FONTE RESPONSIVA PARA A WEB

Dissertação de Mestrado em Design e Multimédia

Beatriz José Monteiro Diogo
Orientador: Artur Rebelo
Co-Orientador: Luís Lucas Pereira

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA 2016/2017

Fonte Responsiva para a Web

Dissertação de Mestrado em Design e Multimédia
Beatriz José Monteiro Diogo
Orientador: Artur Rebelo
Co-Orientador: Luís Lucas Pereira

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE COIMBRA 2016/2017

AGRADECIMENTOS

À minha família, pelo apoio e compreensão oferecidas e um especial obrigado à minha mãe, por apesar de não ter conhecimentos na área, ajudou muito na concretização deste documento.

Aos meus orientadores Artur Rebelo e Luis Lucas Pereira, pela disponibilidade oferecida durante todo o desenvolvimento e pelo apoio e entusiasmo que, desde início, manifestaram pela ideia desta dissertação.

A todos os meus amigos e às gomas, que ajudaram ao longo do desenvolvimento, que estiverem sempre disponíveis para desabafos, discutir ideias e dispensaram um pouco do seu tempo para analisar e comentar os produtos desenvolvidos. Um especial obrigado à Ângela Coelho, pelo enorme apoio e ajuda que ofereceu, desde do início da escrita desta dissertação.

Ao meu namorado, por todo o apoio, conselhos oferecidos e acima de tudo, paciência por me aturar quando o desenvolvimento desta dissertação corria menos bem.

RESUMO

A tipografia, ao longo da História, tem vindo a responder às necessidades da sociedade e às tecnologias que foram surgindo (Lupton, 2014).

Atualmente, a *web* é um meio de comunicação essencial na vida quotidiana, onde a tipografia é o seu principal constituinte (Reichenstein, 2006). Assim, torna-se cada vez mais importante poder transmitir um conceito ou uma ideia, através da tipografia e as formas tipográficas, em abordagens mais experimentais, aplicados na *web*.

Desde o século XIX, que a tipografia é usada de forma a que a largura da fonte se ajuste ao espaço a ocupar no cartaz, que como é o caso do desenho de tipos de letra condensados e expandidos para resolverem problemas de composição tipográfica ou abordagens mais contemporâneas em cartazes. Por outro lado, atualmente, para além da importância da tipografia na *web*, torna-se essencial que os seus conteúdos se adaptem ao dispositivo ou tamanho do ecrã onde estão a ser observados. Assim, pretendo realçar certos aspetos do conceito de responsividade e ajustar a largura de uma fonte à largura do ecrã.

A concretização prática desta dissertação consiste na criação de uma ferramenta que permita uma exploração alternativa da tipografia na *web*. Sendo a *web* um ambiente programático, esta ferramenta vai ser desenvolvida em formato script na linguagem de JavaScript — linguagem para a *web* que explora e trabalha a interatividade dos seus elementos.

Esta ferramenta pretende que um *web designer* possa tirar partido da tipografia, de forma a possibilitar uma maior experimentação e exploração da tipografia na *web*, onde esta se torna flexível, de forma a que a sua largura, se ajuste a diferentes áreas, tamanhos de ecrã e dispositivos.

Palavras-chave

Tipografia, *Web*, Responsivo, *Script*, Glifo, Variável, Flexível, Fonte, Java*Script*

ABSTRACT

Throughout history, typography has been responding to the needs of society and to the emerging technologies (Lupton, 2014).

Today, the web is an essential media in daily life, where typography is its main component (Reichenstein, 2006). Thus, it becomes increasingly important to be able to transmit a concept or an idea, through typography and typographic forms, into more experimental approaches, applied on the web.

Since the nineteenth century, typography has been used so that its width fits the space to fill the poster, that like the condensed and expanded fonts are used to solve typographic composition problems or more contemporary approaches in posters. On the other hand, today, in addition to the importance of typography on the web, it becomes important that its contents are adapted to the device and screen size where they are being watched. So, I want to take advantage of certain aspects of the concept of responsive design and make the font width fit the screen width.

The practical realization of this dissertation consists in the creation of a tool that allows an alternative exploration of typography in the web. Being the web, a programmatic environment, this tool will be developed in a script format, in the language of JavaScript - web language that explores and works the elements interactivity.

This tool will allow a web designer to take advantage of the typography in order to allow a greater experimentation and exploration of typography on the web, where the typography becomes flexible, so that its width, adjustable to different areas, screen sizes and devices.

Keywords

Typography, Web, Responsive, Script, Glyph, Variable, Flexible, Font, JavaScript

ÍNDICE

A. LISTA DE FIGURAS

B. LISTA DE VÍDEOS

Vídeo 1 : Animação do Logo Pequeno, reproduzida a partir do *script*.
https://vimeo.com/230799641

Vídeo 2 : Interação através o rato, da primeira secção da página *web*, que apresenta o *script*.
https://vimeo.com/230797270

Vídeo 3 : Vídeo da demonstração do *script*, apresentado na página *web*.
https://vimeo.com/230797700

Vídeo 4 : Testes da animação, aplicado em diferentes eventos (`window.ready` e `mouseover`).
https://vimeo.com/230798268

C. GLOSSÁRIO

Neste capítulo, serão apresentados alguns termos mais específicos sobre tipografia, web e responsividade. Estes termos serão usados nos capítulos seguintes, sendo nesta secção apresentados ao leitor.

AJAX (Asynchronous JavaScript and XML) — É uma técnica que permite fazer pedidos de dados ao servidor e carregá-los sem ter que recarregar a página inteira (Robbins, 2007).

Array — É um tipo especial de variável, onde em vez de guardar um valor pode guardar uma lista de valores (Duckett, 2014).

Browser — Software do cliente, que faz pedidos ao servidor de documentos e mostra-os ao utilizador que está a navegar na *web* (Duckett, 2011).

CANVAS — É uma área do ecrã que pode ser manipulada via Java*Script* (Fulton, S. and Fulton J. , 2011).

Caractere — Qualquer letra, numeral, pontuação ou qualquer outro símbolo incluído numa fonte, sendo que alguns caracteres podem ser representados por mais que um glifo (Monotype, 2017).

CSS (Cascading Style Sheets)—É usado para descrever a aparência da página *web*. O CSS também fornece métodos que controlam a forma como o documento aparenta em diferentes *media* (Robbins, 2007).

Design Responsivo — É a forma como é construído uma página *web* que facilmente pode ser visto e usado em qualquer tamanho de ecrã, desde os telemóveis mais pequenos até aos monitores de computadores maiores (Robbins, 2007).

Espacejamento — Ajuste global do espaço entre as letras. O aumento do espacejamento é usado para dar um ar mais leve ao texto de modo a facilitar a leitura (Lupton, 2006).

Estilo — Qualquer variante na família tipográfica, equivalente a uma fonte (Monotype, 2017).

Família tipográfica — É uma coleção de fontes que partilham algumas características de design e o nome. Super famílias são conjuntos de muitas fontes, com variações na largura e peso (Monotype, 2017).

Fonte — Coleção de letras, números, pontuação, e outros símbolos. Embora possa ser confundida com um tipo de letra, a fonte refere-se à forma física, quer esta seja em tipos móveis ou num ficheiro de fonte de computador. Já o tipo de letra refere-se ao design, como este é apresentado. A fonte é o que se usa e o tipo de letra o que se vê (Monotype, 2017).

Foundry — Empresa que desenha e distribui tipos de letra, ou seja, um fabricante de tipografia (Monotype, 2017).

Função — Agrupa uma série de afirmações de forma a executar uma determinada tarefa, evitando que, em diferentes locais num *script*, ocorram pedaços de código que efetuam a mesma tarefa (Duckett, 2014).

Glifo — Todos os caracteres numa fonte (ex: M, €,! e 9), são representados por um glifo. Uma tipografia pode conter mais do que um glifo para cada letra, referenciados de alternativos (Monotype, 2017).

HTML (HyperText Markup Language)— É uma linguagem usada para criar páginas *web*, usando elementos para descrever a estrutura de um documento. O HTML não é uma linguagem de programação, mas sim uma linguagem de marcação, ou seja, é um sistema que identifica e descreve vários componentes de um documento. Este tema será abordado com mais detalhe no capítulo do Estado da Arte (Robbins, 2007).

HTML5 — É a versão atual do HTML (Fulton, S. and Fulton J. , 2011).

JavaScript — É uma linguagem usada para adicionar interatividade e comportamentos a uma

página *web* (Robbins, 2007). O Java*Script* é uma linguagem geralmente usada para manipular elementos de uma página ou certas funções do browser (Robbins, 2007).

JQUERY — É um ficheiro *script* em Java*Script* que pode ser incluído em páginas *web*. Oferece formas simples de alcançar uma variedade de tarefas do Java*script*, de forma rápida e consistente em todos os browser (Duckett, 2014).

Kerning — Ajuste automático dos espaços entre diferentes combinações de letras, prevista nas tabelas da fonte digital, pois se o espaço das letras for uniforme, vão aparecer vazios em certos tipos de letra e assim o kerning remove espaço entre letras, de forma a melhorar a legibilidade (Lupton, 2006).

Largura da composição — Corresponde à largura da letra, mais um pequeno espaço que protege umas letras das outras, ilustrado na figura 3, na página 12 (set width) (Kane, 2011).

Largura da letra — Largura total que a letra ocupa, ilustrado na figura 2, na página 12 (Kane, 2011).

Ligaturas — Caracteres especiais que unem duas letras, usados em casos em que as duas letras estariam muito próximas, permitindo que as letras resultem melhor juntas, ilustrado na figura 4, na página 12 (Monotype, 2017).

Master — Conjunto de dados de uma fonte, que inclui os dados completos que a definem, usados no desenvolvimento de uma fonte. Por vezes, no desenvolvimento de fontes, são usadas várias *master*s como fonte de dados para criar diferentes variações dentro da família (Microsoft, 2017).

Multiple Master — extensão das fontes Post*Script* da Adobe Systems, que permite uma interpolação suave entre as diferentes *master*s, ao longo de diferentes eixos (peso, altura-x, largura). Assim, é possível gerar um grande número de variações, a partir de um ficheiro fonte (Lehni, 2011).

Objeto —Agrupam em si um conjunto de variáveis e funções de forma a criar um modelo de algo a representar (Duckett, 2014).

OFL —Open Font License, permite a qualquer utilizador alterar uma fonte e contribuir para a sua evolução (Lupton, 2009).

OpenType — Formato de fonte desenvolvido pela Adobe e a Microsoft. As vantagens do ficheiro de fonte OpenType são a estrutura de arquivo único que oferece compatibilidade entre plataformas e funcionalidades tipográficas avançadas, como ligaduras automáticas e glifos alternativos (Monotype, 2017).

OpenSource — Software desenvolvido num esforço coletivo com o intuito de tornar o código acessível a qualquer programador que o queira mudar. Estando usualmente acessíveis, sem custo associado (Robbins, 2007).

Peso — Um estilo específico ou iteração de um tipo de letra, referindo-se, especificamente, à largura do traço do tipo de letra. Tendo termos comuns como Thin, Light, Regular, Medium, Bold, Heavy, do menos largo para mais largo (Monotype, 2017).

Serifa — Pé oblíquo ou recto, no fim do traço (Kane, 2011).

Script — É uma série de instruções que um computador pode seguir para atingir um objetivo (Duckett, 2014).

Stroke/Traço — Qualquer linha que define a forma da letra, ilustrado na figura 1, na página 12 (Kane, 2011).

SVG (Scalable Vector Graphics) — Formato usado para mostrar imagens vetoriais diretamente na *web* (Robbins, 2007).

Tipo de letra — Interpretação artística de uma coleção de símbolos alfanuméricos. Um tipo de letra pode incluir letras, numerais, pontuação e outros símbolos. Está geralmente agrupado em conjunto numa família, contendo fontes individuais para itálicos,

variações de peso e largura (Monotype, 2017).

Variável — Elementos que um *script* usa temporariamente para guardar bits de dados (Duckett, 2014).

URL (Uniform Resource Locator) — Endereço que identificada cada documento *web* (Robbins, 2007).

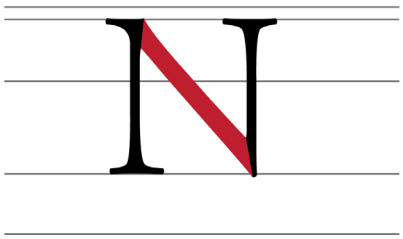


Figura 1 : Traço /*stroke*.

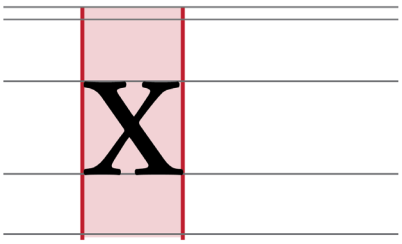


Figura 2 : Largura da letra.

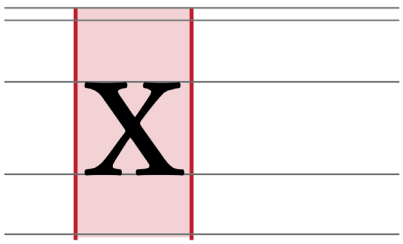


Figura 3 : Largura da composição.

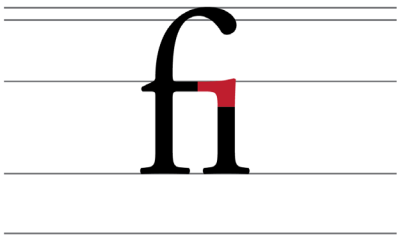


Figura 4 : Ligadura.

1. INTRODUÇÃO

Neste capítulo, é introduzido o tema desenvolvido nesta dissertação, abordando as motivações, as áreas abordadas, os objetivos, os contributos pretendidos e as metodologias que foram seguidas no desenvolvimento do projeto.

1.1 MOTIVAÇÃO

Esta dissertação — Fonte (cf. Glossário) Responsiva para a *Web* — é o resultado do meu interesse, ao longo da licenciatura em *Design* e Multimédia, em tipografia e programação, com uma especial incidência na *web*. A tipografia, desde o início dos meus estudos, fascinou-me pela sua complexidade, pela sua dimensão técnica, histórica e conceptual e pelo seu papel essencial no design; a programação, pelo desafio mental que implica; e por último, o facto de conseguir conciliar num projeto, as duas áreas que mais me despertaram o interesse, sendo que a programação serve de ferramenta, possibilitando novas explorações na tipografia.

Ao conceptualizar a ideia para esta dissertação, por se tratar de um projeto académico na Universidade, com recursos, tempo e professores que pudessem apoiar e ajudar na sua concretização, surgiu a oportunidade de poder desenvolver um trabalho com um carácter mais experimental. Deste modo, pretendo continuar a desenvolver as minhas capacidades em ambas as áreas, a tipografia e a *web*, introduzidas pela licenciatura embora, por outro lado, pretendo também desenvolver algo que ainda não foi abordado.

A *web* está em constante evolução, por isso, na minha opinião, é importante pensar e desenvolver ferramentas que facilitem a exploração da tipografia e as suas formas, neste meio. Assim, espera-se que esta dissertação permita criar uma ferramenta para *web designer*s e *developers*, que possibilite uma maior flexibilidade e maleabilidade das formas tipográficas, na *web*. Por outro lado, pretende-se abrir uma reflexão sobre a importância da tipografia na *web* e o papel essencial de ferramentas que possibilitem uma maior exploração da tipografia, de forma a poder permitir desenvolver, mais facilmente, projetos na *web*, onde a tipografia é o principal elemento expressivo da composição.

1.2 ENQUADRAMENTO

Os temas abordados nesta dissertação são a tipografia na *web*, onde a programação atua como uma ferramenta potenciadora de expressividade tipográfica. Assim, esta dissertação estará direcionada para uma união da disciplina do *design* e da informática.

Uma das áreas abordadas nesta dissertação é a tipografia que é definida como a forma, como o material escrito é organizado e preparado para a impressão ou a arte, ofício ou processo de compor tipos e imprimir a partir dele (Collin English Dictionary).

A outra área abordada nesta dissertação está relacionada com o termo responsivo ou responsividade na *web*, um conceito que se refere ao ajuste da organização de elementos numa página *web* entre diferentes tamanhos de ecrã, evitando que seja necessário desenhar uma página diferente para cada tamanho.

A construção de *website*s tem evoluído ao longo dos anos. Inicialmente, uma página era desenhada para tamanho fixo, atualmente, com o aparecimento de uma grande variedade de tamanhos de ecrã e dispositivos, uma página é planeada de forma a que os seus elementos se ajustem a todos os tamanhos de ecrã.

1.3 ÂMBITO

Os resultados práticos, desenvolvidos no âmbito desta dissertação, baseiam-se em duas áreas, a tipografia e o *web* design. Na área da tipografia, através da análise e estudo de casos de fontes expandidas, desenvolveu-se uma nova fonte expandida, a partir de soluções que se ajustavam mais ao problema proposto. Esta fonte é *opensource* (cf. Glossário), ou seja, está disponível em domínio público, livre para uso e alteração por estudantes e profissionais.

Na área da *web*, pretendeu-se desenvolver um *script* que consiga ler e interpretar os dados de cada fonte de forma a calcular e redesenhar a largura da letra (cf. Glossário), para esta ocupar a largura do dispositivo onde está a ser observada, através de dois ficheiros fonte, um com uma largura normal, outra com uma largura extra larga. O *script* é capaz de ler e interpretar a informação de qualquer ficheiro fonte e, com essa informação, alterar a forma da fonte e desenhá-la. Por último, a sua construção foi elaborada de forma a que a sua posterior implementação por *developers* seja simples.

1.4 OBJETIVOS

Para realizar esta dissertação com sucesso, considera-se importante cumprir os seguintes objetivos gerais:

— Investigar a evolução das formas tipográficas e a forma como a tecnologia e o seu contexto histórico as influenciaram;

— Compreender o processo de construção, desenvolvendo uma fonte e identificando os comportamentos anatómicos que um glifo (cf. Glossário) deve apresentar quando esticado e prever todas as suas iterações intermédias;

— Construir um *script* na linguagem Javascript (cf. Glossário), que permite ajustar a largura base da fonte ao tamanho do dispositivo onde está a ser observada, com uma implementação fácil e intuitiva;

— Construir uma página *web* de apresentação do *script*, que mostre as diferentes possibilidades deste *script* e um guia de como implementá-lo.

— Com a divulgação do *script*, pretende-se que este atue como uma ferramenta para potenciar e facilitar, a *web designer*s e *developers*, uma maior experimentação das formas tipográficas, de qualquer fonte na *web*.

1.5 METODOLOGIAS

O desenvolvimento do projeto irá consistir em duas fases:

Na primeira fase, são analisadas as duas áreas de interesse: a Tipografia, o seu estado da arte, que inclui um estudo aprofundado da anatomia dos tipos de letra e a área da responsividade na *web*, que é introduzida em paralelo com a história da tipografia. Esta fase também vai envolver um estudo sobre as tecnologias utilizadas para a manipulação de fontes na *web* e apresentação de resultados.

Numa segunda fase, com base na informação recolhida sobre o estado da arte e em trabalhos relacionados, foi desenvolvido o *script* que vai atuar numa fonte *opensource*, expandindo-a e adaptando-a ao ecrã. Foi também implementado o site de apresentação ao *script*, a imagem para a divulgação do mesmo através de redes sociais e, para finalizar, o desenvolvimento da dissertação escrita.

