Software License Agreement

HTMl to DOCX Converter

For Win32/Win64

Vesion 14

SUB SYSTEMS, INC.

3200 Maysilee Street

Austin, TX 78728

512-733-2525

Software License Agreement

The Software is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The Software is licensed, not sold. This LICENSE AGREEMENT grants you the following rights:

A. This product is licensed per developer basis only. Each developer working with this package needs to purchase a separate license.

B. The purchaser has the right to modify and link the DLL functions into their application. Such an application is free of distribution royalties with these conditions: the target application is not a stand-alone HTMl to DOCX Converter; the target application uses this product for one operating system platform only; and the source code (or part) of the editor is not distributed in any form.

C. The DESKTOP LICENSE allows for the desktop application development. Your desktop application using this product can be distributed royalty-free. Each desktop license allows one developer to use this product on up to two development computers. A developer must purchase additional licenses to use the product on more than two development computers.

D. The SERVER LICENSE allows for the server application development. The server licenses must be purchased separately when using this product in a server application. Additionally, the product is licensed per developer basis. Only an UNLIMITED SERVER LICENSE allows for royalty-free distribution of your server applications using this product.

E. ENTERPRISE LICENSE: The large corporations with revenue more than $50 million and large government entities must purchase an Enterprise License. An Enterprise license is also applicable if any target customer of your product using the Software have revenue more than $500 million. Please contact us at info@subsystems.com for a quote for an Enterprise License.

F. Your license rights under this LICENSE AGREEMENT are non-exclusive. All rights not expressly granted herein are reserved by Licensor.

G. You may not sell, transfer or convey the software license to any third party without Licensor's prior express written consent.

H. The license remains valid for 12 months after the issue date. The subsequent year license renewal cost is discounted by 20 percent from the license acquisition cost. The

license includes standard technical support, patches and new releases.

I. You may not disable, deactivate or remove any license enforcement mechanism used by the software.

This software is designed keeping the safety and the reliability concerns as the main considerations. Every effort has been made to make the product reliable and error free. However, Sub Systems, Inc. makes no warranties against any damage, direct or indirect, resulting from the use of the software or the manual and can not be held responsible for the same. The product is provided 'as is' without warranty of any kind, either expressed or implied, including but not limited to the implied warranties of suitability for a particular purpose. The buyer assumes the entire risk of any damage caused by this software. In no event shall Sub Systems, Inc. be liable for damage of any kind, loss of data, loss of profits, interruption of business or other financial losses arising directly or indirectly from the use of this product. Any liability of Sub Systems will be exclusively limited to refund of purchase price.

Sub Systems, Inc. offers a 30 day money back guarantee with the product. Must call for an RMA number before returning the product.

Getting Started

This chapter describes the contents of the software diskettes and provides a step by step process of incorporating HTMl to DOCX Converter into your application.

In This Chapter

Files

The package contains the DLL and header files. The package also includes a set of files to construct a demo program. The demo program shows by example the process of linking the DLL to your program.

DLL Demo Files:

The following demo files are included in the c_demo.zip file.

DEMO.C	Source code for the demo program

DEMO.H	Include file for the demo program
DEMO.RC	Resource source file for the demo program
DEMO.DEF	Definition file for linking the demo program
DEMO.EXE	Executable demo program
DEMO_DLG.H	Dialog Identifiers for the demo program
DEMO_DLG.DLG	Dialog templates for the demo program
DEMO_DLG.RES	Compiled dialogs for the demo program
HDS.H	The *include* file to include into a C/C++ application module that calls the Hds routine. It contains the constant definitions and the prototypes for the API functions.
HDS32.DLL	The DLL file
HDS32.LIB	Import library for the HDS32 DLL
ter31.dll	Used internally by the HDS32.DLL
hts26.dll	Used internally by the HDS32.DLL
HDCC.DLL	Wrapper DLL to used with an ASP page

Visual Basic Interface and Demo Files:

HDS.BAS	Function declaration file.
DMO_VB.FRM	Demo form file.
DMO_VB.BAS	Demo variable declaration file.
DMO_VB.VPB	Demo project file.

License Key

Your License Key and License number are e-mailed to you after your order is processed.

You would set the license information using the HdsSetLicenseInfo static function. This should be preferably done before creating the converter session to avoid pop-up nag screens.

int HdsSetLicnseInfo(LPBYTE LicenseKey, LPBYTE LicenseNumber, LPBYTE CompanyName);

LicenseKey:	Your license key is available in the product delivery email sent to you upon the purchase of the product. It consists of a string in the form of "xxxxx-yyyyy-zzzzz".
LicenseNumber:	Your license number is also available in the product delivery email. The license number string starts with a "srab" or "smo" prefix.
CompanyName:	Your company name as specified in your order.

