Thelo

a a a

Thelo est une famille typographique dédiée à la lecture sur écran. À travers ses différents corps optiques, elle propose d'explorer les possibilités d'un affichage dynamique du texte.

Thelo est née d'une réflexion sur l'utilisation de la typographie sur supports numériques. La construction de la famille s'inspire des réglages optiques couramment pratiqués par les graveurs de poinçons à l'époque de la typographie en plomb. Appliqué à la typographie numérique, le principe des corps optiques permet d'optimiser le confort de lecture des caractères à l'écran.

Contrairement aux pratiques communément admises qui visent à conserver des formes identiques quelle que soit la taille de corps sélectionnée, les caractères de Thelo présentent, en fonction de leurs usages et de leur échelle de composition, des variations dans leurs formes : ceci permet de pourvoir répondre à différents contextes de lecture sur le web.

La famille dans son ensemble été conçue à l'aide d'une grille simulant les pixels et les subpixels. Cette structure a été le point de départ pour le dessin de trois corps spécifiques, 8, 16 et 32 pixels, correspondant respectivement aux corps Micro, Texte et Grand. L'utilisation d'une valeur d'UPM *(units per em)* constante permet un rapport cohérent entre les trois variantes. Cette rigoureuse méthode de dessin simule les contraintes d'affichage des supports numériques et permet d'obtenir un rendu à l'écran propre et harmonieux.

Thelo a été nommé d'après le *Thelocactus*, une variété de cactus originaire du Mexique : un rapprochement entre l'aspect rude de l'affichage à l'écran et les terres arides des zones désertiques.

Thelo Texte

Thelo Texte est idéal pour la composition des textes qui nécessitent une lecture soutenue à l'écran. Sa silhouette rectiligne est atténuée par ses connexions lumineuses et ses terminaisons en pointe. Son dessin est optimisé pour un affichage en 16 pixels, corps standard utilisé par la majorité des navigateurs web.

Thelo Texte Italique

L'Italique est inspiré de l'italique moderne de Pierre-Simon Fournier. Sa couleur homogène et ses empattements solides et horizontaux rendent cet italique adapté à la lecture sur écran. Grâce à sa pente modérée cet Italique peut être utilisé pour des passages de texte longs.

Thelo Texte Gras

Le Gras a une silhouette dense, présentant un contraste très marqué avec le romain. La largeur de ses fûts mesure exactement deux fois la largeur des fûts du romain, un rapport qui respecte les principes d'optimisation du dessin pour l'affichage numérique.

Thelo Grand

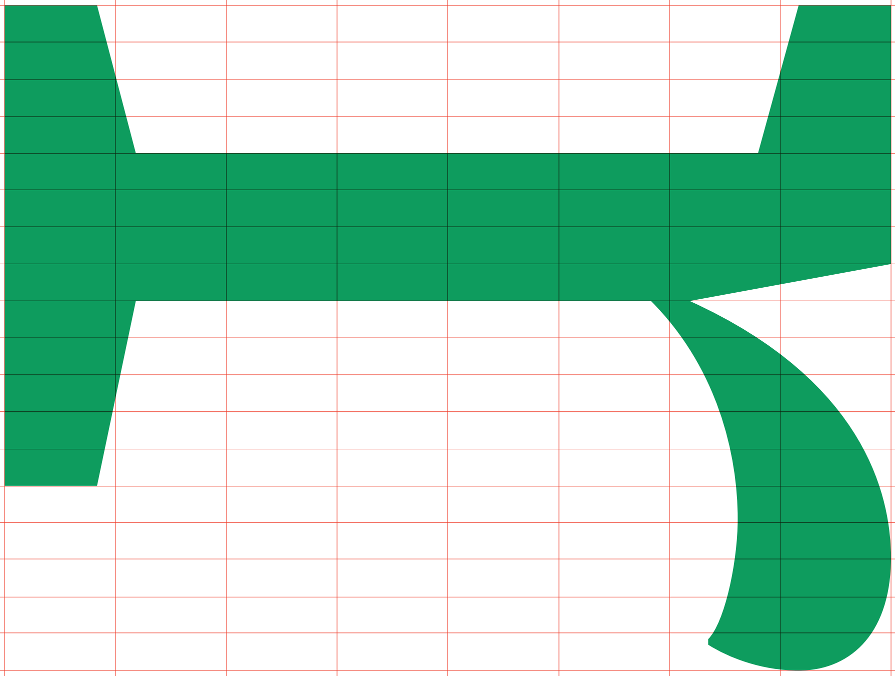
Avec son contraste subtil et ses empattements raffinés, Thelo Grand est destiné au titrage. Son utilisation est conseillée pour des corps allant de 18 à 64 pixels. Le Grand est le seul des trois corps à intégrer des dépassements optiques dans la construction de ses lettres rondes.

