Software License Agreement

HTML to PDF Converter

For Win32/Win64

version 18

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 PDF 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 PDF 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
HPS.H	The *include* file to include into a C/C++ application module that calls the Hps routine. It contains the constant definitions and the prototypes for the API functions.
HPS32.DLL	The DLL file
HPS32.LIB	Import library for the HPS32 DLL
ter31.dll	Used internally by the HPS32.DLL
hts26.dll	Used internally by the HPS32.DLL
pdc32.dll	Used internally by the HPS32.DLL
HPCC.DLL	Wrapper DLL to used with an ASP page

Visual Basic Interface and Demo Files:

HPS.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 HpsSetLicenseInfo static function. This

should be preferably done before creating the converter session to avoid pop-up nag screens.

int HpsSetLicenseInfo(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=HpsSetLicenseInfo("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: *HpsSetLicenseInfo 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 HpsGetLicenseStatus function at anytime to retrieve the license status.

Incorporating the DLL into Your Application

A C/C++ application should include the HPS.h file into the application module that needs to call the HPS32 dll. It also should include the HPS32.LIB as the linker library. Please refer to the demo application for an example.

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

Please also make sure that the hps32.dll, pdc32.dll, hts26.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=HpsSetLicenseInfo("xxxxx-yyyyy-zzzzz",
                "srabnnnnn-n","Your Company Name")

id = HpsNewSession()
```

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

Here are sample code examples to convert HTML to PDF format.

1. Convert an HTML file to a PDF file.

```
HpsConvertFile(id,"test.htm","test.PDF")
```

2. Convert an HtmlString to a PDFString

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

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

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

```
HpsEndSession(id)
```

Application Interface functions

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

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

In This Chapter

HpsConvertBuffer

Convert HTML to PDF using memory buffers.

HGLOBAL HpsConvertBuffer(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 HpsHandleToStr 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 = HpsConvertBuffer(id,HtmlString, Len(HtmlString),
OutSize)
   If (hMem <> 0) Then
      PDFString = Space$(OutSize + 1) ' allocate space for the
                                        output string
      HpsHandleToStr(PDFString, OutSize, hMem) ' copy PDF from
                 hMem global handle to the PDFString variable.
   End If
```

HpsConvertFile

Convert HTML to PDF using disk files.

BOOL HpsConvertFile(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:

```
HpsConvertFile(id,"test.htm","test.PDF")
```

HpsConvertHtmlToPdfBytes

Convert HTML string to PDF byte array.

ByteArray HpsConvertHtmlToPdfBytes(InString)

string InString; Input string containing HTML document.

Remark: This method is available only when using the hpcc.dll COM object.

Return value: This function returns byte array containing the converted documented.

A null return values indicates an error.

Examples:

```
PdfBytes = obj.ConvertHtmlToPdfBytes(HtmlString)
```

```
Response.BinaryWrite(PdfBytes)
```

HpsEndSession

End a conversion session.

BOOL HpsEndSession(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.

HpsGetLastMessage

Get the last message.

int HpsGetLastMessage(id, HpsMessage, DebugMessage);

DWORD id;	Session id.
LPBYTE HpsMessage;	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 HPFLAG_RETURN_MSG_ID flag. This flag is set using the HpsSetFlags function.

HpsGetLicnseStatus

Get the license status.

int HpsGetLicnseStatus()

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 HpsGetLicenseStatus function at anytime to retrieve the license status.

HpsHandleToStr

Convert a global memory handle to a Visual Basic string.

BOOL HpsHandleToStr(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)
HandleToStr(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.

HpsNewSession

Create a new conversion session.

DWORD HpsNewSession()

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

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

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

HpsResetLastMessage

Reset the last editor message.

BOOL HpsResetLastMessage(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
HpsGetLastMessage
HpsSetFlags

HpsSetFlags

Set certain flags or retrieve the values of the flags.

DWORD HpsSetFlags(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:
HPFLAG_RETURN_MSG_ID	Do not display the error messages. Save the error code to be later retrieved using the HpsGetLastMessage function.