1.6 CONTRIBUTOS ESPERADOS

Com esta dissertação, pretende-se contribuir em diversas áreas, realçando as possibilidades da programação aplicada à tipografia. Na área da tipografia, o trabalho pretende contribuir com um estudo teórico e levantamento de projetos, onde a programação é aplicada à tipografia e desenvolvida uma fonte com uma largura extra larga.

Na área da *web*, com o contributo prático desta dissertação, criou-se o *script*, que atua como ferramenta para *web designer*s e *developers*, que permite editar parâmetros específicos de uma fonte, em tempo real numa página *web*, de forma a que a largura da fonte se ajuste automaticamente ao tamanho do ecrã.

1.7 ESTRUTURA DO DOCUMENTO

De forma a explicitar todo o processo e objetivos desta dissertação, este documento seguirá a seguinte estrutura:

Estado da Arte

Capítulo onde se procede à recolha, análise e estudo da área da tipografia e responsividade na *web*, a sua importância e como a tecnologia influenciou as suas formas. Também são apresentados alguns trabalhos relacionados com a tipografia e projetos de programação aplicados na tipografia, que realçam aspetos relevantes para o projeto em desenvolvimento.

Abordagem Metodológica

Capítulo onde serão definidos os aspetos mais formais da dissertação, os seus objetivos e de que forma se pretende atingi-los. Explicita-se, ainda, os problemas que se pretendem solucionar com o desenvolvimento da dissertação e os contributos esperados nas respetivas áreas de foco. Por último, é apresentado o plano de trabalhos que se seguirá, para cumprir os objetivos propostos.

Fonte Responsiva para a Web

Capítulo onde será apresentado e explicado o processo de desenvolvimento do projeto prático, enumerando e desenvolvendo cada etapa; serão explicadas decisões tomadas e por último, serão apresentadas os vários resultados práticos desenvolvidos.

Reflexão

Neste capítulo, será apresentado os resultados obtidos, serão apresentadas as dificuldades encontradas e as perspetivas futuras para o projeto prático discutido nesta dissertação.

2. ESTADO DA ARTE

Neste capítulo, é apresentada uma síntese histórica da tipografia, antes e depois do aparecimento do computador. São apresentados alguns casos ao longo da história, que refletem as necessidades da altura de criar novas soluções para a tipografia, dada uma certa ferramenta ou forma de encarar a tipografia. Com o surgir do computador, são apresentados casos de estudo onde a programação foi usada como auxílio, de alguma forma, para criar ou gerar uma fonte.

Numa segunda parte do capítulo, são introduzidos os termos básicos da *web*, a sua história e a forma como o seu *design* tem evoluído com o aparecimento de novos dispositivos, com diferentes tamanhos, onde é possível visualizar páginas *web*.

2.1 Tipografia

Em relação à história da tipografia, começa-se a detalhar mais a tipografia a partir do século XIX, com a revolução industrial. Isto deve-se à deformação drástica que as formas tipográficas sofreram com a revolução industrial, de forma a ocuparem o maior espaço possível e a causarem um maior impacto. O capítulo é concluído com a apresentação de projetos tecnológicos e linguagens computacionais que são usados no design.

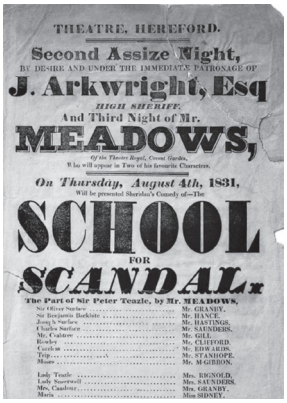

THEATRE, HEREFORD.
Second Assize Night,
J. Arkwright, Esq
HIGH SHERIFF.
And Third Night of Mr.
MEADOWS,
On Thursday, August 4th, 1831,
SCHOOL
FOR
SCANDAL.
The Part of Sir Peter Teazle, by Mr. MEADOWS.

Figura 5 : Poster de 1831 (Kane, 2011).

2.1.1 REVOLUÇÃO INDUSTRIAL

No início do século XIX, depois da invenção da máquina a vapor, muitas das atividades, até à altura feitas manualmente, começaram a ser realizadas com o auxílio de máquinas, entre elas a impressão e fundição de tipos. Assim, a revolução industrial gerou uma grande mudança na comunicação da tipografia. Antes do século XIX, a propagação era feita essencialmente através de livros, mas a partir da revolução industrial, o consumo e a produção em massa desencadeou um aumento significativo da propaganda. Assim, foi necessário a criação de novas abordagens tipográficas, mais expressivas, de forma a causar um maior impacto visual (Meggs & Purvis, 2011).

Com esta necessidade de novas formas tipográficas, surgiram fontes extra pesadas, extra largas, extra condensadas, onde a fonte era usada consoante o espaço que ocupava na folha, deformando as fontes tipográficas clássicas, como pode ser observado nas Figuras 5 e 6 (Lupton, 2006).


FULL MOON.
ST. MICHAEL'S
TEMPERANCE BAND!
Prof. V. Yeager, Leader, will give a
GRAND
MOONLIGHT
EXCURSION
On the Steamer
BELLE!
To Osbrook and Watch Hill,
On Saturday Evening, July 17th,
Leaving Wharf at 7½ o'clock. Returning
at 10½ o'clock. Kenneth will be at Osbrook.
TICKETS, - FORTY CENTS.

Figura 6 : Cartaz Tipográfico de 1875 (Lupton, 2006).

Com as novas formas dos glifos, maiores e mais largas, foi necessário adaptar os métodos de impressão. Até esta altura, os glifos eram fundidos em metal, sendo que eram moles demais para manter a forma em tamanhos maiores. Assim surgiram glifos talhados em madeira, mais baratos e onde, mais facilmente, se criavam glifos gigantes (Lupton, 2006; Ulrich, 2014).

Durante a década de 1840, tornou-se prática comum, para cada *design,* desenvolver conjuntos de várias larguras e estilos(cf. Glossário) para cada tipo de letra (cf. Glossário), como se pode verificar na figura 7. É de notar que sempre que a largura da letra era aumentada, também o seu peso (cf. Glossário) era aumentado, assim os tipos *bold* eram *extended*, e os tipos *light* eram *condensed* (Ulrich, 2014).


RRRRR
RRRRRRRR
RRRRRRR

Figura 7 : Revisão de Rob Roy Kelly's "American wood type: 1828–1900" de 1977, mostra variações das larguras dos tipos de madeira de 1880s (Ulrich, 2014).

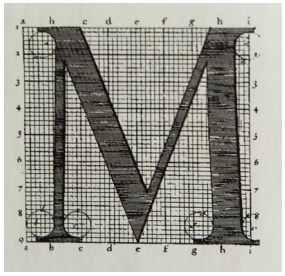


Figura 8 : Letras modelo para a gráfica de Luis XIV de Louis Simonneau (Lupton, 2006).

2.1.2 REFORMA E REVOLUÇÃO

Durante o século XIX, a indústria de impressão falhou em reconhecer as mudanças que vinham a acontecer na sociedade e consequentemente na natureza do que era impresso. O rápido crescimento da industrialização e produção em massa exigiu novas formas de impressão, onde houvesse um controlo mais eficiente do processo de produção e distribuição e, com o aumento da produção e competição, em que a publicidade fosse um meio para criar e estimular a procura de produtos (Spencer, 2004).

Assim, a tipografia moderna surgiu como resposta às novas exigências e oportunidades oferecidas pelo século XIX, criando formas mais geométricas e indo contra as formas humanistas(Figura 8) e grotescas(Figuras 5 e 6). A tipografia moderna apareceu de forma bruta, refletindo a violência com que os novos conceitos, nos vários campos da arte e do *design,* estavam a ignorar as convenções históricas e a desafiar as suas atitudes (Spencer, 2004).

O grupo De Stijl, na Holanda, reduziu o alfabeto a elementos perpendiculares, na Bauhaus foram construidos alfabetos através de formas geométricas básicas (Lupton, 2006). Depois de 1925, Herbert Bayer e Tschichold, encorajaram o uso de fontes sem serifa, e ambos criaram fontes a partir de formas geométricas básicas, como pode ser observado na figura 9, a fonte Universal de Herbert Bayer. Os tipos sem serifas refletiram a noção de "*beauty in utility*", onde as experiências da Bauhaus se centraram(Spencer, 2004; Hillner, 2009).

Figura 9 : Fonte Universal de Herbert Bayer (Spencer, 2004).

Em meados dos anos de 1920, um desconhecido *designer* de tipos italiano desenhou o tipo de letra Fregio Mecano, representada na figura 10. Todas as letras do alfabeto, e todos os numerais são formados apenas usando combinações dos 20 módulos, mostrados na figura 11 (“Nebiolo Specimen,” 1954).

Os segmentos são unidos para criar uma letra e é possível alterar na sua aparência aspetos da letra como o peso, a largura, a altura e as serifas, simplesmente adicionando ou retirando módulos, como é possível observar nas figuras 12 e 13. Como a letra é facilmente alterada, a sua forma pode tomar proporções extremas, onde letras podem ser criadas para ocupar qualquer tipo de espaço (“Nebiolo Specimen,” 1954).

A Futura, desenhada por Paul Renner, foi a primeira fonte geométrica

Figura 10 : Publicação da Fonte Fregio Mecano (“Nebiolo Specimen,” 1954)

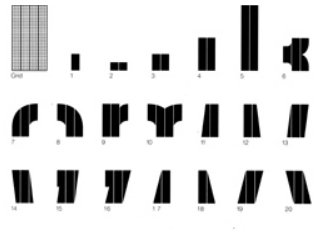


Figura 11 : Módulos usados para a construção de Fregio Mecano(“Nebiolo Specimen,” 1954).

Figura 12 : Exemplificação de fonte sem e com serifas apenas acrescentando mais módulos, na fonte Fregio Mecano (“Nebiolo Specimen,” 1954).

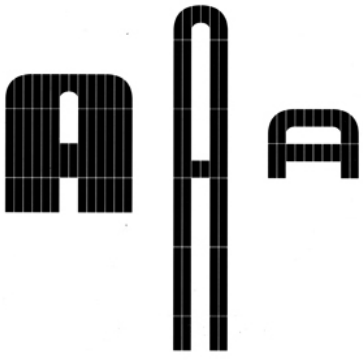


Figura 13 : Exemplificação de como acrescentar módulos na largura e altura, na fonte Fregio Mecano (“Nebiolo Specimen,” 1954).

Figura 14 : Fonte geométrica Futura desenhada por Paul Renner (Kane, 2011).

Figura 15 : Fonte Univers desenhada por Adrian Frutiger (Kane, 2011).

desenhada para ser aplicada em texto. Renner desenhou um número de glifos de caixa baixa e numerais para a Futura, como é mostrado a vermelho, na figura 14 (Kane, 2011).

2.1.3 PÓS II GUERRA MUNDIAL

Após a segunda guerra mundial, os *designer*s modernistas entenderam as ideias fundamentalistas no seu sentido mais pragmático e convencional. Para eles, a tipografia tinha o objetivo de transmitir a informação o mais rápido e eficaz possível. Para enfatizar a clareza visual e a eficácia na comunicação, eles rejeitaram a expressão emotiva, onde a estética e o prazer eram considerados secundários (Hillner, 2009).

Em meados do século XX, a fotocomposição veio substituir os tipos de metal, permitindo assim uma maior liberdade na sua composição, que antes era limitada por restrições técnicas. A fotocomposição tinha as suas desvantagens, que foram finalmente resolvidas com a introdução da publicação electrónica nos anos 1980 (Kane, 2011).

Um *designer* que teve um grande impacto na ideia de famílias tipográficas (cf. Glossário) foi Lucas Groot com a sua fonte Thesis e uma teoria para calcular pesos intermédios. Esta teoria também levou à conclusão de que os traços no eixo vertical devem crescer mais devagar que os traços do eixo horizontal (Ulrich, 2014).

Adrian Frutiger criou a fonte — Univers, que pode ser observada na figura 15, num esforço de eliminar a confusão crescente nas terminologias dos tipos (*thin*/*light*, *regular*/*medium*, *black*/*bold*), usando números em vez de nomes para descrever a palete de pesos e larguras (Kane, 2011).

2.1.4 TERMO GRID/GRELHAS

Após a segunda Guerra Mundial, os *designer*s suíços refinaram as ideias da Nova Tipografia[1], sendo que foi nesta altura que o termo grid começou a ter um uso geral para a construção de composições. Os praticantes e teóricos, desta época, como Max Bill, Karl Gerstner, Josef Müller-Brockmann e Emil Ruder, rejeitaram clichês artísticos de auto expressão e de intuição pura, aspirando a uma beleza controlada (Lupton, 2006).

A grelha divide os espaço a ocupar pela composição, em áreas mais pequenas, criando pequenos espaços entre estas áreas, para os seus conteúdos não se sobreponham. Com o espaço dividido, é possível ordenar e organizar melhor os vários elementos na página. Assim, com uma grelha tipográfica, o processo de incorporação de todos os elementos numa composição é facilitado. Os espaços predefinidos pela áreas da grelha são ocupados e é criada uma ideia de clareza e credibi-

[1]- Movimento de Design entre 1920s e 1930s, que se baseou nas ideias da Bauhaus e no construtivismo Russo. Este movimento rejeitou decorações e defendeu um design racional e funcional. Assim, eram, usualmente, aplicadas fontes não serifadas e cores puras aos cartazes (Meggs & Purvis, 2011).

lidade da informação. As grelhas tipográficas também podem ser aplicadas ao criar identidades visuais para empresas, englobando todos os meios visuais produzidos para divulgá-las (Müller-Brockmann, 2016).

Karl Gerstner escreveu um livro, Designing Programmes (1964), onde defendeu o *design* sistemático e apresentou vários exemplos em que as regras do "programa" de *design* eram definidas para acomodar várias soluções para vários corpos (Lupton, 2006). Gerstner definiu um "programa" de *design* como um conjunto de regras para construir uma gama de soluções visuais, sendo que com essas regras é possível criar várias soluções visuais válidas (Lupton, 2006).

Dois dos exemplos abordados por Karl Gerstner foi o Boîte Musique e Holzäpfel, onde em ambos os casos defende que uma marca não deve ser inalterável e em que os elementos permaneçam estáticos em qualquer situação, mas sim que os elementos devem ser adaptados a todas as proporções possíveis, constituindo a assinatura e o estilo próprio nessas variações (Gerstner, 1964). Na figura 16, é exibida a estrutura da identidade da Boîte Musique, sendo que a inscrição e a moldura são elementos fixos, assim como as conexões entre eles. A partir do canto inferior direito, a moldura pode ser aumentada para cima e para a esquerda. A figura 17 demonstra a capacidade flexível da marca aplicada aos Cartões "New Year's" para a Boîte Musique, com variações adequadas a diferentes proporções (Gerstner, 1964). A figuras 18 mostra uma porção do sistema para a identidade de Holzäpfel, sendo que a espessura da linha é a mesma em todas as variações, mas o tamanho e a proporção mudam (Gerstner, 1964).

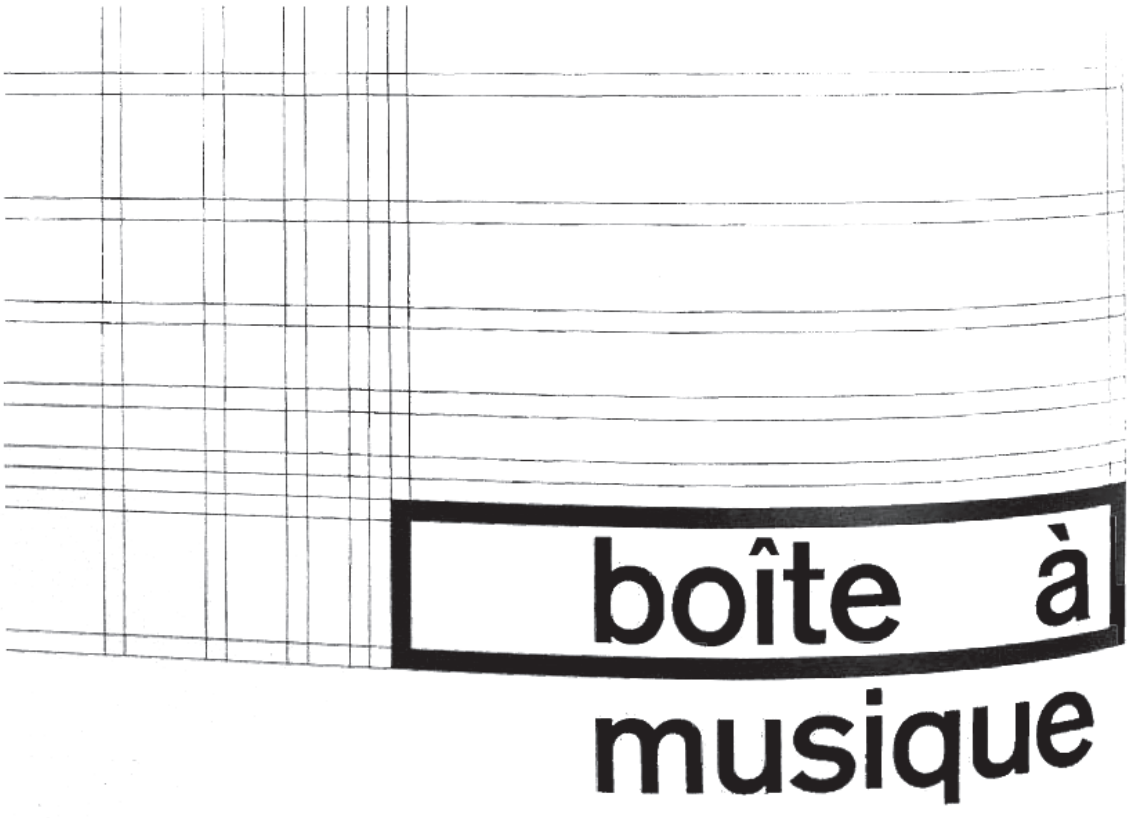


Figura 16 : Estrutura identidade da "Boite à musique" (Gerstner, 1964).