Thelo Micro

Thelo Micro a été pensé pour les très petits corps. Radicale et robuste, cette version est destinée aux situations de lecture extrêmes. Son très faible contraste et ses empattements carrés la rendent stable à l'écran. Son utilisation est conseillée pour les corps allant de 8 à 14 pixels. Sur des supports imprimés, Thelo Micro peut être employé à partir de 5 points.

Thelo Micro Italique

Encore au stade de développement, Thelo Micro Italique a été conçu en reprenant les éléments formelles du romain. Il répond au besoin de hiérarchisation des contenus bibliographiques et textes de référence, souvent présentés en petit corps.



960 UPM ÷ 16 PPEM = 60 UNITÉS = 1 PIXEL

contextual

92 pt

DESIGN

114 pt

@font-face property

51 pt

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z À Á Â Ã Ä Å

Ā Ă Ą Æ Ǽ Ç Ć Ĉ Ċ Č Ď Đ È É Ê Ë Ē Ĕ Ė Ę Ě Ĝ Ğ Ġ Ģ Ĥ Ħ

Ì Í Î Ï Ĩ Ī Ĭ İ Į Ĵ Ķ Ĺ Ľ Ł Ñ Ń Ň Ņ Ò Ó Ô Õ Ö Ø Ǿ Ō Ŏ Ő Œ Ŕ

Ř Ŗ Ś Ŝ Ş Š Ș Ť Ţ Ŧ Ù Ú Û Ü Ũ Ū Ŭ Ů Ű Ų Ẁ Ẃ Ŵ Ẅ Ỳ Ý Ŷ

Ÿ Ź Ž Ż Ŋ Ð Þ ã ä å ā ă à á â ą æ ǽ ć ĉ ċ č ç ď đ è é ê ë ē ĕ ė ę

ě ĝ ğ ġ ǵ ĥ ħ ì í ï ĩ ī ĭ ĵ ķ ĺ ľ ł ń ň ñ ņ ò ó ô õ ö ō ŏ ő ø ǿ œ ŕ ř

ŗ ś ŝ ş š ș ß ť ŧ ù ú û ü ū ŭ ů ű ų ẁ ẃ ŵ ẅ ỳ ý ŷ ÿ ź ž ż ŋ ð þ

0 1 2 3 4 5 6 7 8 9 ¡ ! ¿ ? . , : ; . . . _ - – — / \ | ¦ ( ) [ ] { } ‘ ’ “ ”

‚ „ ' " ‹ › « » ° • + − ± × ÷ = ≠ ≈ < > ≤ ≥ € $ ¢ # ﬁ ﬂ @ &

10/14 pt

HINTING, OR SCREEN OPTIMISING, IS THE PROCESS BY WHICH FONTS ARE ADJUSTED FOR MAXIMUM READABILITY ON COMPUTER MONITORS. I HAVE been designing type since the early 1990s, and for as long as I can remember, type designers have been saying that hinting would soon be made obsolete by new ad-vances in hardware and software. Now, almost 20 years later, hinting seems to be more relevant than ever.

