DrawHTML

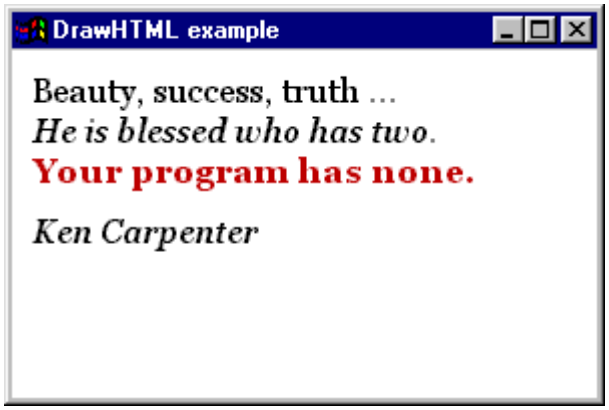
Ukkie9, 29 Oct 2004

4.79 (44 votes)

Rate: Vote!

A drop-in replacement for the DrawText() SDK function with minimal HTML support.

Download source files - 5.51 Kb



Introduction

The `DrawHTML()` function is nearly a drop-in replacement of the standard `DrawText()` function, with limited support for HTML formatting tags. It is only *nearly* a replacement for `DrawText()`, because a few formatting flags of the real `DrawText()` function are not supported. More limiting, perhaps, is that only a minimal subset of HTML tags is supported.

`DrawHTML()` is inspired by *Pocket HTML* by Petter Hesselberg; published in Windows Developer Journal, February 2000. The implementation is fully mine, however, because I felt that the HTML parser in Pocket HTML leaves to be desired. My implementation is very limited, but already more complete than that of Pocket HTML; it is more easily extensible; and it scales better by allocating resources on an "as-needed" basis, rather than grabbing all*possibly needed* resources on start-up of the function.

Why `DrawHTML()` when there are full HTML parsers with comprehensive support for all tags? My reasons for implementing this are:

- Windows API functions do not support markup codes *at all*; very limited markup support is already very helpful. This is why Petter Hesselberg published Pocket HTML (and my implementation improves on it).
- `DrawHTML()` consists of a single, and fairly small, file (less than 400 lines). It is therefore fairly easy to add it to a project. No extra components or DLLs are needed for the application, because `DrawHTML()` can easily be statically linked.
- `DrawHTML()` makes a single pass through the string; apart from obtaining the needed font resources,`DrawHTML()` does not allocate memory or other resources. As a result, `DrawHTML()` is lightweight and quick.
- `DrawHTML()` is actually robust if what you throw at it is not fully valid HTML: many people forget to replace "&" by "`&`", "ü" by "`&uuml;`" and "<" by "`<`". A browser may skip the code or print the wrong character,

but DrawHTML() will print "&", "ü" and "<" if they appear in the string. This way, you can display strings with DrawHTML() without taking extra care for reserved characters like "&" and "<".

Using the code

The function prototype for DrawHTML() is the same as that of the standard Win32 SDK function DrawText(). In your program, you would use DrawHTML() just like you would use DrawText().

The source code archive (see the top of this article) contains a little demo program. The relevant Cls_OnPaint()function is reproduced below:

```
static void Cls_OnPaint(HWND hwnd)
{
  PAINTSTRUCT PaintStruct;
  BeginPaint(hwnd, &PaintStruct);
  HFONT hfontOrg = (HFONT)SelectObject(PaintStruct.hdc,
                                       hfontBase);

  RECT rc;
  GetClientRect(hwnd, &rc);
  SetRect(&rc, rc.left + Margin, rc.top + Margin,
              rc.right - Margin, rc.bottom - Margin);

  DrawHTML(PaintStruct.hdc,
          "<p>Beauty, success, truth ..."
          "<br><em>He is blessed who has two.</em>"
          "<br><font color='#C00000'><b>Your program"
          " has none.</b></font>"
          "<p><em>Ken Carpenter</em>",
          -1,
          &rc,
          DT_WORDBREAK);

  SelectObject(PaintStruct.hdc, hfontOrg);
  EndPaint(hwnd, &PaintStruct);
}
```

There is a bit of scaffolding code around the call to DrawHTML(), to offset the text from the frame of the window and to select a bigger font. The font, hfontBase, is created in the Cls_OnCreate() function (not shown). I used a 20-pixel high TrueType font to better show the effects of italics and bold.