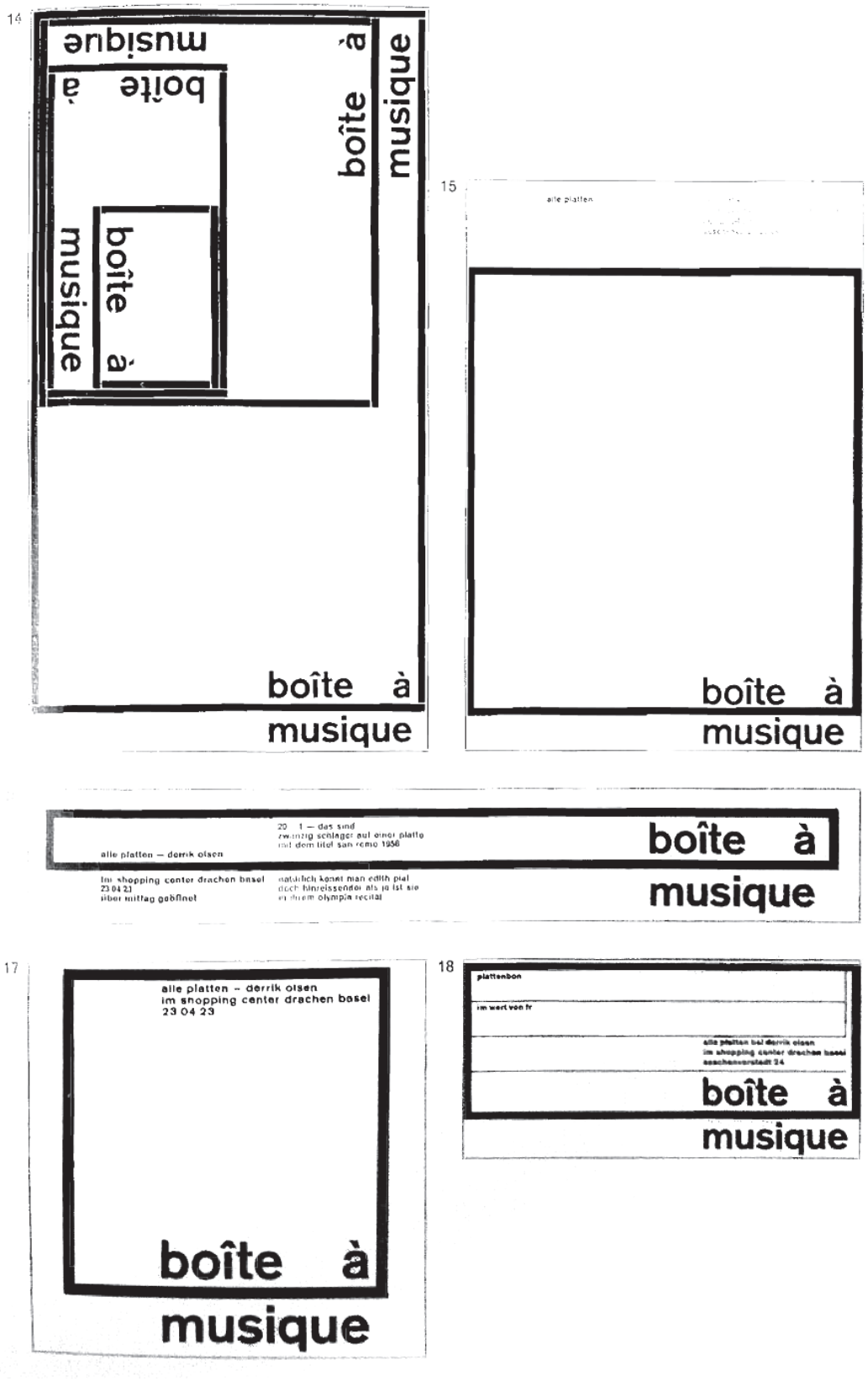


Figura 17 : Cartões para o Ano Novo, com variações adequadas a várias proporções (Gerstner, 1964).

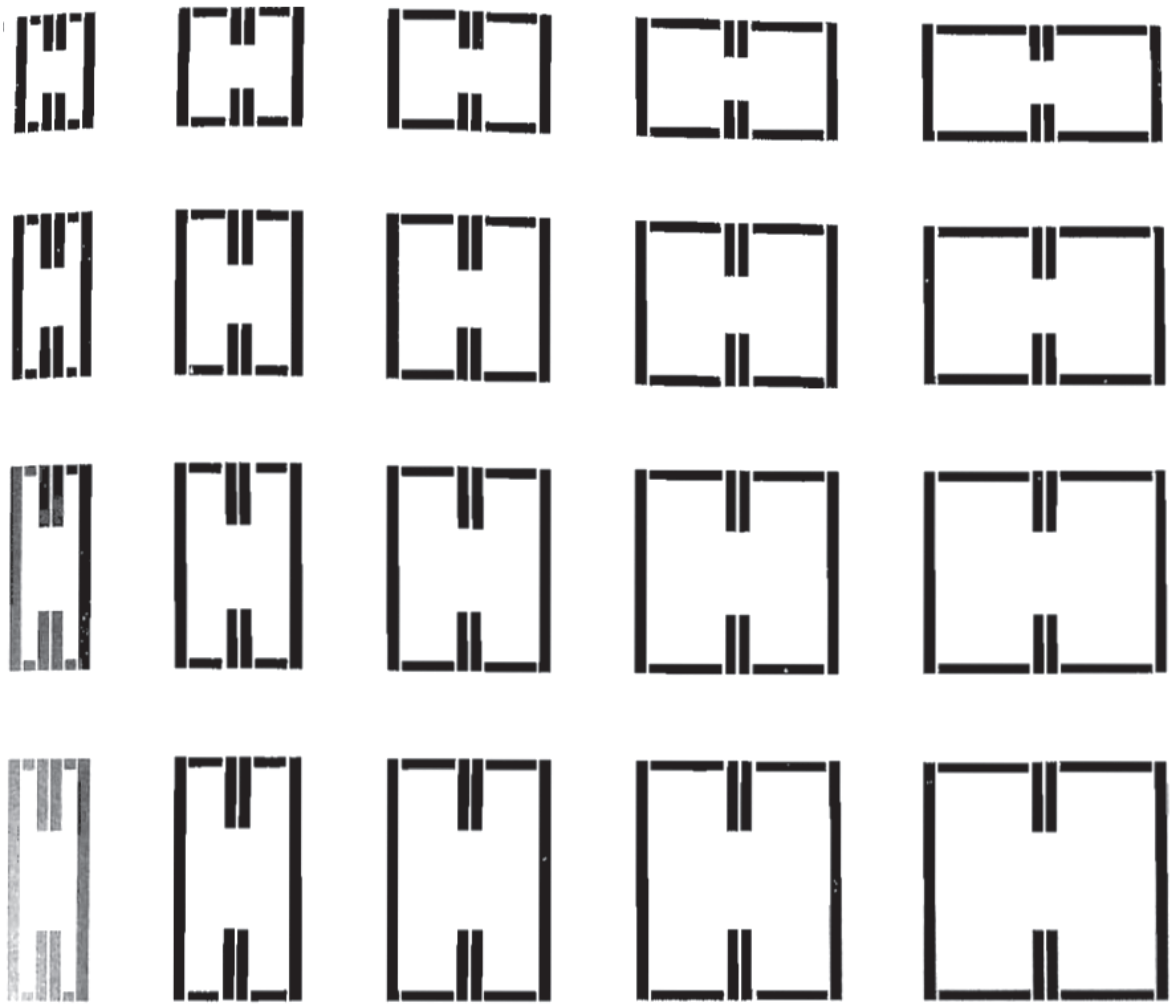


Figura 18 : Porção do sistema Holzäpfel (Gerstner, 1964).

À medida que o trabalho se torna mais complexo, é preciso uma cooperação mais intensificada entre os participantes. O esforço e o tempo dedicado ao desenvolvimento da estrutura compensa, pois facilita na construção de trabalhos pequenos (Gerstner, 1964).

Karl Gerstner esclarece a questão do uso da variabilidade que pode comprometer a marca de *design,* no qual responde que a marca não é uma questão de proporções em que se enquadra, sendo que as proporções são algo nem bom nem mau para a construção de uma marca (Gerstner, 1964).

Este pensamento programático de Karl Gerstner tem vindo a ressuscitar à medida que os *designer*s precisam de lidar com projetos com grandes quantidades de informação. Sendo que, cada vez mais, é exigido que sejam criados "programas" de *design* flexíveis, capazes de ajustar os seus conteúdos a corpos dinâmicos (Gerstner, 1964 e Lupton, 2006).

[2] - Movimento de design que surgiu na Suíça, durante os anos de 1950s. Defendeu o uso de fontes não serifadas, por refletirem melhor um espírito progressivo e o uso de grelhas, por ser a forma mais harmoniosa de organizar a informação. Assim, a sua imagem é caracterizada essencialmente, pela construção e organização matemática de elementos numa grelha, numa imagem clara e pelo uso de fontes não serifadas (Meggs & Purvis, 2011).

2.1.5 ERA DIGITAL

A era digital, no *design,* foi estabelecida em 1983, com a introdução do Apple Lisa, o primeiro computador de secretária com "graphical user interface" (GUI). A existência de uma interface veio permitir um uso mais intuitivo por parte dos utilizadores (Hillner, 2009).

Com a ascensão do computador pessoal e de impressoras de baixa resolução, ocorreram mudanças fundamentais. Ferramentas tipográficas, de *design* e ilustração, começaram a estar disponíveis para um público mais amplo, onde mais utilizadores começaram a tentar dominar mais especialidades (Lupton, 2006).

Em 1981, a Bitstream, Inc, começou a oferecer fontes digitais, que pouco depois a Adobe Systems seguiu o exemplo. Em 1990, grande parte das *Type Foundrys* (cf. Glossário) ofereciam versões digitais das suas fontes e várias lojas começaram a abrir, onde ofereciam exclusivamente fontes digitais (Kane, 2011).

Esteticamente, os primeiros projetos artísticos digitais foram considerados um retrocesso, devido às limitações que os computadores com pouca resolução e memória impunham. Embora a Apple tenha licenciado a *Adobe's PostScript* em 1985, um código que permitia uma conversão eficiente das fontes a partir de vetores, só em 1989 é que o *PostScript* foi distribuído internacionalmente. Assim, os *designer*s tinham de depender de impressoras matriciais que não ofereciam o melhor resultado (Hillner, 2009).

Figura 19 : Fontes Matrix, Emperor, Oakland e Emigre desenhada por Zuzana Licko (Hillner, 2009).

Em 1985, Zuzana Licko começou a projetar fontes que exploravam a textura grosseira dos sistemas da época, criando fontes de baixa resolução, como pode ser observado na figura 19. Enquanto outras fontes digitais tentavam adequar as formas tipográficas tradicionais aos diagramas e monitores da altura, Zuzana Licko quis tirar partido da linguagem própria do equipamento digital (Lupton, 2006).

2.1.6 PÓS-MODERNISMO

No início dos anos 1990, com a adoção da impressora a laser de alta resolução e das tecnologias de desenho por contornos como as da *Adobe PostScript*, os *designer*s deixaram de estar limitados pelos dispositivos de baixa resolução. Assim, a tipografia desenhada a partir do computador era tecnicamente perfeita. Os *designer*s pós-modernistas criaram rutura com os princípios estabelecidos pelo modernismo e pelo estilo internacional[2], expressando uma atitude rebelde em relação à grelha tipográfica e privilegiando a expressão própria. Os *designer*s queriam combater essa perfeição técnica, pois queriam expressar a individualidade e a diversidade, que a tipografia gerada por computador não conseguia alcançar (Lupton, 2006 e Hillner, 2009).

2.1.7 PROGRAMAÇÃO NA TIPOGRAFIA

Como foi indicado anteriormente, nos anos 1990 os *designer*s reagiram contra a perfeição obtida pelos computadores, assim quiseram personalizar e humanizar a perfeição formal da tipografia através da introdução de movimentos, padrões e irregularidades. A programação permite que seja introduzida uma certa aleatoriedade controlando a forma, sem que esta perca a sua integridade estrutural, através da alteração de variáveis e condições. A programação permite automatizar processos básicos, que permitem aos *designer*s produzir iterações rápidas, que seriam mais demoradas se fossem feitas manualmente (Lupton, 2014).

Os *type designer*s holandeses, Erik van Blokland e Just van Rossum, desenvolveram tipos de letra que contêm propriedades incluídas no ficheiro *font*, como: a aleatoriedade, a mudança e a incerteza, conseguidas através da combinação do papel do programador e do *designer* (Lupton, 2006). A fonte Beowolf, ilustrada na figura 20, foi a primeira de uma série de fontes, cujos contornos sofreram um ruído aleatório, sendo que este comportamento foi programado. O código usado para a Beowolf move os pontos aleatoriamente ao longo do contorno, isto faz com que o glifo tenha uma forma diferente sempre que é usado. A Beowolf foi criada de forma a reagir contra a perfeição técnica atingida através das tecnologias computacionais (Lupton, 2006; Hillner, 2009).

Figura 20 : Erik van Blokland e Just van Rossum criam a fonte FF Beowolf em 1990 (MoMa, 2017).

A programação já não é um domínio exclusivo de engenheiros informáticos, muitos *designer*s estão a escrever os seus próprios códigos de forma a potencializarem ou a evitarem as definições de programas padrão. Alguns *designer*s estão a criar novos programas e ferramentas, enquanto outros usam a programação no seu processo de *design* para introduzirem possibilidades visuais inesperadas (Lupton, 2014).

Embora os anos 1990, sejam lembrados por imagens de decadência, os tipógrafos continuaram a desenvolver fontes de uso comum, que eram projetadas para o uso em texto longo. Estas fontes forneceram aos *designer*s paletes flexíveis das letras, organizadas em famílias maiores. Em alguns casos, foram desenhadas famílias que para além de terem uma extensão de pesos (*light/ regular/ bold/ black*), também incorporaram versões com e sem serifas. Um dos exemplos é a família Rotis representada na figura 21, desenhada por Aicher em 1989, que possui quatro variações de serifas (Lupton, 2006; Kane, 2011).

Figura 21 : Família Rotis por Otl Aicher em 1989 (Kane, 2011).

2.1.8 TIPOGRAFIA MUTÁVEL

As fontes mutáveis, são fontes que contém mais do que um desenho para cada glifo, oferecendo ao *designer* diferentes opções de representar a mesma palavra. Embora as fontes mutáveis não sejam exclusivas para o ecrã, o conceito de variabilidade aponta para a possível animação. Alguns tipos mutáveis contêm uma Coleção de ligaturas (cf. Glossário) detalhadas, ou caracteres (cf. Glossário) alternativos, outras oferecem ornamentos não alfabéticos com a fonte (Lupton, 2014).

Walker é uma tipografia desenhada para o "Walker Art Center" por Matthew Carter, e é um bom exemplo das possibilidades que vêm com as fontes mutáveis e da expansão que trazem à tipografia. Walker é uma fonte mutável, onde é possível incorporar diferentes estilos e pesos de serifas à fonte, como se pode observar na figura 22, produzindo vários estilos sem que a fonte perca uma aparência unificada (Lupton, 2014). Assim Walker, como fonte mutável, permitiu que o "Walker Art Center" adequasse a variação da fonte a usar consoante a mensagem que pretendia passar sem que a marca perdesse a sua identidade (Meggs & Purvis, 2011; Lupton, 2014).

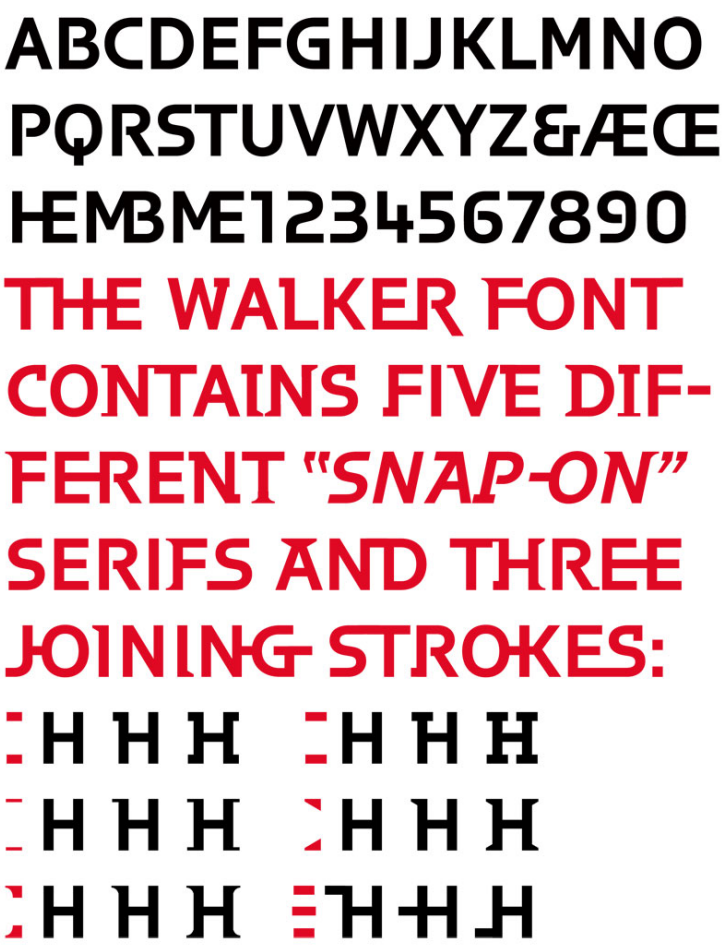


Figura 22 : Representação da Fonte Walker, e suas vários estilos de serifas (Walker Art Center, 2017).

2.2 WEB

Em relação à *web*, começa-se por apresentar os conceitos básicos, explicando os seus constituintes e como contribuem para a página. É abordada, com maior incidência, a evolução do *design* das páginas *web*, como a incorporação de fontes tem evoluindo e como a introdução de novos dispositivos, com tamanhos diferentes, influenciou a forma como eram construídas as páginas. Por último, são abordadas as fontes variáveis e as suas potencialidades ao serem aplicadas na *web*.

2.2.1 HTML

Nos anos 1990, a *web* consistia em páginas de especificações, todas no mesmo ficheiro. Assim, foi criado o HTML (cf. Glossário), que descreve a estrutura de uma página, de forma a lidar com muita informação, oferecendo meios de conectar vários documentos entre si (McLaughlin, 2011).

O HTML não é uma linguagem de programação, mas uma linguagem de marcação, ou seja, contém um sistema para identificar e descrever os seus vários componentes num documento, como parágrafos, títulos, imagens, listas, etc (Peterson, 2014). Cada um destes elementos pode ser inserido com uma etiqueta que abre e fecha o elemento, como é mostrado a seguir:

```
<p> </p>
```

Um *web designer* tem, geralmente, de implementar três linguagens ao desenvolver uma página *web*. Como foi referido, o HTML vai definir a estrutura e conteúdo de uma página, o CSS (cf. Glossário) indica como o HTML é apresentado na página, alterando a sua aparência, por último o JavaScript, permite alterar como a página se comporta, adicionando interatividade à página (Duckett, 2011).

Um *script*, é um sequência de pequenas instruções que um computador segue, para atingir os seus objetivos. Na *web*, são usadas *script*s na linguagem de JavaScript, onde o *browser*(cf. Glossário) consegue interpretar as suas instruções, de forma a adicionar interatividade e a aumentar a usabilidade da página (Duckett, 2011).

2.2.2 TIPOGRAFIA NA WEB

No início da década de 2000, os *web designer*s tinham de trabalhar com uma variedade limitada de fontes, aquelas que vinham instaladas com o *browser*, de lidar com problemas como falta de fontes que tinham uma boa aparência nos diferentes *browsers* e com a baixa resolução dos ecrãs e ainda assim criar uma página *web* com uma boa aplicação tipográfica (Reichenstein, 2006; MacDonald, 2014). Mas, como é reconhecido pelos *designer*s, atualmente, a tipografia representa um papel muito importante na construção de uma página *web*. Assim, a diversidade tipográfica na *web* sempre foi um objetivo dos *web designer*s (MacDonald, 2014).