12/17 pt

THE PROBLEM IS THAT TYPICAL MODERN FONTS ARE NOT DESIGNED PRIMARILY FOR THE 72–96 DPI RESOLUTION OF computer screens, but for the much higher 1,200+ dpi resolution of print media. The letterforms are designed and stored as outlines, mathematically perfect lines and curves which look great at high resolutions, but can be distorted or even illegible when converted into groups of pixels, the on-or-off dots that make up a computer's dis-play. And although there has been much discussion about

15/21 pt

THIS IS THE REASON THAT WEBPAGE DESIGNERS HAVE LONG BEEN MORE OR LESS LIMITED TO A dozen or so fonts (Verdana, Georgia, Arial, etc.) that have been fine-tuned by hand so that typical text sizes (9–14pt) display well at low resolutions. These fonts are so common that most computer users think of them as free, but the reality is that Verdana, for example, is probably the most expensive, labor-intensive font ever produced. It includes characters used to write an extremely wide range of languages, and each of these characters had to be adjusted to be readable at every point size between 9 and 60 (at 60pt the resolution is sufficient to display the letterforms accurately).

dynamic

124 pt

SUBSETTING

66 pt

Beta version baby!

55 pt

10/14 pt

THIS IS EXACTLY WHAT HINTING IS ABOUT: PROGRAMMING INSTRUCTIONS THAT FINE-TUNE A FONT'S RASTERISATION, THE PROCESS BY which its mathematically ideal outlines are mapped onto a monitor's pixels. Hinting can control the heights and widths of a font's uppercase and lowercase letters, the widths of its individual lines, the amount of white space around letters, the size at which uppercase letters start to use different stem-widths from lowercase

12/17 pt

HOW THE ANGLE OF ITALIC CHARACTERS CHANGES TO BEST FIT THE PIXEL GRID, AND MANY OTHER EXTREMELY technical details, all on a pixel-by-pixel basis. If this sounds like a rather tedious, time-consuming activity, it is, (even for type designers, who are accustomed to tedious, time-consuming activities). Last year there was considerable hype about a function that makes it possible

15/20 pt

WHEN FONTS ARE HINTED OPTIMUM ON-SCREEN RESULTS ARE STILL NOT GUARANTEED different font technologies approach hinting differently. In the PostScript system most of the font scaling is handled not by the fonts, but by the rasteriser software,

FUNCTIONAL HARDWARE

33 pt

responsive

84 pt

layout

143 pt

10/14 pt

A LOT HAS BEEN WRITTEN ABOUT HOW MAC OS RENDERS TEXT COMPARED TO WINDOWS. I WILL NOT GO INTO DETAILS HERE, but the primary difference is that Microsoft's rasteriser tries to align characters to whole pixel grid, with the result that 'Regular' weights look lighter, 'Bold' weights look heavier, and subtle details of design can be lost at small point sizes. Apple's rasteriser tries to preserve the

