Line-height

The line-height of text can be controlled using the CSS property `line-height` (since mPDF 4). This used a fixed value for the line-height when `normal` was specified (the default CSS value) - set by the configurable variable `normalLineheight`. It also used a fixed proportion to determine the position of the text baseline (non-configurable).

Form mPDF 6, you can (optionally) use font metrics derived from each font file to:

- Determine the height of a line when line-height is set to 'normal'
- Determine the glyph baseline (previously a fixed value)

Options are set by configurable variables in the `config.php` file:

Default settings in mPDF versions 6 - recommended especially for complex scripts with marks used above or below characters:

```
$this->useFixedNormalLineHeight = false;
$this->useFixedTextBaseline = false;
$this->adjustFontDescLineheight = 1.14;
```

Settings to be backwards compatible with mPDF versions < 6:

```
$this->useFixedNormalLineHeight = true;
$this->useFixedTextBaseline = true;
$this->normalLineheight = 1.33;
```

Using the font metrics will give approximately the same result as the fixed value for many standard Latin script fonts e.g. DejaVu Sans Condensed:

However, for some fonts the normal line-height using font metrics will be significantly taller, to account for the design of the font glyphs e.g. Khmer font:

For more information on how complex normal lineheights are, see Eric Meyers' website: http://meyerweb.com/eric/thoughts/2008/05/06/line-height-abnormal/ and http://typophile.com/node/13081

CSS control of line-height

From mPDF v 6.0 there are new controls for line-height using draft CSS3 properties. These can be set on all block level elements (P, DIV etc) and tables (TABLE/TD/TH).

line-stacking-strategy = `inline-line-height | block-line-height | max-height | grid-height`

- `inline-line-height` - [default] lineheight is initially calculated from the block-level font[-size]; the height is expanded by any inline content, including the calculated lineheight of that inline content;
- `block-line-height` - lineheight is fixed as the lineheight of the block-level font[-size];
- `max-height` - lineheight is initially calculated from the block-level font; the height is expanded by any inline content, EXCLUDING the calculated lineheight of that inline content;
- `grid-height` - lineheight is initially calculated from the block-level font; the height is expanded - AS MULTIPLES OF INITIAL LINEHEIGHT - by any inline content, EXCLUDING the calculated lineheight of that inline content;

Note: XSL has a similar property with the same name, which uses different but equivalent values: `line-height` instead of `inline-line-height`, `font-height` instead of `block-line-height`. It also uses `max-height`. The value `grid-height` is new to the CSS3 property.

line-stacking-shift = `consider-shifts | disregard-shifts`

This property determines whether to include or disregard the adjusted top- and bottom-edge of any characters that have a baseline-shift (e.g. superscript) when calculating lineheight.

Note: XSL has a similar property with a different name: `line-height-shift-adjustment` which uses the same values.

For more details see the CSS3 draft specification.

Note for Advanced users

There are actually three possible metrics that can be used in a TrueType font file. The differences are

summed up quite well in this article at http://typophile.com/node/13081. mPDF will by default use the usWinAscent and usWinDescent values to determine a 'normal' line-height, with two variations:

- if either the usWinAscent or usWinDescent are greater than the font bounding box (yMin yMax), then the values are reduced to equal the yMin/yMax values. NB this works as a fix with Myanmar Text (Windows 8 version) to give a line-height normal that is equivalent to that produced in browsers.
- if the USE_TYPO_METRICS bit is set on fsSelection (OS/2 table), this is telling the font to use the sTypo values and not the usWinAscent values. NB this works as a fix with Cambria Math font to give a normal line-height; at present, this is the only font I have found with this bit set; although note that MS WordPad and Windows FireFox browser use the big line-height from usWinAscent, whilst MS Word 2007 observes the fSelection value.

You can change the font metrics used by mPDF, by editing the defined constant (`_FONT_DESCRIPTOR`) at the top of the mpdf.php file:

- '`winTypo`' uses sTypoAscender etc from the OS/2 table and is the one officially recommended - BUT
- '`win`' use usWinAscent etc from OS/2 and in practice seems to be used most commonly in Windows environment; this is the default in mPDF;
- '`mac`' uses Ascender etc from hhea table, and may be used to give results consistent with a Mac/OSX environment.

Finally, you can override values for Ascent, Descent and Leading for any specific font, by setting values in `config_font.php` e.g.

```
"cambriamath" => array(
    'R' => "cambria.ttc",
    'useOTL' => 0xFF,
    'TTCfontID' => array( 'R' => 2, ),
    'Ascent' => 950,
    'Descent' => -222,
    'Leading' => 0,
),
```

Note - The same values are used for all styles of the font-family. Descent values should be negative. All values should be given using a 1000 units per em scale, regardless of the UnitsPerEm used in the font design.

Notes

Remember that line-height for a TABLE has a default value (1.2) set in the `config.php defaultCSS`. This is left in for backwards compatability. You can change this value to 'normal' for results consistent with most browsers.

Line-height in a <textarea> is fixed and defined in `classes/mpdfform.php` (= 1.2)

Details of the font metrics can be seen by inspecting the temporary font files e.g. `/ttfontdata/[fontname].mtx.php`.

Printed on Wed 05 Aug 2015 12:09:29 GMT +0100 (DST)