Em 1998, a linguagem de estilo CSS2, incluiu a regra `@font-face` que permitia aos *browsers* fazer o *download* da informação de um ficheiro de fonte, assim, virtualmente, qualquer fonte podia ser usada em qualquer site (Lupton, 2014). Razões como problemas de otimização, devido à revolução dos ecrãs e o facto de a maioria das fontes serem pagas, levou a que grandes empresas como a Microsoft estivessem hesitantes em adicionar uma característica que fosse permitir e encorajar os *designer*s a usarem fontes que tinham instaladas no computador e as usassem na página *web* sem a devida permissão. Isto levou que a implementação de *webfonts* e da regra `@font-face,` tivesse sido atrasada (MacDonald, 2014).

Em 2008, a Mozilla Firefox e a Apple Safari incorporaram a regra `@font-face`, tornando-se acessível à maioria dos utilizadores da Internet. Foram criados serviços que hospedavam fontes, como o Typekit, Fontdeck e Google Fonts que preencheram o vazio de licenças e pirataria que ainda existia no CSS3 (Lupton, 2014).

Figura 23 : O antigo site do MSN, tem uma largura fixa, deixando espaço em branco quando a largura do ecrã é maior que o site (Peterson, 2014).

2.2.3 RESPONSIVE WEB DESIGN

Até recentemente, as páginas *web* eram desenhados de forma a ocupar o tamanho do ecrã mais comum, pois não havia uma grande variedade de tamanhos de ecrã. Os *designer*s queriam que o seu *design* fosse igual em todos os ecrãs, por isso desenhavam para uma largura fixa e em ecrãs com largura superior, criavam espaços vazios como pode ser observado na figura 23 (Peterson, 2014).

O surgir do iPhone, em 2007, veio revolucionar a maneira como eram desenhadas as páginas *web*. Como foi anteriormente referido, as páginas *web* eram pensadas para uma largura fixa (perto do 960 pixels), o que na largura de 320 pixels do iPhone obrigava o utilizador a fazer zoom em qualquer área da página. A solução inicial foi encolher todo o desenho da página, para ocupar os 320 pixels, mas os *web designer*s sabiam que não era uma solução ótima, por isso, precisaram de criar novas formas de visualização da *web* em ecrãs pequenos como os do iPhone (Peterson, 2014).

A solução inicial adotada pelos *designer*s, foi simplesmente criar uma versão separada da página *web*, para telemóvel. Levando a um trabalho extra, por parte dos *web designer*s (Peterson, 2014).

Em 2010, a Apple lançou o iPad, que voltou a revolucionar a maneira como as páginas *web* eram pensadas. Assim, os *designer*s aperceberam-se que era necessário pensar em formas de construir uma página *web* que se ajustasse a qualquer tamanho de ecrã (Peterson, 2014).

A comunidade de *web designer*s voltou com o conceito de *design* fluido, usando larguras com base em percentagens e tentando criar soluções para os tamanhos de ecrã mais pequenos. Isto veio permitir que a largura dos elementos da página se ajustasse a qualquer tamanho de ecrã, tornando as diferenças mínimas, de tamanhos de ecrã, fáceis de resolver (Peterson, 2014). Assim, para evitar desenhar diferentes layouts para cada tamanho de ecrã, muitos *designer*s adotaram o conceito de *Responsive Web Design*. Ou seja, a construção da página é pensada para que o layout e a organização da informação na página se adapte consoante o tamanho do meio em que está a ser observado, em vez de criar diferentes páginas, para cada dispositivo (IA, 2012).

2.2.4 ADAPTATIVO E LÍQUIDO

Assim, existem duas abordagens para criar um layout responsivo, um *design* líquido e adaptativo. Um *design* líquido ajusta continuamente os elementos ao tamanho do ecrã do utilizador, como é mostrado na figura 24. Por outro lado, um *design* adaptativo responde consoante a orientação do *browser* ou dispositivo do utilizador, oferecendo geralmente configurações para os diferentes dispositivos, computador, *tablet* e telemóvel, como é mostrado na figura 25. Num *design* líquido, as colunas ficam mais ou menos estreitas consoante o tamanha do dispositivo, enquanto que num *design* adaptativo, certos elementos podem ocupar mais ou menos colunas, diferindo entre dispositivos.

Ambas as abordagens têm vantagens e desvantagens, por exemplo, o *design* simplesmente líquido não ajuda muito na legibilidade do texto, pois esta é afetada pela largura da linha, por outro lado, um *design* líquido é ideal para ajustar o tamanho do texto. Assim, uma combinação das duas abordagens na construção de uma página é o ideal, por exemplo, o tamanho da caixa do conteúdo é adaptativo podendo ocupar 30% num ecrã de computador e 100% num de telemóvel, como é mostrado na figura 25, enquanto que os valores como o tamanho de letra e o espaçamento entre colunas é líquido (IA article 2012 e Lupton, 2014).

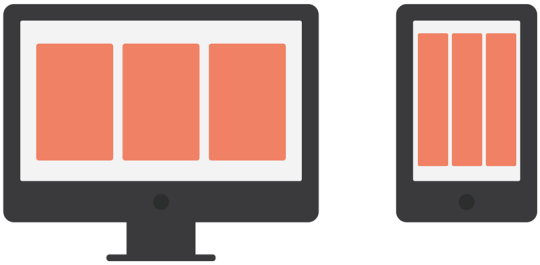


Figura 24 : Exemplificação de design líquido em medidas de colunas.

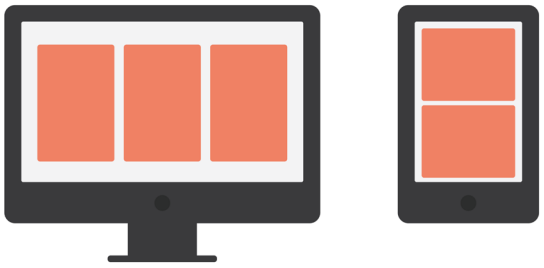


Figura 25 : Exemplificação de design adaptativo em medidas de colunas.

2.2.5 SCRIPTS DESENVOLVIDOS PARA A TIPOGRAFIA

Como já foi abordado, a programação pode ser aplicada na tipografia de forma facilitar o processo de construção de uma fonte, ou acrescentar-lhe variabilidade. Assim, de seguida, vão ser apresentados vários projetos, onde a programação é aplicada na tipografia no ambiente da *web*.

Opentype.js de Frederik De Bleser, mostrado na figura 26, e JsFont.js de Niklas von Hertzen, mostrado na figura 27, são *scripts* semelhantes onde é possível aceder e modificar as tabelas de dados dos ficheiros fonte. Ambos processam uma fonte TrueType ou OpenType (cf. Glossário) através de um pedido AJAX (cf. Glossário), e transformam esses valores binários em informação legível, mapeando-os em objetos JavaScript (cf. Glossário), para uma melhor organização. Assim que os valores estejam processados é possível alterar qualquer parâmetro devolvido pelas tabelas da fonte (Bleser, 2017; Hertzen, 2017).

Embora semelhantes, o OpenType.js tem uma melhor documentação, contendo em si já um conjunto de funções (cf. Glossário) que permite mais facilmente aceder e alterar as informações de uma fonte, descrevendo as diferentes tabelas de uma fonte, ilustrando ainda como a forma de uma letra é definida e que diferentes variáveis da tabela, como é ilustrado na figura 28.

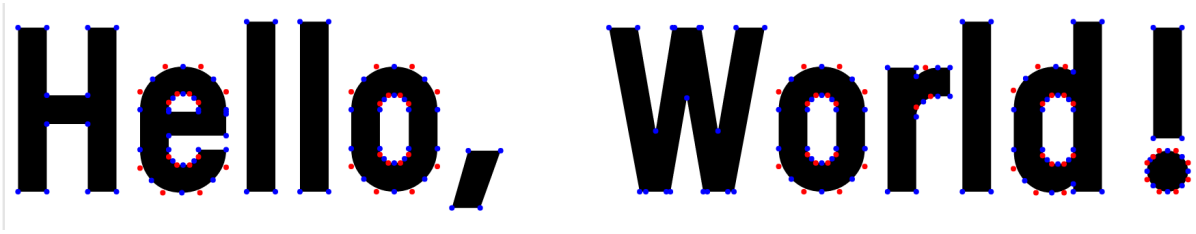
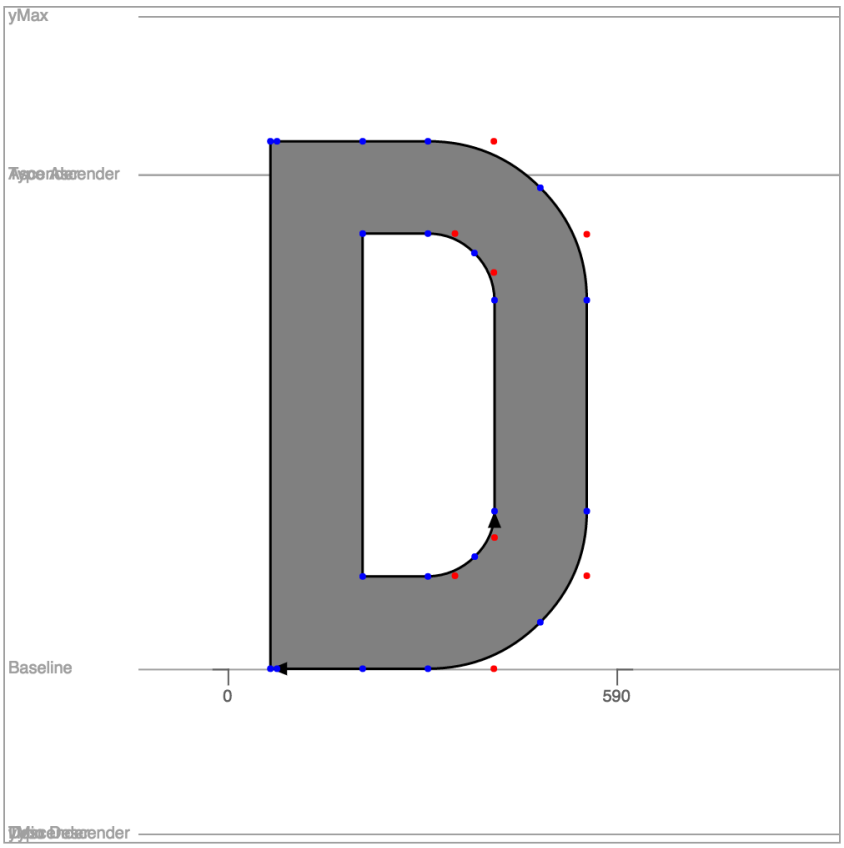


Figura 26 : Opentype.js aplicado, mostrando os vários pontos que definem a letra (Bleser, 2017).

Figura 27 : Jsfont.js aplicado, sendo que as tabelas da fonte são interpretadas e alteradas aleatoriamente (Hertzen, 2017).



name D
unicode 0044
index 33
xMin 65
xMax 545
yMin 0
yMax 800
advanceWidth 590
leftSideBearing 65
contours:

```
x=65 y=0
x=65 y=800
x=75 y=800
x=205 y=800
x=304 y=800
x=404 y=800
x=545 y=659
x=545 y=559
x=545 y=239
x=545 y=141
x=404 y=0
x=304 y=0
x=205 y=0
x=75 y=0
```

```
x=405 y=239
x=405 y=559
x=404 y=601
x=345 y=660
x=304 y=660
x=205 y=660
x=205 y=140
x=304 y=140
x=345 y=141
x=405 y=199
```

Figura 28 : Dados completos de um glifo, retirados das tabelas da fonte (Bleser, 2017).

Plumin.js foi desenvolvido por Yannick Mathey e Louis-Rémi Babé e é um *script* que permite manipular fontes através de JavaScript, como pode ser observado na figura 29. Com este *script* pretendiam que fossem criados novos tipos de letra, aumentando a sua qualidade e automatizando certas tarefas. Plumin.js ainda está em desenvolvimento, sendo que ainda não existe nenhuma documentação. Para a sua construção tirou-se partido de projetos como o Paper.js, que é um *script* que trabalha com vetores, oferecendo um modelo limpo e funcionalidades úteis para criar e trabalhar vetores e usa o Opentype.js para a análise de fontes (Mathey & Babé, 2017).

Figura 29 : Plumin.js em funcionamento, onde as letras "a" e "o" da fonte, foram substituídas por formas desenhadas em JavaScript (Mathey & Babé, 2017).

2.2.6 FONTES VARIÁVEIS

As fontes variáveis foram anunciadas em 14 de Setembro de 2016, na conferência ATypI em Warsaw, onde se reuniram representantes de várias marcas, como: Adobe, Apple, Google e Microsoft. Estas empresas têm vindo a trabalhar juntas ao longo de meio ano, para melhorar as especificações do OpenType, de modo a incluir uma nova tecnologia, a *OpenType Font Variation*. Assim, um ficheiro de uma fonte comporta-se como um conjunto de fontes, ganhando uma flexibilidade infinita na largura, na altura e em outros atributos, sem o contratempo de o ficheiro aumentar de tamanho (Brown, 2016).

Isto iria permitir que o glifo de uma fonte fosse condensado e estendido de forma a acomodar larguras de ecrã grandes e pequenas. Na figura 30, é mostrado algum tipo de flexibilidade que esta tecnologia irá permitir. As possibilidades que pode trazer são enormes e vêm facilitar a maleabilidade de uma fonte na construção de objetos de *design* em tempo real na *web* (Brown, 2016).

Figura 30 : Sequência de gif apresentado por Erik van Blokland (Brown, 2016).

2.3 Trabalhos Relacionados

Neste capítulo, são apresentados e explorados alguns projetos. Estes, embora não sejam uma representação direta do que se pretende desenvolver no projeto de dissertação, são casos que têm algum valor na área da Tipografia/Responsividade ou uso da programação na tipografia, de forma a criar resultados tipográficos mais aleatórios ou permitir originar resultados nas formas tipográficas, que humanamente seria muito difícil de realizar. Assim, são analisados casos onde é realçado o valor da mutabilidade numa fonte como característica que pode possibilitar novas abordagens para a tipografia e, consequentemente, o uso da programação para obter uma extensão mais alargada de resultados. Por outro lado, são analisados casos onde a largura da fonte é alargada ao limite, para esta se ajuste da melhor forma ao suporte onde vai ser aplicada.

O objetivo é, através da análise destes trabalhos, referenciar projetos importantes e retirar conclusões pertinentes que possam ajudar na fundamentação e construção do produto a desenvolver nesta dissertação.

2.3.1 FONTE DÚBIA

A fonte Dúbia, que se pode observar na figura 31, surgiu em 2014 com a necessidade de criar uma fonte responsiva, foi desenhada de raiz pelo estúdio de Coimbra Bürocratik de forma a que houvesse liberdade na deformação segundo as suas próprias regras. Dúbia surgiu da necessidade de usar uma fonte como elemento de preenchimento de espaço em função da estética de um dado projeto, sendo este um cartaz ou uma identidade (Esteves, 2016; Rodrigues 2016).

Os seus criadores inspiraram-se no comportamento da fonte "esticar" com a imprevisibilidade das fontes mutáveis, que tem vindo a ser debatido desde os anos 80. Por outro lado, entenderam que o esticar a fonte poderia resolver a falha entre vários dispositivos, sem valores fixos, para que estes não fiquem datados com o percorrer do tempo (Esteves, 2016; Rodrigues 2016).

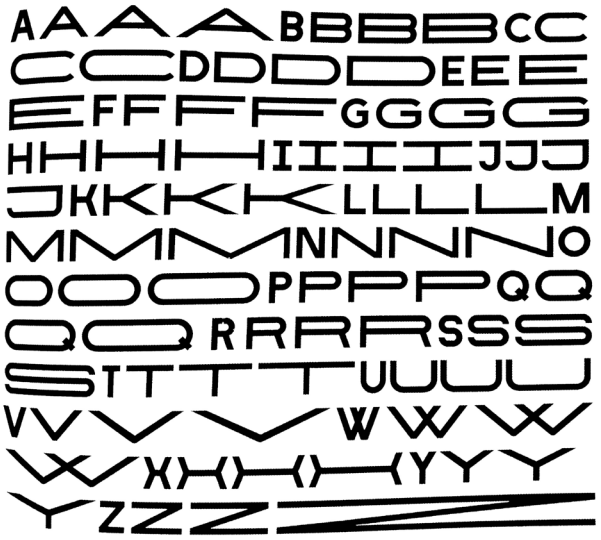


Figura 31 : Fonte Dúbia, desenhada pelo estúdio Bürocratik (Slanted, 2016).

Figura 32 : Fonte Dúbia, aplicada na identidade gráfica do Crossfit Solum em Coimbra (Crossfit-Solum, 2017).

A aplicação da fonte Dúbia é possível ser observada num projeto realizado pelo estúdio em Coimbra, para a identidade gráfica do Crossfit Solum, mostrado na figura 32, sendo possível observar o seu potencial. A fonte Dúbia, foi desenhada para o propósito deste projeto, pois a própria mutabilidade da fonte, complementava bem o conceito de maleabilidade e extensão ligadas ao esforço da modalidade física (Esteves, 2016; Rodrigues 2016).

2.3.2 FONTE FRANCIS

Francis é uma fonte que contém um conjunto de estilos gradiente que dinamicamente aumentam e diminuem a largura do glifo, dependendo do gradiente escolhido. A lista de gradientes pode ser observada na figura 33. O efeito de gradiente é conseguido pois cada estilo de gradiente possui 2,690 glifos que são selecionados automaticamente usando uma característica do OpenType (Typotheque, 2016).

Francis foi desenhada por Nikola Djurek em 2016 e o código que controla a amplitude da fonte foi escrito por Karsten Luecke. A inspiração para a criação da Francis veio dos estilos de letra do início do século XX, muitas vezes usados na publicidade Europeia, mas também pela racionalidade geométrica (Typotheque, 2016).

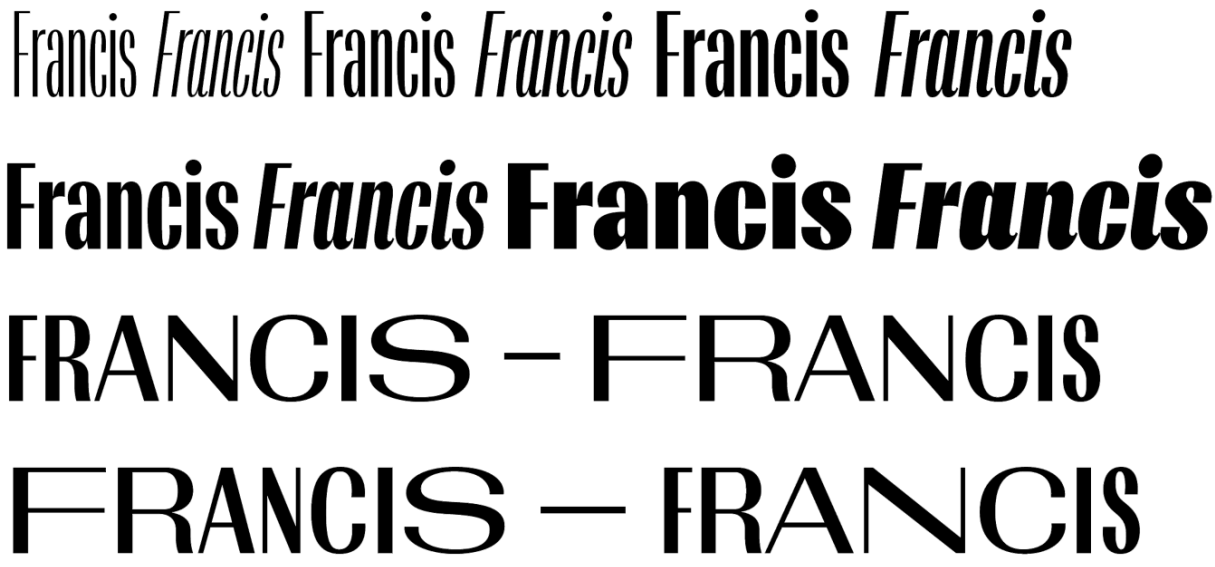


Figura 33 : Lista de “Gradientes” da fonte Francis (Typotheque, 2016).