12/17 pt

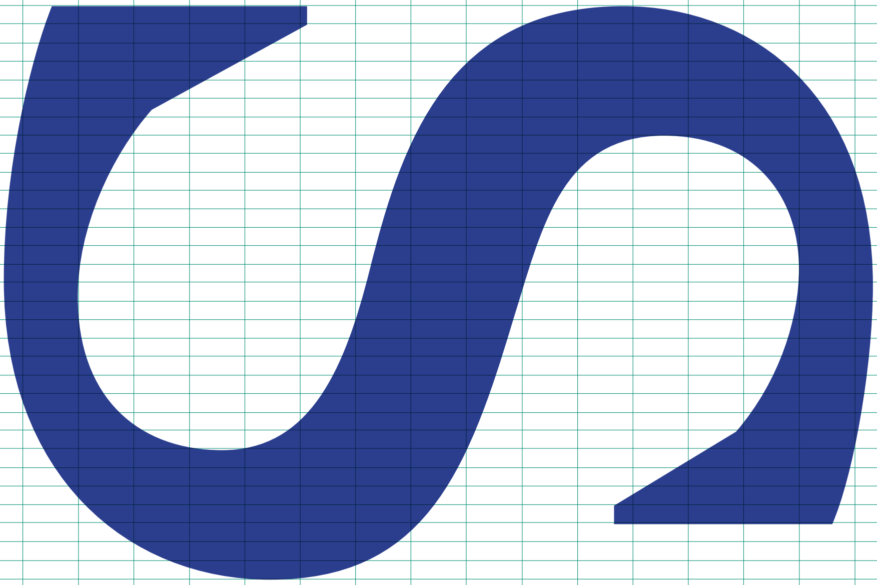
BLACK AND WHITE HINTING, DEVELOPED IN THE DAYS WHEN OPERATING SYSTEMS COULD ONLY TURN PIXELS on or off, controls which pixels will be displayed at a given point size. This kind of hinting is called grid-fitting because the outlines of the font are significantly modified to fit the

15/20 pt

ANTI-ALIASING IS A TECHNIQUE THAT WAS INTRODUCED IN WINDOWS 98. IT SMOOTHS visibly jagged lines by using varying shades of grey to render type on screen, so instead of being limit-ed to only black or white pixels, the rasteriser can also choose to compromise between them.