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.

HpsSetBoolProp

Set a boolean property for the PDF document.

BOOL HpsSetBoolProp(id, prop, val)

DWORD id;	Session id.
int prop;	One of the following property type to set:
HPPROP_USE_WEB_BROWSER	Use this property to instruct the converter to use the Internet Explorer control to render the html page to convert to pdf. This option results in pdf pages similar to as displayed by IE. When this property is not set, the converter

	uses a built-in html renderer which might be suitable for html files that use simpler html formatting.
HPPROP_COMPRESS_TEXT	Compress text inside the PDF document. Default = false.
HPPROP_EMBED_FONTS	Embed font data in the PDF document. Default = false.
HPPROP_BOOKMARK	Include bookmarks. Default = true.
HPPROP_OPEN_BOOKMARK_PANE	Open the bookmark pane when PDF is displayed. Default = false.
HPPROP_HI_RES	Produce PDF in 1440 dpi resolution.
HPPROP_RC4_128	Enable RC4 128 bit security when password is specified.
HPPROP_AES_128	Enable AES 128 bit security when password is specified.
HPPROP_HYPERLINK	Include hyperlinks. Default = true.
HPPROP_HIDE_EMPTY_CELLS	Hide empty cells. Defalt = true.

LPBYTE val; The text value of the selected property.

Return value: This function returns TRUE when successful.

HpsSetHdrFtrText

Set header or footer text.

BOOL HpsSetHdrFtrText(id, HdrFtrType, TextType, text)

DWORD id; Session id.

int HdrFtrType; Select header or footer to set:

HF_FIRST_HDR	Header text to print on the first page.
HF_FIRST_FTR	Footer text to print on the first page..

	HF_HDR	Regular header for all pages. When the first page header is also set, then the regular header text is printed on all pages except the first page.
	HF_FTR	Regular footer for all pages. When the first page footer is also set, then the regular footer text is printed on all pages except the first page.

int TextType;	Text type:	
	HFTYPE_TEXT	Plain text.
	HFTYPE_RTF	RTF text.
	HFTYPE_HTML	Html text.
LPBYTE text;	Header or footer text. The header/footer text must be specified as plain text or RTF text depending upon the value passed for the 'TextType' parameter.	

Comment: The function should be called before calling the conversion functions to set the header or footer text. You can call this function multiple times to set various types of header or footer.

Return value: This function returns TRUE when successful.

Examples:

```
HpsSetHdrFtrText(id, HF_FIRST_HDR, HFTYPE_TEXT,
                 "This is first page header.");
HpsSetHdrFtrText(id, HF_FIRST_FTR, HFTYPE_TEXT,
                 "This is first page footer.");
HpsSetHdrFtrText(id,HF_HDR, HFTYPE_TEXT,
                 "This is regular page header.");
HpsSetHdrFtrText(id,HF_FTR, HFTYPE_RTF,"{\\rtf1 \\qc
    Page: {\\field{\\fldinst PAGE}{\\fldrslt 12}} of
    {\\field{\\fldinst NUMPAGES}{\\fldrslt 12}}
    \\par}" );  // rtf example to insert page: n of m string
```

HpsSetNumProp

Set a numeric property for the PDF document.

BOOL HpsSetNumProp(id, prop, val)

DWORD id; Session id.

int prop; One of the following property type to set:

HPPROP_ZOOM_FACTOR — To compress or enlarge the page image. The default zoom factor is 1000.

One of the uses of this property is to fit a large text into one page. You would use the HpsSetPaperSize function to specify a larger paper size so that the entire text is displayed on one page. You would then use a zoom factor less than 1000 to reset the page to actual size. For example if the page was enlarged by 20 percent, then you would set zoom factor to 800 (20 percent less than the default 1000) value.

HPPROP_RES_FACTOR — Set the resolution factor from 1 to 10, default = 1.