2.3.3 TIPOGRAFIA COWHAND

Cowhand é uma tipografia desenhada por Toshi Omagari numa maratona de três dias e meio no escritório da Monotype em Nova York. Cowhand através de um *script* desenvolvido em conjunto com a fonte, permite calcular a largura que cada glifo numa palavra, a qual deve ocupar o espaço de forma a preencher toda a linha, como se pode observar na figura 34. Assim palavras com menos caracteres vão ter uma largura maior do que uma palavra com mais caracteres (Chahine, 2015).

A inspiração de Toshi Omagari que o levou a querer esticar os glifos veio do seu interesse pelo trabalho do artista Xu Bing, que escreve palavras inglesas com a aparência de caracteres chineses. Um exemplo do seu trabalho pode ser visto na figura 35, onde se pode ler "Art for the people". Toshi Omagari gostou da ideia de tratar uma palavra como uma forma constante e queria fazer o mesmo no Latim, não seria uma réplica pois não conseguia automaticamente criar palavras em letras, por isso optou por ajustar as palavras à largura da linha (Omagari, 2016).

O auxílio da programação na criação desta fonte também foi muito importante, pois Toshi Omagari criou o *script* que lhe permitiu gerar mais facilmente os glifos com as diferente larguras, e como Omagari referiu numa entrevista (Chahine, 2015):

> *"You don't have to program to be a good type designer. But when used well, scripts can take you where you humanly can't, and they can be the most important design tool"*

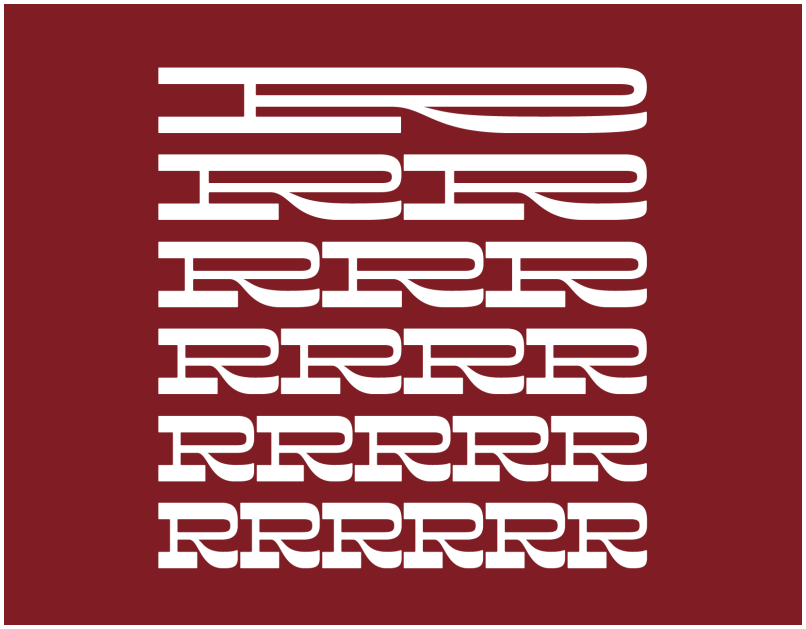


Figura 34 : Fonte Cowhand desenhada por Toshi Omagari (Chahine, 2015).

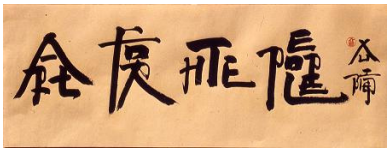


Figura 35 : Xu Bing, "Art for the People".

2.3.4 SONU

Sonu é um fonte desenhada pelos Thirst, apresentada na figura 36, baseando-se na fonte Fregio Mecano, anteriormente abordado no Estado da Arte. A fonte foi criada para um projeto de identidade do primeiro edifício de micro-unidades em Chicago e foi desenhada com módulos, permitindo expandir e encolher a fonte consoante o espaço existente. A aplicação da fonte pode ser observado nas figuras 36, 37 e 38 (Thirst, 2017).

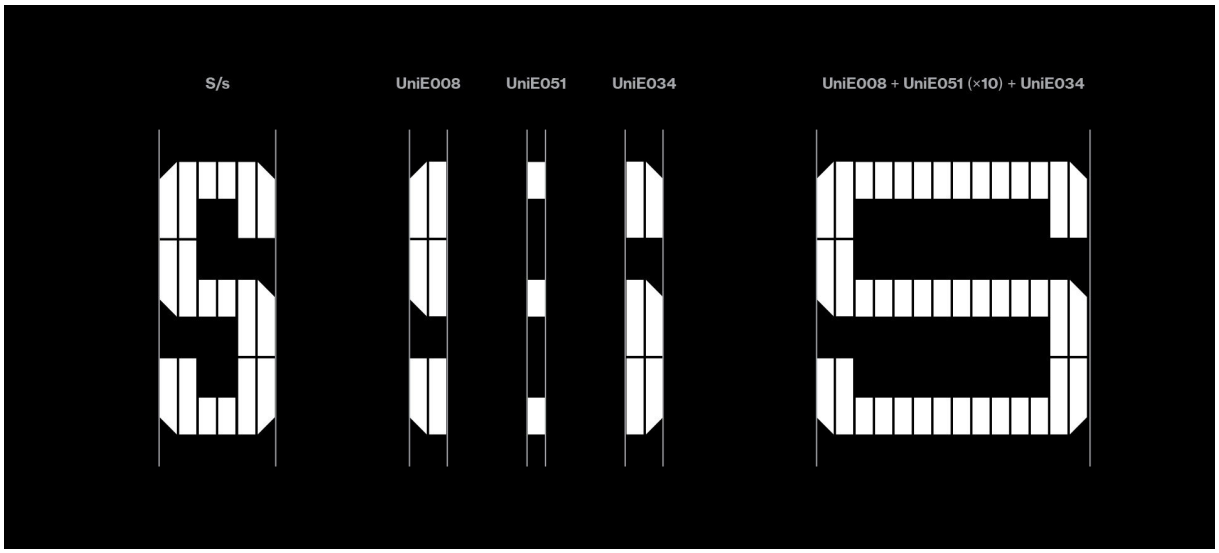


Figura 36 : Fonte Sonu (Thirst, 2017).

Figura 37 : Aplicação da Fonte Sonu, na web (Thirst, 2017).

Figura 38 : Aplicação da Fonte Sonu (Thirst, 2017).

2.3.5 LAIKA

Michael Flückiger e Nicolas Kunz desenharam a tipografia Laika em 2009, uma tipografia dinâmica, onde o peso da fonte, inclinação do itálico, serifas e contraste podem ser alterados de forma dinâmica. Assim, através de inputs são gerados os valores intermédios das características que afetam. Por exemplo, na página *web* que faz uma apresentação da fonte, as diferentes características variam consoante o posicionamento do rato no ecrã, alguns dos resultados são possíveis de observar na figura 39 (Flückiger & Kunz, 2016).

Michael Flückiger defende que uma fonte deve ser entendida dinamicamente, tendo em conta que está a ser observada,num meio dinâmico, a *web*. Assim, defende que as formas da tipografia também deviam ter a possibilidade de se transformarem e responderem às circunstâncias (Flückiger & Kunz, 2016).

Figura 39 : Sequência de Screenshots, da tipografia mutável Laika em tempo real na *web* (Flückiger & Kunz, 2016).

2.3.6 FONTES VARIÁVEIS NA WEB

No ano de 2015, Andrew Johnson escreveu um artigo sobre o uso de fontes variáveis na *web*, para aumentar e diminuir o peso da fonte, de forma a oferecer aos utilizadores a melhor experiência, com o peso adequado ao dispositivo que estão a utilizar (Johnson, 2015).

Andrew Johnson define a interpolação em fontes como o processo que gera novas fontes intermédias a partir de fontes *master* (cf. Glossário). Calculando automaticamente os valores entre os pontos, os tipógrafos conseguem criar fontes novas com novos pesos, através da interpolação, em vez de estar a desenhar os vários pesos manualmente. Atualmente, já existem ferramentas que permitem aplicar a interpolação dentro de programas de construção de fontes, facilitando a criação de vários pesos numa família. De qualquer modo, a interpolação é apenas usada dentro dos programas, não vem exportada com o ficheiro fonte.

Um exemplo onde a interpolação foi usada dentro de um programa, para originar vários pesos, foi a família TheSans, mostrada na figura 40 (Johnson, 2015). A interpolação pode ser usada para alterar, para além do peso, aspetos fundamentais da estrutura dos glifos da fonte, como as serifas, o contraste e proporções a partir de duas fontes *master* (Johnson, 2015).

Figura 40 : Pesos individuais gerados a partir da interpolação, da família TheSans (Johnson, 2015).

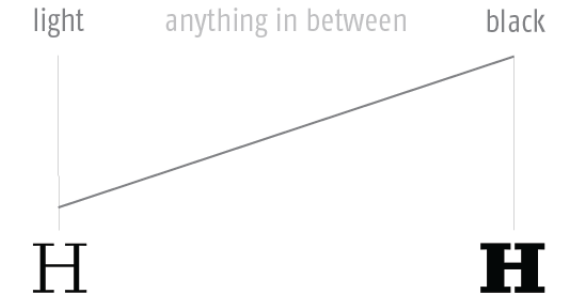


Figura 41 : Esquema explicativo, da interpolação em tempo real (Johnson, 2015).

Andrew Johnson defende que o mesmo conceito pode ser usado na *web*, para oferecer diferentes pesos da fonte aos utilizadores. A tipografia sempre esteve conectada ao seu meio, como por exemplo, os tipos móveis e a fotocomposição influenciaram a maneira como os tipos eram desenhados na sua altura. Atualmente, com a responsividade, torna-se necessário que existam elementos flexíveis que se ajustem a todos os dispositivos (Johnson, 2015).

O que Andrew Johnson pretende com a interpolação de fontes na *web*, é que a interpolação seja feita em tempo real, em vez de estar a carregar vários ficheiros de fonte. Assim as próprias fontes conseguem responder ao seu contexto, para além, de se evitarem mudanças de pesos repentinas, em pontos de quebra, como é ilustrado na figura 41 (Johnson, 2015).

A interpolação na *web* consegue ser feita a partir de imagens ou canvas(cf. Glossário), mas estas abordagens não permitem que o texto seja selecionável e acessível através de leitores de ecrã. As fontes SVG (cf. Glossário) oferecem manipulação à fonte, mas

faltam-lhe características que a tornem uma fonte robusta, como a otimização no ecrã e tabelas do OpenType com suporte na linguagem e estilos alternativos (Johnson, 2015) .

Ao contrário dos ficheiros SVG, que são modificados facilmente, os formatos de ficheiros fonte são compilados em ficheiros binários, o que complica o processo de fazer mudanças em tempo real. Um ficheiro fonte consiste em séries de informação que descrevem a fonte e que são guardados em tabelas. Assim, para a interpolação em tempo real na *web* ser possível, é necessário uma ferramenta adequada à *web*, que converte os ficheiros fonte num formato que seja possível ler e editar em JavaScript (Johnson, 2015).

Andrew Johnson encontrou projetos, como o jsfont.js e o opentype.js que permitem aceder e modificar facilmente as tabelas da fonte num *browser*. Andrew Johnson explica o processo como "like a game of connect-the-dots", onde cada glifo é composto por uma série de pontos posicionados numa grelha x-y, como pode ser observado na figura 42 (Johnson, 2015).

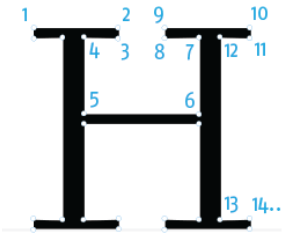


Figura 42 : Série de pontos numerados num glifo H (Johnson, 2015).

A interpolação requer uma modificação de um glifo perto do ponto original da fonte *master*. Para fazer a interpolação na *web*, é preciso mover os pontos individualmente. Ao mesmo tempo que os valores dos pontos são alterados, as propriedades como *xMin* e *xMax* também têm de ser interpolados de forma a assegurar que a caixa que delimita um glifo é larga o suficiente para mostrar o glifo completo. As várias medidas que devem ser tidas em conta na interpolação do glifo, de forma a manter a integridade da fonte, podem ser observadas na figura 43 (Johnson, 2015).

Andrew Johnson tirou partido do *script* Jsfont de Niklas von Hertzen para aceder e modificar as tabelas da fonte e desenvolveu um *script* onde cria a interpolação de pesos em tempo real na *web* através de duas fontes *master*, como é possível observar na figura 44.

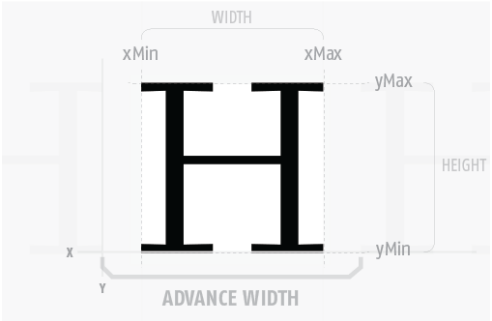


Figura 43 : Propriedades do glifo, onde ambos *xMin*, *xMax* e *advancewidth* devem ser interpolados em conjuntos com os pontos de coordenadas do glifo (Johnson, 2015).

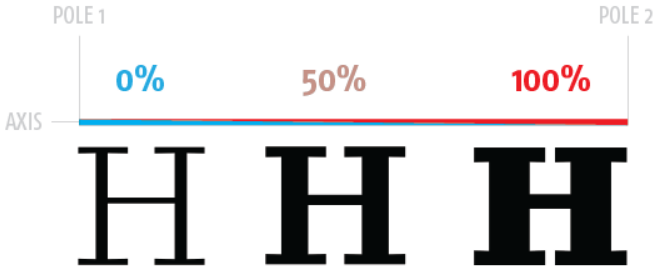


Figura 44 : Interpolação do glifo H, usa 50% do peso *light* e 50% do peso *bold* (Johnson, 2015).

2.3.7 AXIS-PRAXIS

Após a conferência da ATyp, onde com a nova versão do OpenType, são introduzidas as fontes variáveis, Laurence Penney criou um projeto, o Axis-Praxis, onde tira partido da existência de fontes variáveis na *web*. Axis-Praxis é uma ferramenta *web*, de construção tipográfica simples, como é apresentado na figura 45, na qual é possível alterar os eixos da fonte variável, como: peso, altura-x, largura, etc (Penney, 2017).

O Axis-Praxis foi construído através das linguagens Php, JavaScript com auxílio do JQuery (cf. Glossário), aproveitando as fontes variáveis e da especificação `font-variation-settings` do CSS4, que ainda está em desenvolvimento. Assim, por tirar partido de ferramentas relativamente recentes ou ainda em desenvolvimento, não é facilmente testado. O utilizador tem de ter o sistema operativo atualizado e o *browser* para *developers*, que contém ferramentas e especificações para a *web* mais atuais, que ainda estão a ser testados (Penney, 2017).

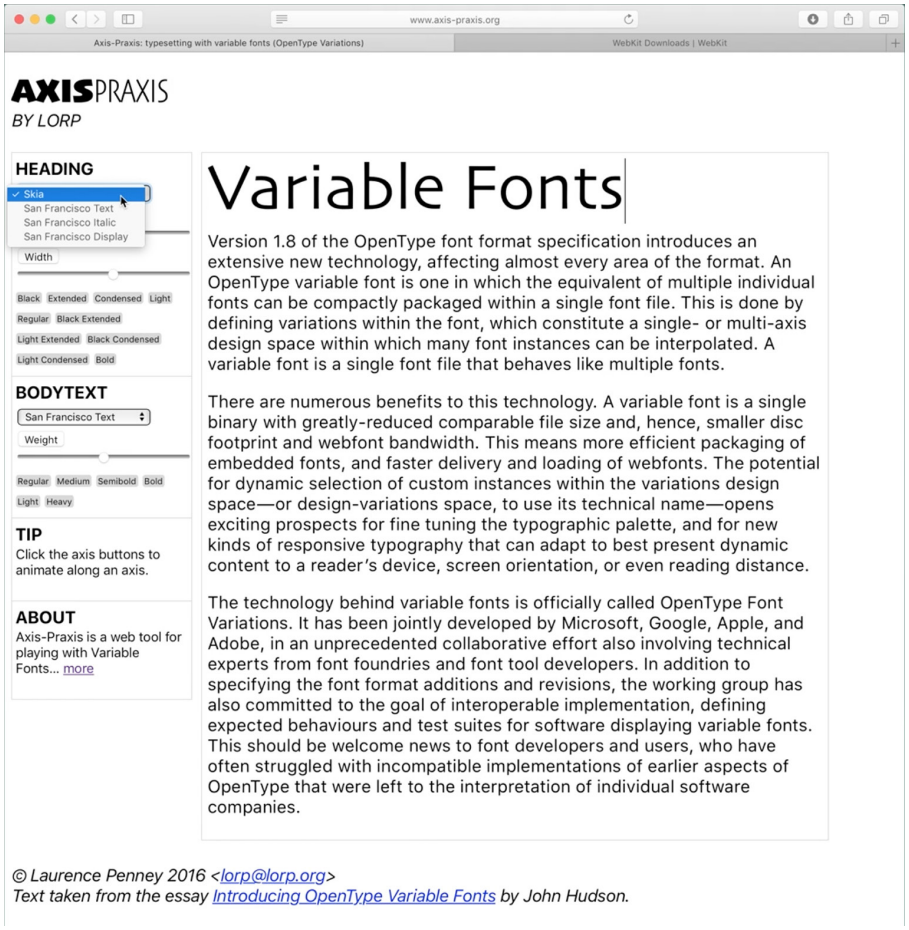


Figura 45 : Captura de ecrã do Axis-Praxis em funcionamento (Penney, 2017).

2.4 Análise do Estado da Arte

Como foi referido, a tipografia é um elemento essencial no *design,* onde as suas formas têm refletindo a sociedade e as suas necessidades, na época em que surgiram. Tendo em conta o tema desta dissertação, certos períodos históricos da tipografia, foram abordados com mais detalhe, pois as suas formas e pensamentos tornaram-se essenciais à realização do projeto prático.

Foram realçadas as formas tipográficas da Revolução Industrial, pelo facto da largura e do peso da fonte, a usar nos cartazes, serem escolhidas para causarem o maior impacto possível no observador. Assim, pela primeira vez, houve um cuidado e preocupação em escolher o peso e a largura da fonte, de forma a melhor se ajustar ao suporte onde era impresso.