Thelo Grand

664 pt



960 UPM ÷ 32 PPEM = 30 UNITÉS = 1 PIXEL

Live Interpolation &

52 pt

OPTICAL

106 pt

adjustments

85 pt

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z À Á Â Ã Ä Å Ā

Ă Ą Æ Ǽ Ç Ć Ĉ Ċ Č Ď Đ È É Ê Ē Ĕ Ė Ę Ě Ĝ Ğ Ġ Ģ Ĥ Ħ Ì Í Î

Ï Ĩ Ī Ĭ İ Į Ĵ Ķ Ĺ Ľ Ł Ñ Ń Ň Ņ Ò Ó Ô Õ Ö Ø Ǿ Ō Ŏ Ő Œ Ŕ Ř Ŗ

Ś Ŝ Ş Š Ș Ť Ţ Ŧ Ù Ú Û Ü Ũ Ū Ŭ Ů Ű Ų Ẁ Ẃ Ŵ Ẅ Ỳ Ý Ŷ Ÿ

Ź Ž Ż Ŋ Ð Þ ã ä å ā ă à á â ą æ ǽ ć ĉ ċ č ç ď đ è é ê ë ē ĕ ė ę ě

ĝ ğ ġ ģ ĥ ħ ì í ï ĩ ī ĭ ĵ ķ ĺ ľ ł ń ň ñ ņ ò ó ô õ ö ō ŏ ő ø ǿ œ ŕ ř ŗ ś

ŝ ş š ș ß ť ŧ ù ú û ü ū ŭ ů ű ų ẁ ẃ ŵ ẅ ỳ ý ŷ ÿ ź ž ż ŋ ð þ 0 1 2

3 4 5 6 7 8 9 ¡ ! ¿ ? . , : ; … _ - – — / \ | ¦ ( ) [ ] { } ‘ ’ “ ” ‚ „

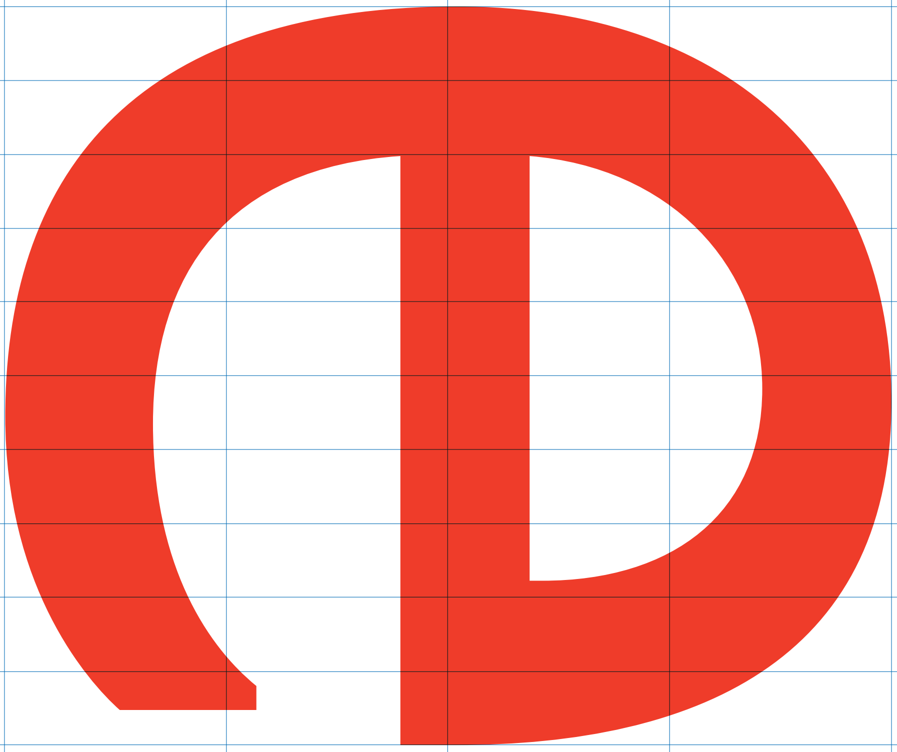
' " ‹ › « » ° • + − ± × ÷ = ≠ ≈ < > ≤ ≥ € $ ¢ # ﬁ ﬂ @ &

18/26 pt

CLEARTYPE IS A MICROSOFT PROPRIETARY SUBPIXEL RENDERING TECHNOLOGY. It attempts to improve the appearance of text on flat-panel monitors by using the fact that every pix-el is made from three elements which can be con-trolled separately. ClearType takes advantage of the way your eyes perceive colour, using shades of blue, red and green to simulate higher screen resolution. This means that text resolution can be three times

25/34 pt

GREATER, BUT ONLY HORIZON-TALLY. THAT ALSO MEANS THAT in ClearType hinting, characters are not adjusted along the vertical axis, which effectively halves the amount of work. A single font of 256 charac-ters takes about 40 hours to hint.



960 UPM ÷ 8 PPEM = 120 UNITÉS = 1 PIXEL

fallback file sharing

46 pt

DO YOU

113 pt

read me?

101 pt

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

a b c d e f g h i j k l m n o p q r s t u v w x y z À Á Â Ã Ä Å

Ā Ă Ą Æ Ǽ Ç Ć Ĉ Ċ Č Ď Đ È É Ê Ë Ē Ĕ Ė Ę Ě Ĝ Ğ Ġ Ģ Ĥ Ħ Ì Í

Î Ï Ĩ Ī Ĭ İ Į Ĵ Ķ Ĺ Ľ Ł Ñ Ń Ň Ņ Ò Ó Ô Õ Ö Ø Ǿ Ō Ŏ Ő Œ Ŕ Ř Ŗ

Ś Ŝ Ş Š Ș Ť Ţ Ŧ Ù Ú Û Ü Ũ Ū Ŭ Ů Ű Ų Ẁ Ẃ Ŵ Ẅ Ỳ Ý Ŷ Ÿ

Ź Ž Ż Ŋ Ð Þ ã ä å ā ă à á â ą æ ǽ ć ĉ ċ č ç ď đ è é ê ë ē ĕ ė ę

ě ĝ ğ ġ ģ ĥ ħ ì í ï ĩ ī ĭ ĵ ķ ĺ ľ ł ń ň ñ ņ ò ó ô õ ö ō ŏ ő ø ǿ œ ŕ

ř ŗ ś ŝ ş š ș ß ť ŧ ù ú û ü ū ŭ ů ű ų ẁ ẃ ŵ ẅ ỳ ý ŷ ÿ ź ž ż ŋ ð þ

0 1 2 3 4 5 6 7 8 9 ¡ ! ¿ ? . , : ; . . . _ - – — / \ | ¦ ( ) [ ] { } ‘ ’ “ ”