Return Value: This method returns 0 when successful. A non-zero return value indicates an error condition. Here are the possible return values:

0	License application successful.
1	Invalid License Key.
2	Invalid License Number.
3	Ran out of available licenses. Please consider purchasing additional licenses.

Example:

result=HdsSetLicenseInfo("xxxxx-yyyyy-zzzzz","srabnnnnn-n","Your Company Name")

Replace the 'xxxxx-yyyyy-zzzzz' by your license key, replace "srabnnnnn-n" with your license number, and "Your Company Name" with your company name as specified in your order.

Note: *HdsSetLicenseInfo method should be called only once at the beginning of your application. Calling this method for each conversion would degrade the conversion performance.*

Also, you can use the HdsGetLicenseStatus function at anytime to retrieve the license status.

Incorporating the DLL into Your Application

A C/C++ application should include the HDS.h file into the application module that needs to call the HDS32 dll. It also should include the HDS32.LIB as the linker library. Please refer to the demo application for an example.

A Visual Basic application needs to include the HDS.BAS file in the project. Please refer to the DMO_VB project for an example.

Please also make sure that the hds32.dll, txml2.dll and ter31.dll files are copied to a directory available at run-time.

.

Sample Conversion Code

First you would create a new conversion session:

```
dim id as long
```

Set the product license key and create a session id:

result=HdsSetLicenseInfo("xxxxx-yyyyy-zzzzz","srabnnnnn-n","Your Company Name")

```
id = HdsNewSession()
```

You would use the session id to call other conversion functions.

Here are sample code examples to convert HTMl to DOCX format.

1. Convert an HTML file to an DOCX file.

```
HdsConvertFile(id,"test.htm","test.docx")
```

2. Convert an HTML string to an DOCX string

```
Dim hMem as long
Dim OutSize as long
Dim HtmlString as string

hMem = HdsConvertBuffer(id,HtmlString, Len(HtmlString),
OutSize)
If (hMem <> 0) Then
   DocxString = Space$(OutSize + 1) ' allocate space for the
                                      output string
   HdsHandleToStr(DocxString, OutSize, hMem) ' copy docx from
              hMem global handle to the DocxString variable.
End If
```

After the conversion process, end the session by calling the HdsEndSession function. This frees up the memory used by the session.

```
HdsEndSession(id)
```

Application Interface functions

These API functions allow you to convert from HTMl to DOCX format. Your application must include the HDS.H file (c/c++), or HDS.BAS (VB) files. These files declare these functions.

The following is a description of the Hds API functions in an alphabetic order:

In This Chapter

HdsConvertBuffer

Convert HTMl to DOCX using memory buffers.

HGLOBAL HdsConvertBuffer(id, InString, InStringLen, OutStringLen)

DWORD id;	Session id.
LPBYTE InString;	Input string containing HTML document.

int InStringLen;	length of the input document string.
LPINT OutStringLen;	The variable to receive the length of the converted document.

Return value: This function returns a global memory handle containing the converted documented. You can either use the HdsHandleToStr or GlobalLock functions to access the data string contained in this global memory handle. GlobalLock is a Windows SDK function.

A null return values indicates an error.

Examples:

```
  Dim hMem as long
  Dim OutSize as long
  Dim HtmlString as string


  hMem = HdsConvertBuffer(id,HtmlString, Len(HtmlString),
OutSize)
  If (hMem <> 0) Then
     DocxString = Space$(OutSize + 1) ' allocate space for the
                                        output string
     HdsHandleToStr(DocxString, OutSize, hMem) ' copy docx
from
                hMem global handle to the DocxString variable.
  End If
```

HdsConvertFile

Convert HTMl to DOCX using disk files.

BOOL HdsConvertFile(id, InFile, OutFile)

DWORD id;	Session id.
LPBYTE InFile;	Input file containing HTML document
LPBYTE OutFile;	Output files, contains the converted document

Return value: This function returns TRUE when successful.

Examples:

```
HdsConvertFile(id,"test.htm","test.docx")
```

HdsEndSession

End a conversion session.

BOOL HdsEndSession(id)

DWORD id;	Session id.

Description: This function is called at the end of the conversion process to free up the session related resources.

Return Value: The function returns TRUE when successful.