O movimento modernista na tipografia também foi muito referenciado, pelo aparecimento de fontes geométricas e modulares, que por sua vez, influenciaram muito a escolha da fonte a usar para testar o *script*, por permitir uma fácil alteração de aspetos da anatomia da fonte, sem a deformar. Por outro lado, o aparecimento de grelhas e de ideias de sistemas de *design* flexíveis de Karl Gerstner, em que uma identidade é desenvolvida com um sistema, planeado para se ajustar a qualquer suporte, vieram influenciar e ajudar a fundamentar a ideia do projeto e realçar a importância do desenvolvimento de projetos flexíveis. Este tipo de pensamento, torna-se cada vez mais importante no desenvolvimento de um projeto de *design* e essencial num projeto para a *web*, pelas exigências que requer o desenvolvimento de um projeto para um grande número de tamanhos de ecrã.

Também foi discutido o aparecimento do computador e, naturalmente, o aparecimento da *web*. Foi abordado como a aparência da *web* e as ferramentas estéticas têm evoluindo, de forma a responder às necessidades de produção de *design* na *web*, sendo que a *web* foi se tornando um meio de comunicação cada vez mais importante. Foi discutido com mais detalhe, a implementação de fontes na *web* e como o aparecimento de diferentes tamanhos de ecrã alterou a forma como eram planeadas e desenvolvidas as páginas *web*. Assim, desde o seu aparecimento, a *web* focou-se no desenvolvimento e implementação de ferramentas que permitem uma maior liberdade expressiva no *web* design. A par desta preocupação pelo desenvolvimento de novas ferramentas, foi abordado o atual desenvolvimento da nova versão do OpenType, que inclui as características de fontes variáveis, permitindo que um ficheiro fonte se comporte como se fossem muitos.

Como se pode observar nos trabalhos relacionados, desenvolvidos para a *web*, ainda há limitações tecnológicas na incorporação de mais variabilidade e expressividade da tipografia na *web*, mas há um esforço em combater essas limitações. As quais juntamente com a falta de tecnologias, incentivaram o desenvolvimento deste projeto, pela vontade de desenvolver ferramentas que superem parte dessas limitações, sendo que o *script* permite o desenvolvimento de abordagens mais exploratórias da tipografia na *web*, tirando partido dos diferentes tamanhos de ecrã.

O projeto prático que me propus desenvolver, é um projeto experimental, com o objetivo de possibilitar uma maior expressividade da tipografia na *web*. Tratando-se de um projeto experimental, o seu processo consiste na procura de novas soluções para um dado problema, ou seja, requer uma certa inovação. No entanto, projetos experimentais na área da tipografia, nunca são completamente novos, baseando-se em práticas e pensamentos já existentes, ou seja, foi necessário fazer um levantamento do que já foi feito, nas áreas em que o projeto se insere, a tipografia e a *web*, de forma a melhor fundamentar e ajudar o desenvolvimento do projeto (Biľak 2005).

3. Abordagens Metodológicas

Neste capítulo, serão exploradas algumas características da dissertação referidas na introdução como os objetivos, as funcionalidades do projeto a desenvolver, o processo a seguir, os problemas a ultrapassar e os resultados que se pretendem obter. Por último, será ainda apresentado o plano de trabalhos.

3.1 OBJETIVOS

Com a realização desta dissertação pretende-se atingir um conjunto de objetivos que influenciam a orientação do projeto e os métodos de trabalho. A nível prático, tenciona-se:

— Investigar sobre a evolução das formas tipográficas e como a tecnologia e o seu contexto histórico as influenciaram;

— Fazer um levantamento de trabalhos relacionados com os temas discutidos nesta dissertação, em que a tipografia se torna um elemento de preenchimento do suporte e trabalhos em que a programação atua na tipografia;

— Compreender o processo de construção e desenvolver uma fonte e os comportamentos anatómicos que um glifo deve apresentar quando esticado e prever todos as suas iterações intermédias;

— Identificar diferentes abordagens possíveis a fazer, de forma a desenvolver o projeto prático proposto;

— Construir um *script* na linguagem de Javascript, que permita ajustar a largura da tipografia na *web* de forma a ocupar o tamanho do ecrã em que está a ser observada, através de duas fontes *Master*, uma com uma largura regular, outra com uma largura extra larga;

— Desenvolver o *script* de forma a que este seja fácil implementar e intuitivo, permitir ao utilizador alterar e aceder, facilmente a todos as variáveis do *script* e oferecer diferentes modos de visualização;

— Construir uma página *web* de apresentação do *script*, onde mostre as diferentes possibilidades deste *script* e um guia de como implementá-lo;

— Planear e desenvolver a divulgação do *script*.

A construção de uma fonte extra larga desenvolveu-se num programa focado na criação de fontes, o FontLab, o qual oferece as ferramentas necessárias para criar a fonte da forma desejada e devolver um ficheiro fonte, em que é possível ler a informação a partir do *script* OpenType.

Com a implementação e divulgação do *script* pretende-se que este atue como ferramenta para *web designer*s e *developers*, potenciando e facilitando uma maior experimentação, permitindo uma maior maleabilidade da tipografia, a exploração das duas formas de modo a ocuparem o tamanho do ecrã e permitir que esta ferramenta seja aplicável a qualquer par de fontes.

3.2 PROCESSO

Para ir de encontro aos objetivos anteriormente propostos, o processo de desenvolvimento do projeto está dividido em duas etapas, como explicitado na Figura 46.

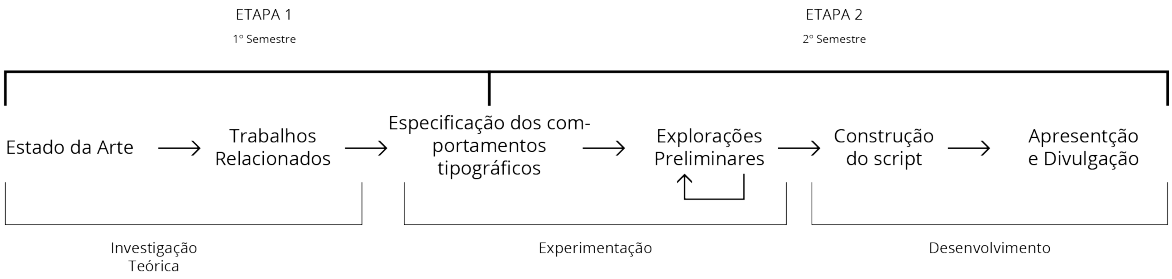


Figura 46 : Cronograma elucidativo do processo seguido.

Numa primeira etapa, foi feita investigação, tanto na área da tipografia como da *web* e consequentemente responsividade, que foi desenvolvida no primeiro semestre. O levantamento do estado da arte na área da tipografia inclui a evolução desta e o modo como a tecnologia influenciou as suas formas. Na área da *web*, inclui-se a sua evolução, o surgir do termo *responsive* e um levantamento das tecnologias existentes que permitem alterar a anatomia de uma fonte. Nesta etapa, incluiu-se um levantamento e análise de trabalhos que, de certa forma, estão relacionados com o trabalho prático que me proponho desenvolver, foi feita, também, uma conceptualização da ideia para o projeto prático e começou-se a prever e experimentar as formas que seriam tomadas pela tipografia, quando sofressem deformações pelo *script*.

Por último, nesta fase, foi feito um planeamento detalhado do processo de desenvolvimento do projeto prático, fazendo o levantamento das tecnologias a usar, como o seu processo deveria ser abordado de forma a atingir todos os objetivos propostos e a planear as funcionalidades do *script*.

Na segunda etapa, foi desenvolvido o projeto prático e a documentação dos resultados obtidos. O desenvolvimento prático teve como ponto de partida a criação da fonte extra larga e começou-se a tentar aceder à informação dessa fonte, a partir do *script*. De seguida, foram aperfeiçoadas as formas da fonte e começou-se a trabalhar no *script*, com a fonte desenvolvida.

Foi planeado a divulgação do *script*, que consistiu no desenvolvimento da imagem, planeamento e implementação do site onde é apresentado. Com o *site* pronto, seguiram-se os testes, de modo a poder avaliar, de certa forma, a implementação do *script*, o parecer de *designer*s sobre a utilidade do *script* e encontrar possíveis erros.

3.3 PROBLEMAS RESOLVIDOS

Os problemas resolvidos nesta dissertação estiveram relacionados com os desafios em desenvolver um projeto experimental que, nesta dissertação em específico, consiste em idealizar e produzir uma fonte quer ao nível do *design* quer ao nível de código, onde apenas com a combinação dos dois saberes, é possível que a fonte seja alargada. Assim, foi necessário analisar, planear e construir a fonte, de forma a que os seus comportamentos, sob o efeito do *script*, sejam os esperados e que mantenham a integridade da fonte.

Um dos objetivos é oferecer uma ferramenta que permita uma maior experimentação das formas da tipografia na *web*, permitindo que a tipografia se torne um elemento expressivo e que preencha espaço num ecrã de computador. Assim, tornou-se importante que esta ferramenta funcione em qualquer ficheiro fonte, que é recebida antes de correr o *script*. Isto vai permitir uma maior personalização da ferramenta em cada projeto onde vai ser implementada.

3.4 RESULTADO

Este projeto tem como resultado principal o *script*, embora mais artefactos tenham sido construídos de forma a testar as suas funcionalidades, desenvolvendo uma fonte e divulgando-a devidamente.

A nível teórico, com esta dissertação, pretende-se contribuir com um documento que reúna de forma clara, uma síntese da evolução das formas tipográficas ao longo da história; das possibilidades que a programação traz à tipografia e esclarecer a necessidade da presença de fontes mais flexíveis e variáveis na *web*. Em relação ao *script*, foi idealizada a construção de uma ferramenta para a *web*, que permite que a largura de qualquer fonte seja alterada, de acordo com o tamanho do ecrã.

A nível do *Web Design,* pretende-se oferecer uma ferramenta que incentive uma maior experimentação e expressão através da tipografia na *web*, permitindo uma maior maleabilidade das formas tipográficas e possível interatividade por parte dos utilizadores. Por outro lado, demonstrar a importância, da criação de ferramentas e suportes que incentivam a criação artística e gráfica na *web*. Será, ainda, inserido no desenvolvimento, a imagem do projeto desenvolvida para a sua apresentação e comunicação.

3.5 PLANEAMENTO DE TAREFAS

Na figura 48 são mostradas as diferentes tarefas que me proponho realizar ao longo da dissertação, com as respetivas metas assinaladas.

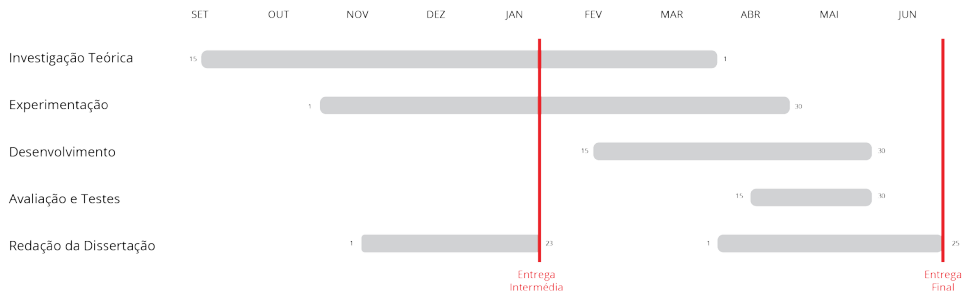


Figura 47 : Cronograma do planeamento de tarefas apresentado na entrega intermédia.

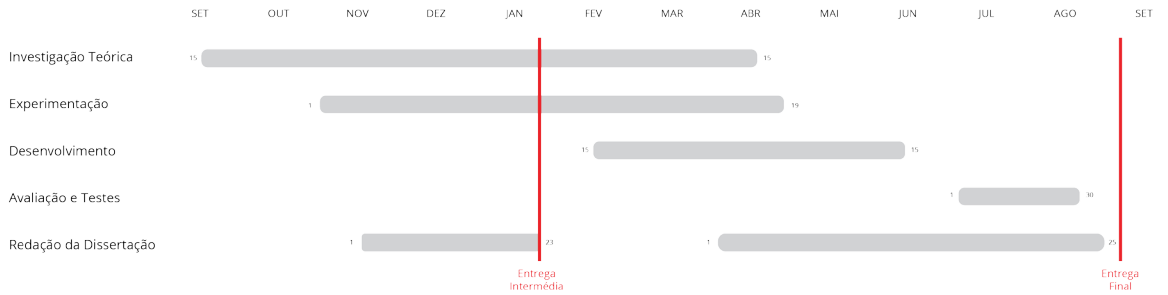


Figura 48 : Cronograma do planeamento de tarefas seguido.

O desenvolvimento do trabalho iniciou-se no primeiro semestre com a investigação teórica, que se tornou a tarefa mais importante da primeira etapa do trabalho, tendo uma grande importância quando se iniciou o trabalho prático, na segunda etapa. A investigação tem um papel importante, pelo facto de ser necessário compreender a tipografia, como esta se comporta na *web* e encontrar projetos onde é mostrada a variabilidade da tipografia na *web*, os seus problemas e limitações. Assim, a investigação teórica terminou no dia 15 de Abril, de forma a que esta tarefa fosse feita em simultâneo com os requisitos do projeto prático.

A experimentação do trabalho iniciou-se em Novembro e refere-se a três tarefas: a exploração de diferentes abordagens tecnológicas, que permitem chegar a uma solução no projeto prático; a uma exploração das diferentes soluções que o *script* poderá vir a permitir e a exploração das formas tipográficas que a fonte extra larga desenhada apresentou. Assim, esta etapa estendeu-se até ao fim de Abril, acompanhando o desenvolvimento do script e fonte, por levantarem questões que exigiram a procura de, novas soluções.

O desenvolvimento da aplicação foi iniciado após a entrega da defesa intermédia e inicialmente, andou a par com a experimentação, de forma a testar as diferentes abordagens tecnológicas e a definir soluções. Esta tarefa foi programada para ser desenvolvida em quatro meses. Na sua fase final, foi desenvolvida em conjunto com os testes realizados por diferentes utilizadores, de forma a compreender até que ponto o projeto estava a atingir os seus objetivos e como podia ser melhorado.

Como foi anteriormente referido, foram desenvolvidos quatro artefactos: a dissertação escrita, a fonte extra larga, o *script* que vai atuar na fonte e por último, a divulgação do *script*, incluindo o desenvolvimento de um *site*. Assim, os processos de experimentação e desenvolvimento foram realizados em sintonia, ao longo dos vários artefactos. O esquema a seguir apresenta, separadamente, o foco de cada artefacto realizado ao longo do desenvolvimento e experimentação do projeto prático. O desenvolvimento ocorreu da seguinte forma:

— Começou com uma análise do *script* e foram realizados testes, onde o *script* foi usado para ler a informação de uma fonte já exportada do FontLab, pois ocorriam erros quando a fonte era exportada do Glyphs. Em simultâneo, a fonte extra larga começou a ser desenvolvida.

— Ao conseguir aceder à informação da fonte, a partir do *script*, começou-se por filtrar e aceder à informação necessária e começar a comparar as diferenças de largura entre duas fontes *Master*. Assim, foi possível fazer a interpolação entre as duas fontes *Master*, de forma a que a fonte ocupe o tamanho do ecrã.

— Ao desenvolver a fonte, percebeu-se que certas coordenadas dos pontos dos glifos, exigiam cálculos mais complexos (função exponencial), de forma a devolver resultados satisfatórios. Mais à frente, serão apresentados os casos em que estes cálculos foram aplicados e como foram resolvidos.

— Organizar o código num objeto e as funções, de forma a permitir e facilitar a sua posterior implementação pelo *web designer*. Em simultâneo, começou-se a escrever a dissertação.

— Foram levantadas mais questões sobre o comportamento de certas letras extra largas, que também serão desenvolvidas mais à frente, neste documento.

— Foram implementados mais modos de visualização do resultado, permitindo que o mesmo texto possa ser observado de diferentes formas.

— De seguida, tendo o objetivo principal do *script* já implementado, começou-se a criar as funções que permitem alterar variáveis necessárias, corrigir erros nos cálculos, mostrar mensagens de erro, de forma a informar ao *web designer* o que correu mal e adicionou-se a função animar, que anima a mudança de largura da fonte. Por esta altura, o desenvolvimento da fonte terminou.

— Começou-se a planear a imagem e divulgação do *site* e de seguida, a sua implementação.

— A partir de 20 de Abril, o desenvolvimento consistiu no melhoramento da imagem do *site*, na continuação da correção de pequenos erros no *script* e com maior atenção, a partir de 1 Junho, na escrita da dissertação.

Em termos de calendarização previa-se que a tarefa de testes fosse realizado mais cedo, mas como exigiam que o desenvolvimento do site estivesse concluído, a sua realização foi adiada para o início de julho. Também a redação da dissertação sofreu alterações no seu calendário, pois sentiu-se a necessidade de alongar o tempo dedicado a esta tarefa, de forma a ultrapassar obstáculos apresentados na entrega intermédia e, porque em relação às outras atividades, revelou-se uma tarefa que saía da minha zona de conforto.

Assim, a entrega final foi adiada para setembro, sendo que a partir de meio de junho, o tempo foi dedicado principalmente à escrita do documento, à correção de pequenos erros no *script* e aos testes. A redação da dissertação consiste na documentação sobre o que foi feito no projeto. Sendo que, a escrita do documento é feita em dois momentos, um na primeira etapa terminando na entrega intermédia e um segundo que termina com a entrega final.

	Dissertação	Script	Fonte	Divulgação
25 FEV		1- Testar uso script do script OpenType.js para ler informação de fontes exportadas pelo Glyph/ Fontlab; 3- Filtrar código do opentype.js; 4- Começar a recollher informação necessária; 5- Comparar informação entre diferentes fontes; 6- Aplicar mudanças na fonte; 7- Interpolação entre duas fontes; 9- Aplicar funções mais complexas em certos pontos da interpolação (exponencial em coordenadas y); 10- Fonte responder a tamanho do CANVAS; 11- Organizar código, para poder ser chamado apenas com um contrutor;	1- Testar Glyphs /FontLab; 2- Começar a construir fonte; 8- Anotar certos aspectos nas diagonais e como se devem comportar ao serem esticadas;	
	12- Documentação de evolução.			
27 MAR		14- Desenvolver os vários modos de alargar a fonte; 15- Aperfeiçoar o "responder a ecrã" 16- Mensagens de erro, excecções quando o construtor é corrido. 17- mudar para modo deformar se só uma fonte está definida 18- "space" entre palavras, corrigir erros; 19- Implementar kerning; 20- Corrigir erros no cálculo do limite; 21- Permitir mudar width do container;	13- Definir comportamentos quero optar quando a fonte esticada em certas situações. 22- Terminar fonte; 23- Corrigir letras K, M e W;	
10 ABR		24- Filtrar chosenLetters que não existem, execessões na variável de entrada; 25- Altura do canvas atualizar com tamanho da fonte; 26- Animar quando página é carregada; 27- Adicionar o parâmetro "Mode width" para pixels and percentagem. 30-Função animar; 31- Aperfeiçoar "resize" do canvas quando tamanho da janela é alterado e tamanho aplicado nas letras;		28- Primeiros mockups site; 29- Implementar site;
19 ABR			35- Nova fonte (Arvo);	32- Definir nome; 33- Terminado implementação do site; 34- Colocar online;
27 ABR	36- Escrita;	37- Corrigir pequenos erros quando a fonte é esticada;		
5 MAI	36- Escrita;	39- Corrigir detalhes de fonte a esticar, quando a a palavra contém "space";		40- Correção de pequenos pormenores do site; 41- aplicação de script em outras páginas
24 MAI	42- Escrita;			43- Mudança do nome do script; 44- Redesign do site; 45- Pequenos reparos.