‚ „ ' " ‹ › « » • + − ± × ÷ = ≠ ≈ < > ≤ ≥ € $ ¢ # g ﬁ ﬂ @ &

5/9 pt

WHAT DOES ALL THIS MEAN TO A TYPE DESIGNER? HINTING CAN IMPROVE THE RENDERING OF FONTS. BUT THOSE FONTS WILL BE RENDERED DIFFERENTLY BY DIFFERENT RASTERISERS on different platforms and often in different applications as well, (compare for example text in the Safari and Explorer browsers on the same Windows computer). If the designer's intention is consis-tent cross-platform rendering, the fonts also need to use consistent design across similar letters. It is clear that one day font hinting will finally become obsolete, but it is not clear when that day will come.

7/11 pt

THE MOST WIDELY USED OPERATING SYSTEM IN THE WORLD, WINDOWS XP (STILL 58.4 % MARKET SHARE, as of this writing), has ClearType turned off by default, so unhinted fonts typically do not display well at small sizes. Whether we like it or not, it looks like hinting will be around for quite some time. But if you like how fonts display on the Mac at small sizes, you can take that as proof that it is already possible to render text well without any hinting at all.

9/13 pt

HINTING, OR SCREEN OPTIMISING, IS THE PROCESS BY WHICH FONTS ARE ADJUSTED for maximum readability on computer monitors. I have been designing type since the early 1990s, and for as long as I can remember, type designers have been saying that hinting would soon be made obsolete by new advances in hardware and soft-ware. Now, almost 20 years later, hinting seems to be more relevant than ever. The problem is that typical modern fonts are not designed primarily for the 72–96 dpi resolution of computer screens, but for the much higher 1,200 + dpi resolution of print media. The letterforms are designed and stored as outlines, mathematically perfect lines and curves which look great at high resolutions, but can be distorted or even illegible when convert-ed into groups of pixels, the on-or-off dots that make up a computer's display.

Screen

146 pt

oriented type

70 pt

blue zones and overshoot

36 pt

5/9 pt

EVEN WHEN FONTS ARE HINTED OPTIMUM ONSCREEN RESULTS ARE STILL NOT GUARANTEED, AS DIFFERENT FONT TECHNOLOGIES APPROACH HINTING DIFFERENTLY. IN THE PostScript system most of the font scaling is handled not by the fonts, but by the rasteriser software, so fonts in PostScript format look often good with relatively simple hinting or no hinting at all. In the TrueType system, however, the rasteriser is controlled by sophisticated hinting instructions contained within the font software; without this information TrueType fonts do not display well onscreen. Both systems have advantages and disadvantages. One advantage of the PostScript system is that the 'intelligence' is concentrated in the rasteriser, so any improvement to the raster-iser immediately produces better rendering of all PostScript fonts. Even twenty year old fonts render nicely on the latest Mac.

7/11 pt

IN THE TRUETYPE SYSTEM, RASTERISER UP-DATES REQUIRE ALL FONTS TO BE UPDATED AS WELL FOR OPTIMAL RESULTS. THUS FONTS hinted for black and white or greyscale rendering will not work as well with Windows ClearType rasteriser. On the other hand, TrueType hinting provides direct, pixel-by-pixel control over the rasterising process, which PostScript hinting does not. A lot has been written about how Mac OS renders text compared to Windows. I will not go into details here, but the primary difference is

9/13 pt

THIS IS EXACTLY WHAT HINTING IS ABOUT PROGRAMMING INSTRUCTIONS THAT FINE-TUNE A FONT'S RASTERISATION, THE PROCESS BY WHICH ITS mathematically ideal outlines are mapped onto a monitor's pixels. Hinting can control the heights and widths of a font's uppercase and lowercase letters, the widths of its individual lines, the amount of white space around letters, the size at which uppercase letters start to use different stem-widths from lowercase letters, how the angle of italic characters changes to best fit the pixel grid, and many other extremely technical details, all on a pixel-by-pixel basis. If this sounds like a rather tedious, time-consuming activity, it is, even for type designers, who are accustomed to tedious, time-consuming activities.

