
        _getDetectionMethodsCount =
(getDetectionMethodsCount)GetProcAddress( hInst, "getDetectionMethodsCount" );
        if( _getDetectionMethodsCount )
        {
            cout << "Detection Methods: " << _getDetectionMethodsCount()
<< ::endl;
        }
        FreeLibrary( hInst ); ///< Free up DLL.
    }
    else
    {
        cout << "Fail" << ::endl;
    }
    return ERROR_SUCCESS;
}
```

GETPRODUCTKEY(INT DETECTIONMETHOD, CHAR* REMOTEHOST)

This method returns a std::vector of strings, the indexes are as follows:

0	Product Name
1	License Number

For example:

0	Microsoft Windows
1	XXXXX-XXXXX-XXXXX-XXXXX-XXXXX

If a license number cannot be found, the 1st index is NULL.

PARAMS

getProductKey() has two parameters:

int detectionMethod	The detection method to run. See getDetectionMethodsCount().
char* remoteHost	The host to check, use NULL to check the local system.

Both parameters are required.

EXAMPLE

```
#include <iostream>
#include <windows.h> /* Required for LoadLibrary etc */
#include <vector>    /* Required for std::vector */
#include <string>   /* Required for std::string */

using namespace std;

typedef int (*getDetectionMethodsCount)();
typedef std::vector< std::string > (*getProductKey)(int, char);

int main()
{
    std::vector< std::string > keyInfo;
    getDetectionMethodsCount _getDetectionMethodsCount;
    getProductKey _getProductKey;

    HINSTANCE hInst = LoadLibraryW( L"ProductKeyFinder.dll" );
    if( hInst )
    {
        _getDetectionMethodsCount =
(getDetectionMethodsCount)GetProcAddress( hInst, "getDetectionMethodsCount" );
        _getProductKey = (getProductKey)GetProcAddress( hInst,
"getProductKey" );

        for (int n =0; n<_getDetectionMethodsCount(); n++)
        {
            keyInfo = _getProductKey(n, NULL);

            if ( keyInfo[1].length() > 0 )
            {
                cout << keyInfo[0] << " " << keyInfo[1] <<
::endl;
            }
        }
        FreeLibrary( hInst ); ///< Free up DLL.
    }
}
```

```
else
{
    cout << "Fail" << ::endl;
}

::cin.get(); ///< Wait for user to hit return key before exiting.
return 0;
}
```

FREQUENTLY ASKED QUESTIONS

HOW DO I USE YOUR DLL FROM MY .NET (C# / VB) APPLICATION?

getProductKey() returns an std::vector which .Net doesn't understand. To make use of this in c# you'll need to write a c++ wrapper which returns the data as a managed array or list.

PROGRAM X ISN'T DETECTED. CAN YOU ADD SUPPORT FOR IT?

Generally the answer to this question is yes. See "Obtaining support" and I'll do my best to add it to the next maintenance release.

OBTAINING SUPPORT

Free e-mail support is offered to anyone who wants it. All I ask is that you be as descriptive as possible with your problems, including examples where possible.

For support, please e-mail support@davehope.co.uk