4. Fonte Responsiva para a Web

Neste capítulo, é documentada a proposta desta dissertação, onde são apresentadas as fontes de inspiração, justificado o conceito; discutidas as diferentes abordagens tecnológicas possíveis; especificados os resultados práticos e as dificuldades encontradas no seu desenvolvimento; clarificadas certas escolhas; apresentada a fonte, o *script*, as suas funcionalidades e como este funciona internamente. Por último, é abordada a divulgação do *script* e desenvolvimento da página *web* que o apresenta.

4.1 Especificar Proposta

O desenvolvimento prático desta dissertação consiste no desenvolvimento de vários conteúdos: a construção de um *script* para a *web* que permita a um *web designer* mudar a largura de qualquer fonte de modo a que esta ocupe a largura do ecrã onde está a ser observada; a construção de uma fonte que sofra as alterações da largura; e a devida divulgação do script, feita através da idealização de uma imagem gráfica e a construção de uma página web que o apresente.

De forma a não sobrecarregar o calendário, pelas exigências que a construção de uma fonte exigem, optou-se por usar uma fonte *opensource* já existente e, a partir desta, criar uma versão extra larga, para fazer a interpolação. Inicialmente, foi escolhida a fonte Reglo, apresentada na figura 49, por ser *opensource*, por permitir o seu uso e alteração e por se tratar de uma fonte geométrica, tornando o desenvolvimento da sua versão extra larga mais simples.

Figura 49 : Fonte Reglo desenhada por Sebastien Sanfilippo.

Assim, os resultados práticos desta dissertação consistem na construção de uma versão da fonte extra larga, para ser possível testar o efeito do *script* na sua construção, capaz de alterar a largura de qualquer ficheiro fonte e a devida divulgação do *script*, que irá consistir na construção de um site e respetivo material de divulgação online.

Toda a construção do *script* foi desenvolvida de forma a tornar possível uma maior exploração da tipografia na *web*, permitindo que seja aplicado a qualquer ficheiro fonte e oferecendo um conjunto variado de soluções visuais possíveis, que vão ser especificadas mais à frente. Esta abertura no produto visual devolvido pelo *script* e a sua fácil implementação, permitem aos *web designer*s criar páginas *web*, com uma maior expressividade e interatividade da tipografia na *web*.

4.2 Inspirações Visuais

Ao conceptualizar a proposta desta dissertação, foi necessário reunir projetos onde a fonte se torna um elemento de preenchimento do espaço disponível, de forma a fazer um levantamento das diferentes soluções possíveis ao construir uma fonte extra larga. Nas figuras seguintes são apresentados projetos onde a largura da fonte foi aumentada, de forma a ocupar a largura do cartaz, sendo que este efeito foi usado para realçar o tema ou o conceito a ser representado.

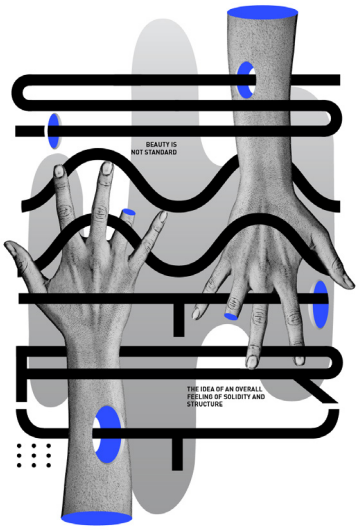


Figura 50 : Tipografia experimental SYMMETRY por Joseph Navarro (Navarro, 2017).

Figura 51 : Um projecto do estúdio Munk em 2011 (Studio, 2011).

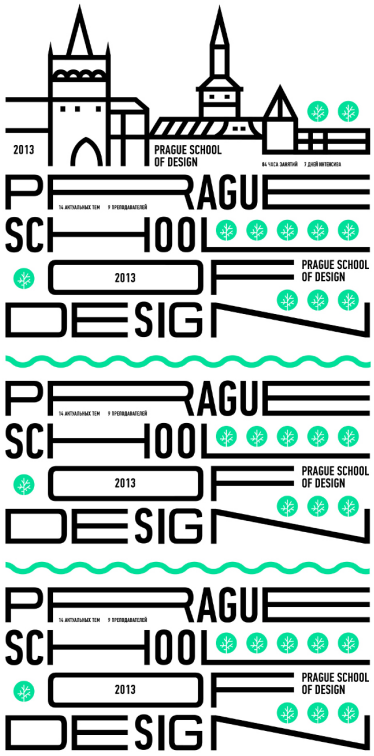


Figura 52 : Identidade para a escola de design de Praga em 2013 por Anna Kulachëk (Kulachek, 2013).

Figura 53 : Pause Fest 2014 Branding by Pennant (Grid, 2014).

Como foi apresentado apresentado nos cartazes anteriores, o *designer* teve de editar vetorialmente os glifos, de forma a alterar a sua largura, de modo a não deformar por completo as letras. Na maioria dos casos, como é possível observar nas figuras, certas letras são mais complexas de esticar, sendo, por isso, evitadas, como as letras A e X, por exigirem uma maior atenção por parte do *designer*. Assume-se que estas letras ao alargar tornam-se mais complexas, por facilmente poderem destoar um glifo do resto da fonte, ou torná-lo confundível com outros glifos.

De qualquer modo, também foram encontrados projetos onde os *designer*s desenvolveram fontes completas, que podem ser observadas nas figuras seguintes. Com estes projetos em específico, é possível fazer um levantamento das várias soluções possíveis para cada glifo e como cada um se enquadra em toda a fonte.

Como já foi discutido, a *web* está a tornar-se um meio de comunicação cada vez mais importante, por isso, a devida divulgação de um projeto ou evento, deve ser feito também na *web*. A falta de ferramentas na *web* que permitem uma maior maleabilidade da forma da tipografia, levou-me a querer desenvolver uma ferramenta que permite adicionar uma maior maleabilidade e interatividade.

Figura 54 : Cindie Mono, uma fonte com múltiplas larguras, desenhada por Lewis McGuffie (McGuffie, 2017).

Figura 55 : Fonte desenvolvida para a identidade para o projeto "attachez vous à la cause" (Breton-Allaire, Guilleaume et al. 2017).

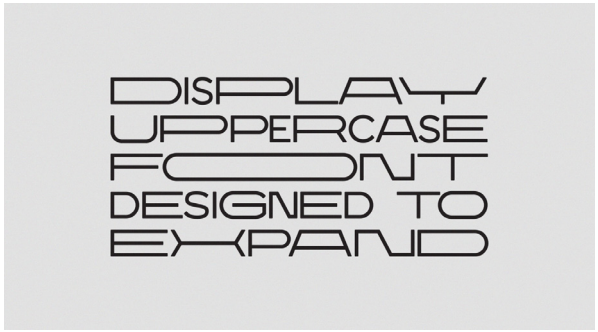


Figura 56 : Fonte personalizada para a marca de moda Épiq of Africa, por Pedro Cruz (Cruz, 2017).

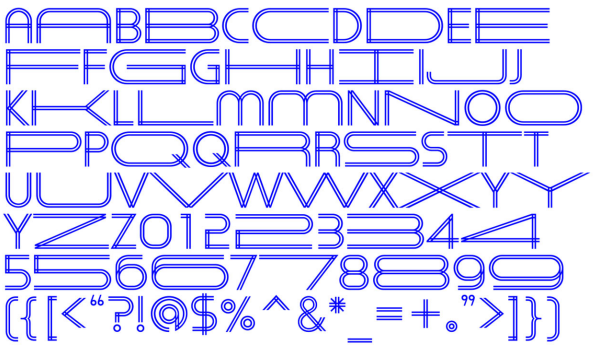


Figura 57 : Natasha Jen e a sua equipa, do estúdio Pentagram, criaram esta fonte para o evento Heritage Ball 2015 (Pentagram, 2017).

Desde o início, foi assumida a preocupação, de que fosse possível aplicar o *script* a qualquer fonte, de forma a que mais utilizadores fiquem interessados e como *designer*s usem a ferramenta para ajudar num projeto específico com a fonte que pretendem. Sendo assim, nesta dissertação, assume-se a preocupação pela forma que a fonte iria apresentar, sendo desenvolvida uma fonte extra larga, mas esta foi desenvolvida de forma a testar os resultados do *script*.

4.3 Tecnologias

Após um levantamento das possibilidades tecnológicas usadas na *web*, para permitir uma maior variabilidade da tipografia, foram encontradas três abordagens possíveis.

A abordagem predominante, para criar o efeito de variabilidade numa fonte na *web*, foi o uso de SVG, que pode ser diretamente exportado em programas como Adobe Illustrator, editor de imagens vetoriais. Assim, o que é apresentado na *web*, não se trata de fontes, mas de imagens vetoriais destas.

Outra abordagem encontrada foi o uso de um dos *script*s Jsfont.js ou OpenType.js , onde é possível aceder e modificar as tabelas de dados dos ficheiros fonte. Com estes *scripts*, a partir de um ficheiro fonte, é possível aceder à forma da letra e desenhá-la no HTML, a partir de um CANVAS. Num trabalho relacionado com esta dissertação, "Live Font Interpolation on the Web", de Andrew Johnson, é possível ver este *script*, onde os dados são alterados de forma a variar o peso da fonte, contendo em si, uma infinidade de pesos apenas com dois ficheiros fonte.

Por último, foi encontrada uma abordagem onde a fonte era desenhada a partir de CSS, a tipografia Curtis desenhada por David DeSandro. Este desenhou a fonte através de atributos simples do CSS, como o `background-color`, `border width`, `border-radius` e posicionamento `absolute`/`relative`. Esta tipografia também existe no formato fonte, mas David DeSandro quis testar se seria possível desenhar as formas das letras, apenas com CSS.

A abordagem tomada foi a do uso dos *scripts*, por duas razões, já anteriormente apresentadas nesta dissertação. O CSS foi a primeira abordagem excluída, pois, como já foi explicado, o CSS é primeiramente uma linguagem que trata da aparência de elementos HTML e o JavaScript a linguagem que trata da interatividade e mudança de elementos de uma página. Assim, para obter os resultados pretendidos em CSS, seria necessário desenhar a fonte em CSS e posteriormente alterar a sua forma em JavaScript, tornando-se uma abordagem pouco prática e acrescentando complexidade ao problema. Por último, foi escolhido o *script* Jsfont.js ou OpenType.js, pois estes oferecem a opção de poderem funcionar com qualquer ficheiro fonte e por conterem em si, já uma base de funções que facilita o trabalho sobre a informação das fontes.

4.4 Software

Numa fase inicial, foram testados os diferentes softwares possíveis para criar a fonte e simultaneamente testados, se era possível obter a sua informação na *web*, através dos *script*s JSFont.js e OpenType.js.

Após alguns testes as fontes exportadas pelo Glyphs não conseguiam ser lidas pelos *scripts*, devido à forma como eram codificadas, por isso a construção da fonte foi realizada no FontLab. Em relação ao *script* a usar para aceder aos dados da fonte na *web*, optou-se pelo OpenType.js por oferecer um conjunto de funções nele incluídas, que facilitam o acesso e sua posterior alteração, a todo o tipo de informações necessárias para implementar o projeto prático desta dissertação.

4.5 Desenvolvimento

Para o funcionamento do *script* ser possível, foi preciso que este atuasse sobre duas fontes *master*, em que cada uma delas define o limite máximo e mínimo do eixo da largura (Figura 58). Como já é possível encontrar *software* que desenvolve fontes, ao determinar o máximo e mínimo de um eixo para gerar novas fontes intermédias através da sua interpolação, assegura-se que todos os resultados intermédios não deformam a fonte de forma inesperada.

Assim, o funcionamento do *script* consiste em receber as duas fontes *master* e com auxílio do OpenType.js, é processada a informação das duas e determinada a largura a aplicar ao texto, de forma a que a sua largura se ajuste ao tamanho do ecrã, como está ilustrado na figura 59 e 60.

Figura 58 : Duas fontes, cada uma uma *master* que define um extremo da largura.

O cálculo do largura da fonte é feito através do mínimo e do máximo da largura da fonte, definidos por cada *master*, mostrados na figura 58. Para fazer os cálculos, assumiu-se que a fonte com a largura mínima representa uma percentagem de zero e a fonte com a largura máxima uma percentagem de 1. Assim dada a largura do ecrã, é calculada a percentagem que a fonte tem de assumir para ocupar o espaço pretendido, como é ilustrado no gráfico da figura 61.

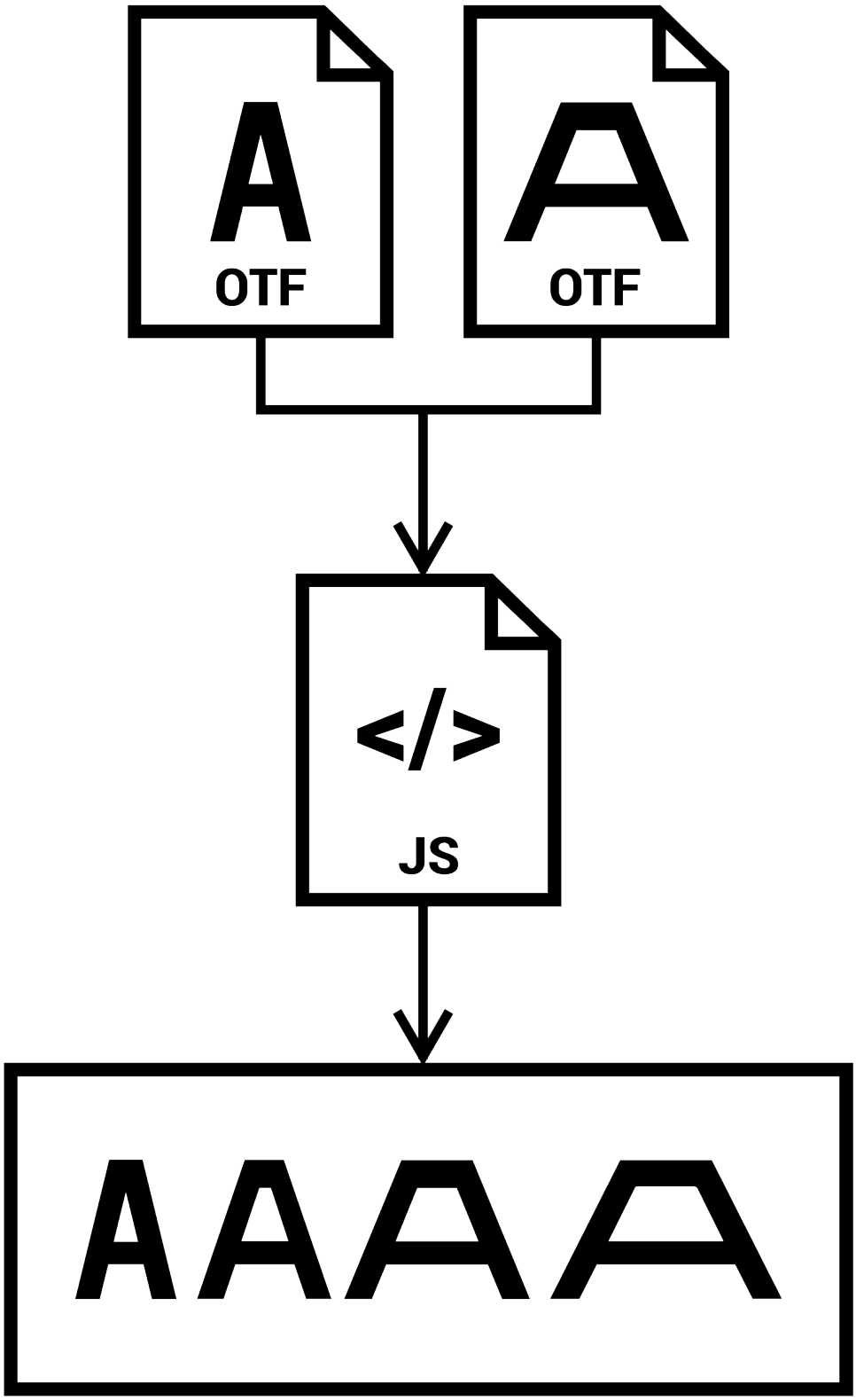


Figura 59 : Esquema do funcionamento do *script*.

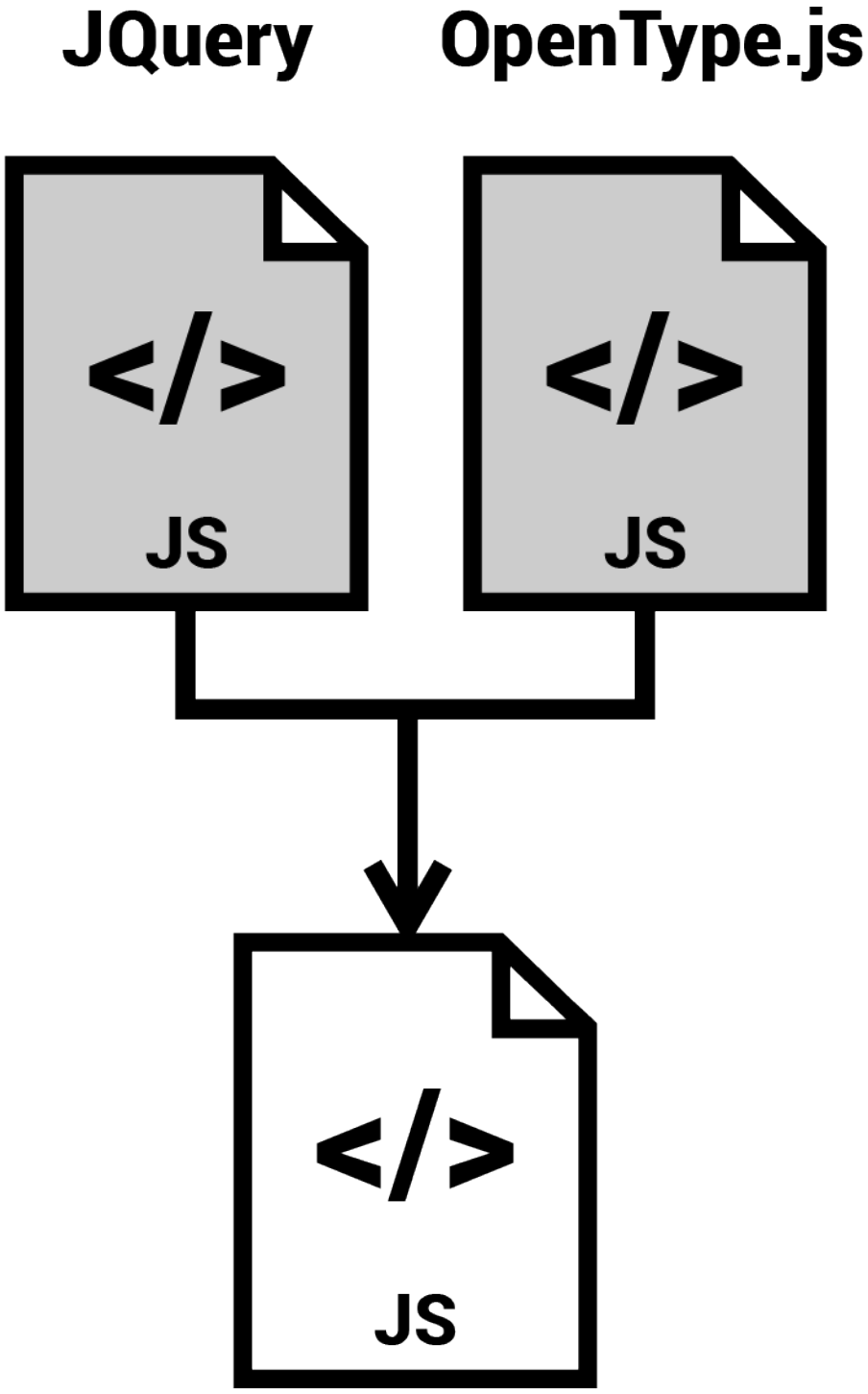


Figura 60 : Esquema do auxílio dos *script* Jquery e OpenType.js para o script poder funcionar.