HdsGetLastMessage

Get the last message.

int HdsGetLastMessage(id, HdsMessage, DebugMessage);

DWORD id;	Session id.
LPBYTE HdsMessage;	Returns the default user message text in English
LPBYTE DebugMsg;	Returns any debug message associated with the last message. The debug message need not be displayed to the user.

Return Value: This function returns the last message generated by the editor. This value is valid only if saving of the messages is enabled by setting the HRFLAG_RETURN_MSG_ID flag. This flag is set using the HdsSetFlags function.

HdsGetLicnseStatus

Get the license status.

int HdsGetLicnseStatus()

Return Value:

0 License application successful.

1 Invalid License Key.

2 Invalid License Number.

3 Ran out of available licenses. Please consider purchasing additional licenses.

4 The evaluation period has expired.

You can use the HdsGetLicenseStatus function at anytime to retrieve the license status.

HdsHandleToStr

Convert a global memory handle to a Visual Basic string.

BOOL HdsHandleToStr(string, length, hMem)

LPBYTE string;	pointer to a visual basic string
long length	length of the string
HGLOBAL hMem;	Global memory handle

Description: This function can be used to copy the contents of a global memory handle to a given visual basic string. The calling routine must expand the string to appropriate length before calling this function.

Example:

```
string=space(length)
HdsHandleToStr(string,length,hMem)
```

The input global memory handle is freed up after copying its contents to the string.

Return Value: This function returns TRUE if successful.

HdsNewSession

Create a new conversion session.

DWORD HdsNewSession()

Description: This function needs to be called before calling any other conversion function. This function creates a new conversion session.

The HdsEndSession must be called at the end to free up the session resources. All other conversion functions are called between the calls to the HdsNewSession and HdsEndSession functions.

Return Value: The function returns a non-zero session-id when successful. A zero value indicates a fail return.

HdsResetLastMessage

Reset the last editor message.

BOOL HdsResetLastMessage(id)

DWORD id;	Session id.

Description: This function can be called before calling any other function to reset the last error message.

Return Value: The function returns TRUE when successful.

See Also
HdsGetLastMessage
HdsSetFlags

HdsSetFlags

Set certain flags or retrieve the values of the flags.

DWORD HdsSetFlags(id, set, flags)

DWORD id;	Session id.
BOOL set;	TRUE to set the given flags, FALSE to reset the given flags
DWORD flags;	Flags (bits) to set or reset. Currently, the following flag

values are available:

HDFLAG_RETURN_MSG_ID	Do not display the error messages. Save the error code to be later retrieved using the HdsGetLastMessage function.
HDFLAG_EMBED_PICTURE	Convert the 'linked' pictures to the embedded pictures when saving to the DOCX format.

Return value: This function returns the new value of all the flags. Call this function with the 'flags' parameter set to zero to retrieve flag values without modifying it.

HdsSetPageMargin

Set the page margins for HTML output.

BOOL HdsSetPageMargin(id, left, right, top, bottom)

DWORD id;	Session id.
int left;	Left margin in twip units (1440 twips = 1 inch)
int right;	Right margin in twip units
int top;	Top margin in twip units
int bottom	Bottom margin in twip units

Return Value: The function returns TRUE when successful.

Comment: This function is used to override the default page margins when converting an HTML document to the DOCX format. This function should be called before calling the HdsConvertFile or HdsConvertBuffer if you wish override the page margin values.

HdsSetPaperOrient

Set the page orientation for HTML output.

BOOL HdsSetPaperOrient(id, orient)

DWORD id;	Session id.
int orient;	Orientation: DMORIENT_PORTRAIT or DMORIENT_LANDSCAPE

Return Value: The function returns TRUE when successful.

Comment: This function is used to override the default portrait orientation when converting an HTML document to the DOCX format. This function should be called before calling the HdsConvertFile or HdsConvertBuffer if you wish override the paper orientation.

HdsSetPaperSize

Set the page size for HTML output.

BOOL HdsSetPaperSize(id, PageSize, PageWidth, PageHeight)

DWORD id;	Session id.
int PageSize;	Use one of the following Windows SDK defined constants:

Constant	Value
DMPAPER_LETTER	1
DMPAPER_LEGAL	5
DMPAPER_LEDGER	4
DMPAPER_TABLOID	3
DMPAPER_STATEMENT	6
DMPAPER_EXECUTIVE	7
DMPAPER_A3	8
DMPAPER_A4	9
DMPAPER_A5	11
DMPAPER_B4	12
DMPAPER_B5	13

If you need to use a paper size not listed above, please set the PageSize argument to zero and specify the page width and height using the next two arguments.

int PageWidth;	The page width in twips units (1440 twips = 1 inch). This argument is ignored if the PageSize is set to one of the

	defined page sizes listed above.
int PageHeight;	The page height in twips units (1440 twips = 1 inch). This argument is ignored if the PageSize is set to one of the defined page sizes listed above

Return Value: The function returns TRUE when successful.