Font Hinting & the Future of Responsive Typography

by NICK SHERMAN

February 22, 2013
Published in alistapart.com
Typography & Web Fonts
18 Comments

Font hinting has been the source of countless headaches for type designers and users. Meanwhile, some of the most fundamental and important elements of typography still can't be addressed with the web of today. Rather than being seen as a tedious chore whose demise will be celebrated, hinting might actually provide the essentials for truly responsive design, and vastly expand the possibilities of digital typography for designers, publishers, and readers.

THE FUNDAMENTALS OF HINTING

Type and web designers usually think of "hinting" as instructions built into digital fonts to improve their rendering on a grid of pixels. Hinting pushes the points of a font's Bézier curves around according to contextual conditions, such as the font's rendering size. Though it's now associated with type on screens, hinting was first used in the 1980s to improve rendering on low-resolution printers.

Thinking about it in these terms, hinting is responsive type that existed before the web. The font performs a media query of sorts to learn its size, then responds by repositioning points in each glyph according to built-in instructions, or "hints." In other words, hinting is to fonts what responsive layout is to websites. It allows a single font file to adapt to a variety of contexts, the same way CSS allows a single HTML file to adapt to a variety of contexts. In fact, Håkon W. Lie used the term "presentation hints" in 1994 to summarize his original proposal for CSS.

HATING HINTING

Developing hinting instructions can be extremely difficult, expensive, and time-consuming. Automated hinting tools have begun to ease some of this pain, but for smaller body type—often the most important type on any page—there's still no substitute for the quality that can be achieved through tedious

who deal with hinting today eagerly anticipate a future when they'll no longer need to worry about it. They optimistically cite advances in display resolutions and rendering software as sure signs that hinting will be obsolete within a few years. (Such claims have been made for the past twenty years, and will probably continue to be made for some time to come). But this ardent desire to see the end of hinting is based on a narrow understanding of what hinting can be.

I used to be among the impatient hinting haters, cursing Apple (yes, Apple—whose rendering engines now all but ignore hinting data) for popularizing the concept of hinted screen fonts with their TrueType spec in 1991. And I would be lying if I claimed that the issues currently related to hinting don't still cause me trouble. However, I'm very reluctant to dismiss the general concept of hinting just because current implementations of the idea are limited and hard to deal with.

PUSHING TOWARD MACRO-HINTING

The hinting that people know and use today only represents a small sliver of the possibilities for contextual typeface modification. Considering that it's impossible to query a font's absolute size—never mind things like font smoothing conditions or physical pixel density—many of the most basic fundamentals of typography, typeface design, and readability are still lost on the web. With better media query features in place, type-

FONTS ARE MADE OF VECTORS,
BUT COMPUTER SCREENS HAVE PIXELS.

glyphsapp.com/tutorials

Pixel Perfection

When fonts are displayed on the screen, the vectors are rasterized. This is done by laying out the vectors across the pixel grid. Pixels that fall completely on the inside of the vectors are turned **black,** pixels on the outside stay **white,** or whatever your foreground and background colors are. Pixels on the outline get some shade of **grey,** at least as long as anti-aliasing is applied. Hinting does have some effect, but the basic principle stays the same.

Now, the effective font size on the screen is determined by the PPM (pixels per em) value. This value, surprise, surprise, tells you how many pixels are used per em, duh. Glyphs measures all sizes in units. The UPM (units per em) value tells you how many units there are per em, double-duh.