As I wrote already, the HTML support by DrawHTML() is very limited:

- The only supported tags are <p>, <br>, .., .., .., <u>..</u>,.., .., _{..} and ^{..}. The tags .. are equivalent to .., and .. map to ...

- The `<font>` tag is only supported in the extent that you can change the text color with it. It is also the only tag that can take a parameter, and this parameter should be "`color`" with a value in the well known HTML hexadecimal notation. For example, "`<font color='#ffff00'>`" (this is yellow, by the way).
- With the exception of tags that take parameters (currently only the `<font>` tag), there may be no spaces in the tags; `<p>` is okay, but `<p align='right'>` will be considered as two words "<p" and "align='right'>". That's right: when `DrawHTML()` considers that something is not a valid HTML tag, it prints it as a word.
- Any special characters like `&lt;` and `&agrave;` are unsupported, you must just type in the correct characters. That is, you can just use the characters "à" and "&" in the text, and "<" too.

Points of Interest

`DrawHTML()` is Unicode-compatible, but in a way different than a web-browser does it: instead of using an 8-bit encoding for the Unicode data (UTF-8), you just pass in a "wide character" string. To have Unicode support, you should compile the `DrawHTML()` source code with the `UNICODE` and `_UNICODE` macros defined.

The code for `DrawHTML()` consists of three blocks:

1. There is a simple HTML parser. The parser is fairly strict, and it has a fall-back in that everything that it does not recognize is "plain text". This includes unknown tags, and there `DrawHTML()` differs from browsers, which ignore unknown HTML tags.
2. The text drawing function, which outputs text word for word, and handles line breaking and the calculation of the the size of the bounding box.
3. A simple small color stack for the colors set with the `<font>` tag. When changing a color, the old color must be saved somewhere, so that the `</font>` tag can restore it. Hence, the stack.

A few "formatting flags" of the `DrawText()` function are not supported by `DrawHTML()`. These are related to alignment (horizontal and vertical) and to setting TAB stops. Supporting horizontal and vertical alignment requires an extra pass over the text, to get the full height and the width of each individual line. Specifically, the following formatting flags of the `DrawText()` function are *not* supported:

Flag	Description
`DT_CENTER`	center text lines horizontally
`DT_RIGHT`	align text lines to the right border
`DT_TABSTOP`	expand TAB characters (to 8 spaces)

These three flags are ignored if they are set. The "&" character is *never* a "prefix character" in `DrawHTML()`, so the`DT_NOPREFIX` flag is not necessary.

More noteworthy, in fact, is that all the other flags *are* supported, specifically the flags `DT_SINGLELINE`, which causes the tags `<p>` and `<br>` to be ignored, and `DT_CALCRECT`, which calculates the bounding rectangle for the text without actually drawing it. Compatibility with `DrawText()` is furthermore improved by using `DrawText()`in the back end to actually *draw* the text after having parsed the HTML code.

History

- 26 October 2004:
  - `DT_BOTTOM` and `DT_VCENTER` are allowed, as the original function `DrawText()` defines them only for single line text and this is compatible with `DrawHTML()`.

- The underline style (`<u>...</u>`) no longer causes gaps in the underline between the words; i.e., space characters between the words are now underlined too.
- A fix for negative heights was implemented (thanks to "Sims", see the comments at the bottom of this article).
- I added support for the HTML tags `<sup>...</sup>` and `<sub>...</sub>`, for superscript and subscript respectively. Only one level of these tags is supported, however (so a`<sub>b<sub>c</sub></sub>` will put "b" and "c" at the same base line).

- 11 August 2004: The article was enhanced with a screen shot and an example for the use of the function. The archive now contains a demo program.
- 10 August 2004: First release.

License

This article has no explicit license attached to it but may contain usage terms in the article text or the download files themselves. If in doubt please contact the author via the discussion board below.

A list of licenses authors might use can be found here

About the Author

Ukkie9

Software Developer (Senior) ITB CompuPhase

Netherlands

Thiadmer Riemersma develops system software, embedded software and multimedia software for his company CompuPhase in the Netherlands.