Comment: This function is used to override the default letter size paper when converting an HTML document to the DOCX format. This function should be called before calling the HdsConvertFile or HdsConvertBuffer if you wish override the paper size.

HdsSetTextProp

Set a text property for the session.

BOOL HdsSetTextProp(id, prop, val)

DWORD id;	Session id.
HDPROP_DOWNLOAD_DIR	Specify the project folder to store the temporary files.
LPBYTE val;	The text value of the selected property.

Return value: This function returns TRUE when successful.

HdsWriteToFile

Write the content of a global memory handle to a disk file.

BOOL HdsWriteToFile(id, OutFile, hMem, InSize)

DWORD id;	Session id.
LPBYTE OutFile;	Output file name.
HGLOBAL hMem;	Global memory handle

long length length of the string

Description: This function can be used to write the contents of a global memory handle to a file.

Example:

```
   hMem = HdsConvertBuffer(id, HtmlString, Len(HtmlString),
 OutSize)
   If (hMem <> 0) Then
      HdsHandleToStr(id, "test.docx",hMem,OutSize)
   End If
```

The input global memory handle is freed up after writing its contents to the file.

Return Value: This function returns TRUE if successful.

ASP Interface

This chapter describes the usage of the HTMl to DOCX Converter within an ASP page. The product includes an additional wrapper DLL called HDCC.DLL which is used to access the converter within an ASP page. Please follow the following steps:

Copy ter31.dll, hts26.dll, hds32.dll and hdcc.dll to the Windows system directory, or any other directory available at the run-time. Now register hdcc.dll using the regsvr32 system utility. The other dlls do not need registration. Now you are ready to use this product within an ASP page.

Here is an example ASP page to show a conversion of HTML string into an DOCX string:

```
<%@ LANGUAGE = "VBSCRIPT"%>
<%
Option Explicit

Dim sHtml
Dim DocxBytes
Dim obj

Set obj = Server.CreateObject("hdcc.converter")

call obj.SetLicenseInfo("Your-Licence-Key","Your-Order-
number","Your-company-name")

result=obj.SetTextProp(obj.VAL_HDPROP_DOWNLOAD_DIR,"c:\temp")
```

```
sHTML = "<html><body> This <b> is </b> a test of <i> HTML </i>
to <i> PDF </i> Conversion.</body></html>"


if len(sHtml) > 0 then
  DocxBytes = obj.ConvertHtmlToDocxBytes(CStr(sHtml))
End If


Set obj = Nothing



%>


<html>
<head>
</head>


<body>
<%


      if ArrayEmpty(DocxBytes) then
         call Response.Write("Conversion failed!" + " <br/>")
         if IsArray(DocxBytes) then Response.Write("size:
"+CStr(UBound(DocxBytes))+" <br/>") else
Response.Write("ConvertHtmlToDocxBytes returned null"+ "<br/")
      else


         Response.Clear()
         Response.Charset = ""
         Response.ContentType =
"application/vnd.openxmlformats-
officedocument.wordprocessingml.document"


         Response.AddHeader "Content-Disposition",
"attachment;filename=" + "test.docx"
         Response.AddHeader "Content-Length",
CStr(UBound(DocxBytes))


         call Response.BinaryWrite(DocxBytes)


         Response.Flush()
         Response.End()
```

```
        end if

        function ArrayEmpty (a)
          ArrayEmpty = true
          if IsArray(a) then
              if UBound(a) > 0 then ArrayEmpty = false
          end if
        end function
 %>
 </body>
 </html>
```

When the above asp file is loaded, IE displays generated docx codes

---

The method names used by the hdcc.dll are the same as the functions mentioned in the Application Interface functions. However the 'Hds' prefix is not used by the hdcc method names. For example, the HdsConvertFile function is named as ConvertFile within the hdcc.dll file.

Also, the constants values are prefixed with an 'VAL_' prefix. For example, the constant RDFLAG_SEGMENT_ONLY becomes VAL_RDFLAG_SEGMENT_ONLY.