So if we want to know how many units represent one pixel, we need to know the size at which the font is used on screen, i.e. the PPM, and the coordinate resolution of the font file, i.e. the UPM. Then we divide the UPM by the PPM, and we know the unit size of a pixel. It's that easy.

Make sure you don't get too far off the original 1000 value. Different applications seem to tolerate different UPMs. Experience shows it starts to get hairy at 3000 UPM.

EXAMPLE 1 we're going to use our font at a size of 20 screen pixels, thus the PPM = 20. The default UPM value is 1000. Now remember: UPM ÷ PPM = size of one pixel, so one pixel on the screen corresponds to 1000 ÷ 20 = 50 units in the glyph.

EXAMPLE 2 we're going to use our font at a size of 16 screen pixels, thus the PPM = 16. The default UPM value still is 1000, so we calculate: 1000 ÷ 16 = 62.5, duh. Not a good number. *What are we going to do now?* Fear not, let's first see what would be a better number. I suggest 60 instead of 62.5, because it's close enough and it is a nice round number. Now, open your Font Info and change the UPM value to (Size x PPM =) 60 x 16 = 960.

Typographic work in progress (3)

José Luis Acosta
para mí

16 de enero (hace 4 días)

Hola Tassiana.

Comienzo con dos datos, para mayor claridad. Hay dos variantes de fuentes OpenType: las que contienen curvas PostScript (extensión .otf) y las que tienen curvas TrueType (extensión .ttf); las curvas PostScript son de tercer grado (porque las ecuaciones que las describen son de tercer grado), mientras que las curvas TrueType son de segundo grado. La diferencia, en términos prácticos, es que para lograr un mismo dibujo suelen requerirse más nodos en TT que en PS. El segundo dato es que hay dos modelos de hinting, en correspondencia con los dos tipos de curvas que mencioné: el hinting TrueType, que es un lenguaje completo, y el hinting de PostScript, que es sólo un sistema de "guías" ortogonales que se asocian a los nodos para orientar al rasterizador en el momento de dibujar los signos en la resolución requerida. Una diferencia importante entre ambos modelos es que el hinting de TrueType te da mucho más control sobre el resultado final en cada PPM, pero el costo es una mayor dificultad de aplicación y más tiempo de trabajo; el hinting de PostScript es mucho más sen

- En las capturas de pantalla de Windows veo un problema en la fuente Micro a 10 pixeles: varios glifos que deben verse del mismo tamaño se dibujan con alturas diferentes. Claramente, es algo relacionado con las zonas de alineación. Al revisar la fuente en FontLab, pude ver que las dos zonas que tiene definidas están bien colocadas, pero los rasgos curvos no tienen compensación de tamaño respecto a los trazos rectos (la o tiene la misma altura que la x, en lugar de rebasarla un poco arriba y abajo), lo que inutiliza la zona en algunos PPM y puede causar problemas de alineación para ciertos rasterizadores. Entiendo que estabas tratando de ceñirte a una retícula, pero yo diría que una pequeña compensación de curvas no afectaría el desempeño de la fuente en pantalla —considerando que de hecho no todos los rasgos de los glifos caen exactamente en múltiplos de la retícula y aun así se dibujan bien en tamaños pequeños—, y sí ayudaría a que los rasgos curvos quedaran dentro de las zonas de alineación, lo que evitaría el problema que se observa en la Micro. Finalmente, el "aplanado" de esas curvas en PPM bajos lo controlan precisamente las zonas de alineación a la hora de rasterizar. No estaría de más hacer una versión de prueba de la Micro y ver si así se resuelve ese problema.

Es lo que he podido ver hasta ahora, espero que estas sugerencias ayuden.

Saludos.

Dessiné par Tassiana Nuñez Costa
Post-Diplôme Typographie & Langage
Session 2013–2015, ÉSAD Amiens