HPPROP_PERM_FLAGS — Permission flags. The permission flags is effective only when UserPassword or OwnerPassword is also specified using the HpsSetTextProp method.

One or more of the following permission flags can be specified using the 'val' parameter:

PERM_PRINT	Allow printing of the document
PERM_MOD	Allow modification
PERM_COPY	Allow copying of the document to clipboard.

You can specify more than one permission flags using the 'or' operator.

int val; The numeric value of the selected property.

Return value: This function returns TRUE when successful.

HpsSetPageMargin

Set the page margins for PDF output.

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

BOOL HpsSetPageMargin2(id, left, right, top, bottom, HdrDist, FtrDist)

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
int HdrDist	The distance of the header text from the top of the page in twip units (default 720 twips). This parameter is used by the TerSetPageMargin2 method only.
int FtrDist	The distance of the footer text from the bottom of the page in twip units (default 720 twips). This parameter is used by the TerSetPageMargin2 method only.

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 PDF format. This function should be called before calling the HpsConvertFile or HpsConvertBuffer if you wish override the page margin values.

HpsSetPaperOrient

Set the page orientation for PDF output.

BOOL HpsSetPaperOrient(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 PDF format. This function should be called before calling the HpsConvertFile or HpsConvertBuffer if you wish override the paper orientation.

HpsSetPaperSize

Set the page size for PDF output.

BOOL HpsSetPaperSize(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 PDF format. This function should be called before calling the HpsConvertFile or HpsConvertBuffer if you wish override the paper size.

HpsSetTextProp

Set a text property for the PDF document.

BOOL HpsSetTextProp(id, prop, val)

DWORD id;	Session id.
int prop;	One of the following property type to set:

HPPROP_AUTHOR	Author of the document
HPPROP_TITLE	Document title
HPPROP_SUBJECT	Subject
HPPROP_KEYWORDS	Key words
HPPROP_CRE_DATE	Creation date
HPPROP_MOD_DATE	Modification date
HPPROP_USER_PASSWORD	User password
HPPROP_OWNER_PASSWORD	Owner password
HPPROP_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.

ASP Interface

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

Copy ter31.dll, hts26.dll, pdc32.dll, hps32.dll and hpcc.dll to the Windows system directory, or any other directory available at the run-time. Now register hpcc.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 a PDF string:

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

Dim sHTML
Dim sPdf
Dim obj
Dim result
Dim ErrorMessage

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

result=obj.SetTextProp(obj.VAL_HPPROP_DOWNLOAD_DIR,
                    "c:\inetpub\wwwroot\MyProjectFolder")
result=obj.SetBoolProp(obj.VAL_HPPROP_USE_WEB_BROWSER,1)

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

sPDF=""
if len(sHTML) > 0 then
  sPdf = obj.ConvertBuffer(CStr(sHTML))
End If

if sPDF = "" then
  ErrorMessage = obj.GetLastMessage
  response.write(ErrorMessage)
end if

Set obj = Nothing
%>

<html>
<head>
</head>

<body>
```

```
<%

    Function StringToByteArray(S)
      Dim i, ByteArray
      For i=1 To Len(S)
        ByteArray = ByteArray & ChrB(Asc(Mid(S,i,1)))
      Next
      StringToByteArray = ByteArray
    End Function

    if sPDF<>"" then
        Response.Clear()
        Response.Charset = ""
        Response.ContentType = "application/pdf"

        Response.AddHeader "Content-Disposition",
                    "inline;filename=" + "test.pdf"

        Response.BinaryWrite(StringToByteArray(sPDF))

        Response.Flush()
        Response.End()
     end if

%>

</body>
</html>
```

When the above asp file is loaded, IE displays generated PDF file.

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

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

FAQ

How to Insert Page break in the generated PDF?

Please insert the following html code in your html document to generate a page break in PDF

```
<br style="page-break-before: always;" />
```

How to fit large html text on one page?

You would set the zoom factor property to shrink the text to fit on one page. Please refer to the TerSetNumProp method for the description of the HPPROP_ZOOM_FACTOR property.