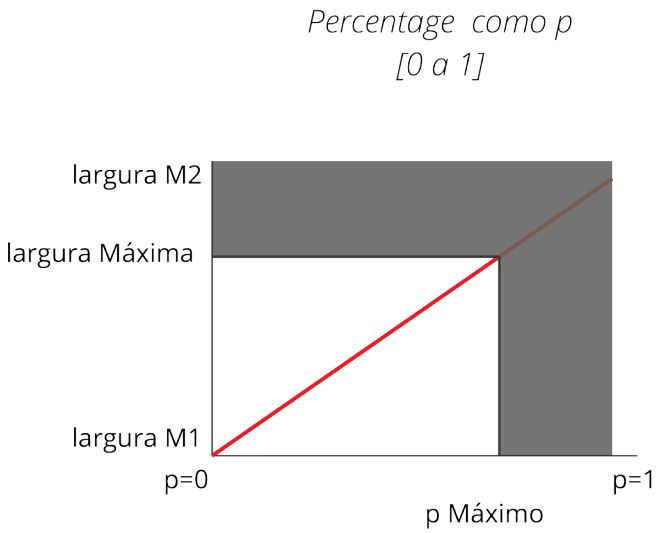


Figura 61 : Gráfico que explica como a largura da fonte é calculada, tendo em conta as duas *masters*.

Ao desenvolver o *script*, foram encontrados problemas quando a fonte era esticada. Em glifos, ao esticar a largura, houve alterações na coordenada y (altura do ponto) e, em certas iterações, os seus intermédios sofreram deformações. Como pode ser observado na figura 63, na letra N em interpolações intermédias, a diagonal torna-se mais larga do que o pretendido, isto porque as coordenadas y aumentam linearmente como a coordenadas x, sendo que o resultado pretendido é o que está apresentado na figura 64.

Para resolver este problema e de forma a que todos os cálculos intermédios em todos os glifos não sofram deformações não desejadas, concluiu-se que quando as coordenadas y de um ponto são diferentes entre as duas masters, esta coordenada deve crescer exponencialmente ao contrário das coordenadas x, que crescem linearmente, como é representado na figura 62.

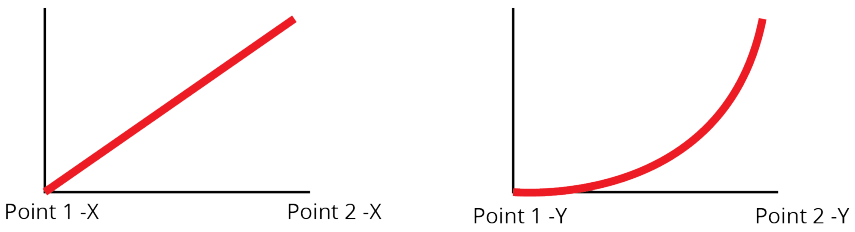


Figura 62 : Gráfico elucidativo do cálculo para as coordenadas x e y.

Figura 63 : Interpolação onde as coordenadas *y* são calculadas linearmente.

Figura 64 : Interpolação onde as coordenadas *y* são calculadas exponencialmente.

4.5.1 FONTE

Como já anteriormente referido, foi necessário criar uma versão extra larga da fonte escolhida, para que seja possível testar o *script*, tirando partido da fonte Reglo de Sebastien Sanfilippo. Algumas alterações foram realizadas em algumas letras, que podem ser observadas na figura 65, de forma a que a sua transição para a fonte mais larga, seja o mais fácil possível e que todos os intermédios gerados entre as duas fontes não aparentem estar deformados.

Figura 65 : Alterações feitas à Fonte Reglo, para facilitar criar a sua versão extra larga.

Na maioria das letras foi simples de criar a versão larga, pois consiste apenas em alongar as horizontais já existentes, como as letras P, B, F, etc, ou acrescentar horizontais entre curvas como as letras O, G, C, J, S e U. Ao contrastar letras que possuem diagonais, como as letras A, X, W e V, surgiram algumas dúvidas, por isso foi feito um levantamento das diferentes possibilidades.

Tendo em conta os exemplos práticos apresentados anteriormente nas fontes de inspiração, foram encontrados exemplos de fontes completas, onde são apresentadas várias soluções visuais possíveis para as letras que exigem uma maior atenção e cuidado.

Pode-se assumir que foram encontradas duas soluções. Uma delas seria alongar indiscriminadamente as diagonais, enquanto a outra seria não alongar as diagonais, mas acrescentar uma horizontal entre as diagonais, que sofreram o alongar da fonte. Foram encontrados problemas em ambas as soluções, se por um lado o alongar indiscriminado das diagonais destoava estas letras do resto da fonte, acrescentar uma horizontal para alongar a fonte, tornava certas letras difíceis de se diferenciarem de outras, como por exemplo a letra X, que pode ser observado na figura 66, torna-se confundível com a letra H e a letra V com a letra U.

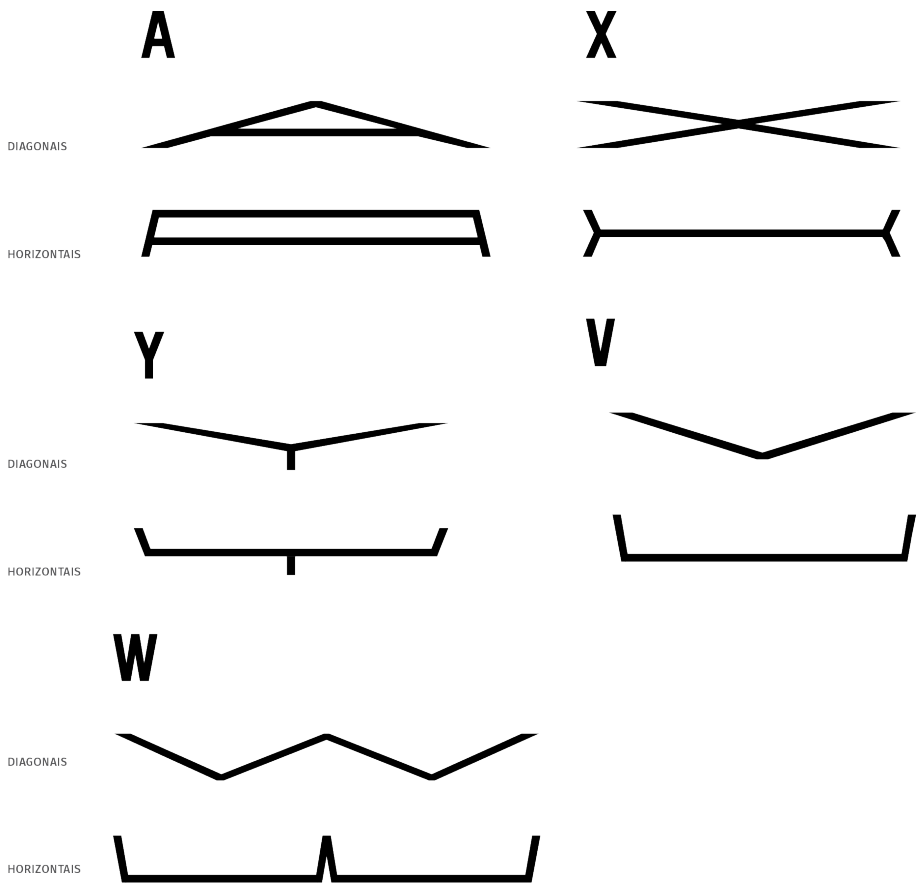
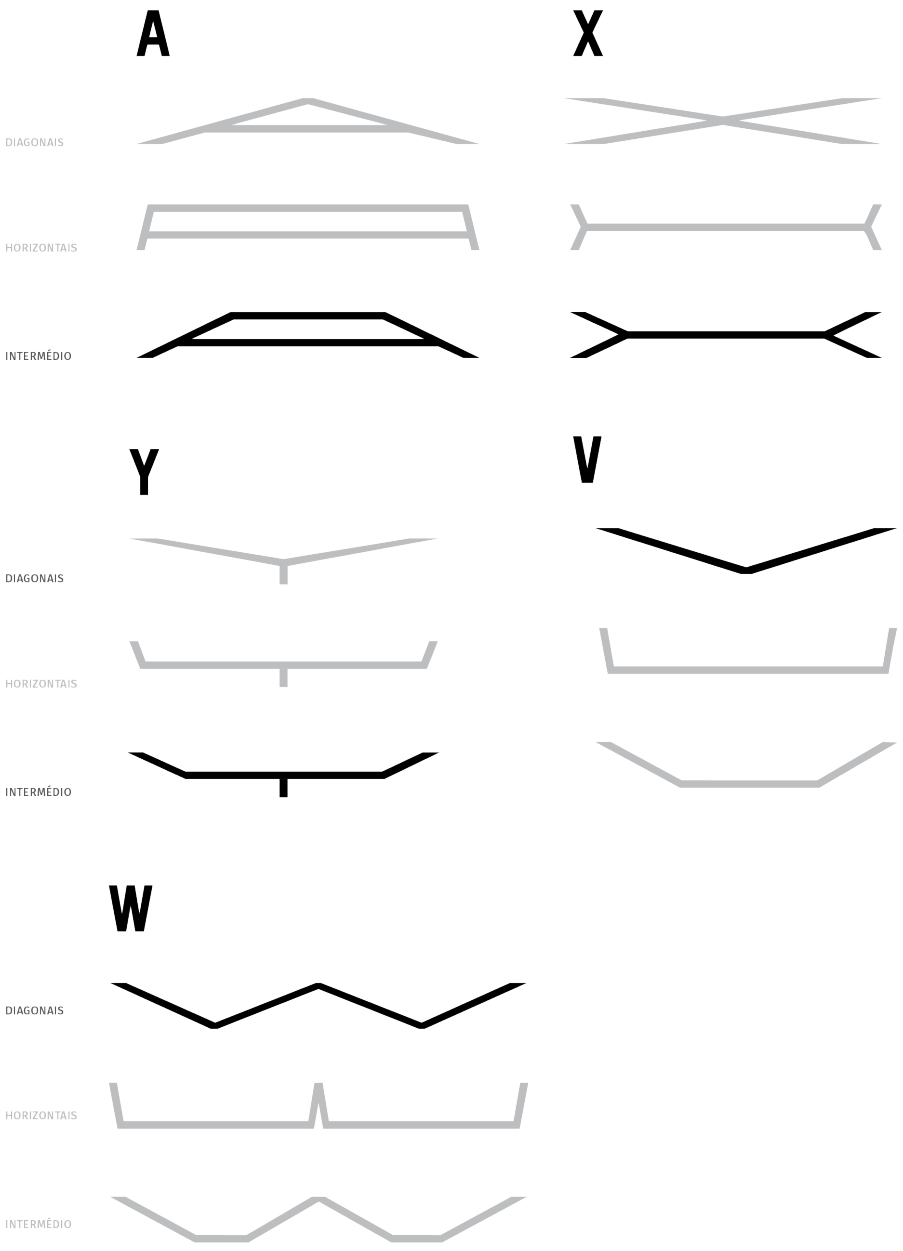


Figura 66 : Soluções possíveis para versão extra larga da Fonte Reglo.

Assim de forma a que toda a fonte seja o mais uniforme e legível possível, em certas letras, como a W e V optou-se apenas por alongar a diagonal, porque a sua versão intermédia tornava a letra inconsistente com o resto da fonte ou confundível com outras letras. Em letras como a A, X e Y optou-se por fazer uma versão intermédia das duas soluções, ou seja, é alongada ligeiramente a diagonal de modo a possuir o mesmo ângulo obtido pelas letras W e V e é acrescentada uma horizontal, como pode ser observado na figura 67.

Figura 67 : Soluções escolhidas para versão extra larga da Fonte Reglo.



4.5.2 OPENTYPE.JS

O opentype.js é um *script* em JavaScript escolhido para desenvolver o projeto prático desta dissertação, que permite aceder e modificar as informações devolvidas por um ficheiro fonte, contendo todos os pontos que definem um glifo, as suas caixas delimitadoras e o seu *kerning* (cf. Glossário).

O opentype.js contém três objetos JavaScript predominantes, que se tornam importantes na realização deste projeto: "`Font`", "`Glyph`" e "`Path`". O objeto "`Font`" contém toda a informação da fonte que foi escolhida. O objeto "`Glyph`" contém a informação detalhada de um glifo específico de uma dada fonte, indicando todos os pontos que o formam, a sua largura e as informações básicas (pode ser observado na imagem 68). O objeto "`Path`" é uma sequência de pontos que define cada glifo, sendo que as coordenadas já estão calculadas consoante a fonte, o texto e o tamanho a desenhar, como pode ser observado na figura 69.

O OpenType.js, também oferece um conjunto de funções que devolvem

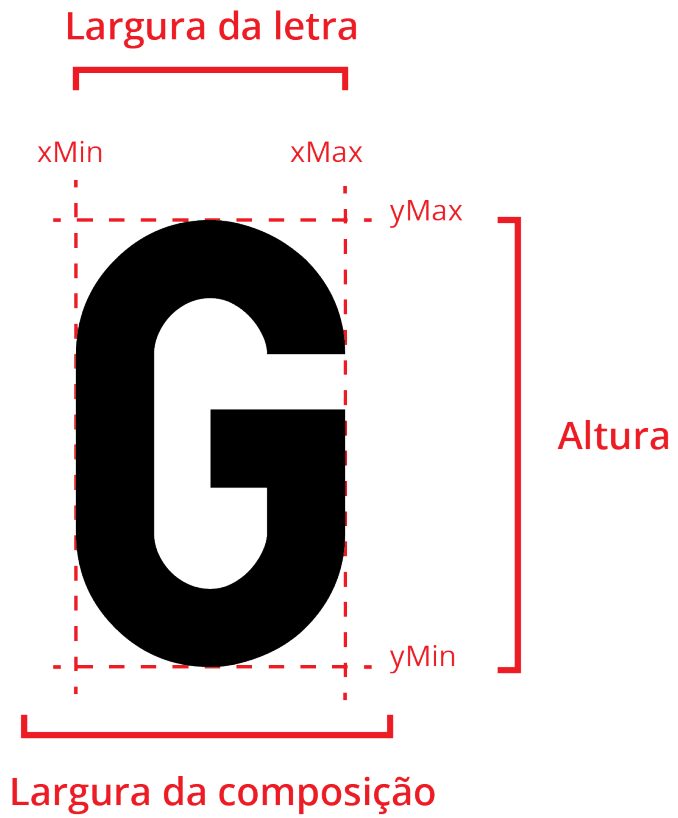


Figura 68 : Ilustração de informação retirada de um objecto `Glyph` pelo OpenType.JS.

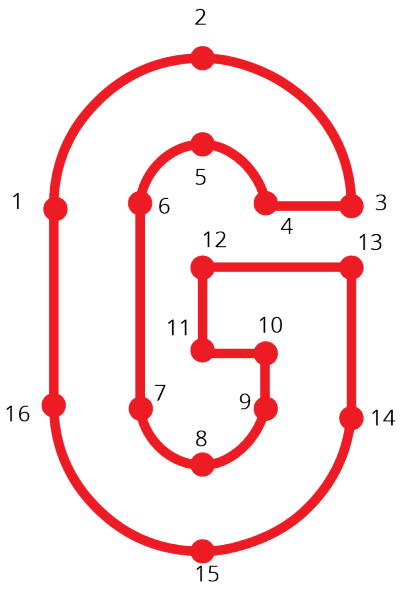


Figura 69 : Ilustração de informação retirada de um objecto `Path` pelo OpenType.JS.

as diferentes informações, referentes a cada objeto. As mais usadas foram as seguintes:

```
Font.stringToGlyphs(text)
Font.getKerningValue(leftGlyph, rightGlyph)
Font.getPath(text, x, y, fontSize, options)
Font.draw(canvas)
```

A função "`Font.stringToGlyphs(text)`", perante um texto a desenhar, devolve um conjunto de glifos que o definem. A função "`Font.getKerningValue (leftGlyph, rightGlyph)`" devolve o valor do kerning entre os dois glifos indicados. A função "`Font.getPath(text,...)`" devolve o caminho do texto para desenhar, depois de inseridas todas as especificações necessárias. A função "`Path.draw(canvas)`" desenha o caminho no canvas indicado.

De forma a ir ao encontro do que foi proposto, foi necessário acrescentar certas funções ao opentype.js. O qual converte todo o texto indicado num caminho contínuo, precisando de ter um maior controlo do caminho de cada glifo individualmente, para posteriormente, poderem ser alterados (mais à frente será especificado em que situações foi preciso acrescentar essas determinadas funções). Para além disso, foi acrescentada uma função que devolve a largura total do texto inserido.

```
Font.checkWords()
Font.getPathLetter()
Font.checkWordsLetter()
```

A função "`Font.checkWords()`", soma a largura de cada glifo, devolvendo a largura total de uma palavra. A função "`Font.getPathLetter()`", devolve array (cf. Glossário) de caminhos sendo que cada iteração do array define um caminho para cada glifo. A função "`Font.checkWordsLetters()`", da mesma forma que função "`Font.checkWords()`", devolve a largura total de cada glifo, a diferença é que em vez de devolver a largura total da palavra, devolve um array com a largura para cada letra.

4.5.3 FUNCIONAMENTO

De forma a que o *script* cumpra o que foi proposto nesta dissertação e para que o máximo de código redundante fosse diminuído, as suas funções foram organizadas da seguinte maneira — o *script* é inicializado no construtor [3], com a definição de certos parâmetro obrigatórios: o caminho para a fonte condensada ou regular e extendida, um texto para correr e definir o identificador do CANVAS onde a fonte vai ser desenhada. O identificador do CANVAS é definido por um atributo em HTML, como é mostrado:

```
<canvas id="canvas_id"> </canvas>
```

[3] - A linguagem de JavaScript não contém assumidamente um construtor, mas por via de facilitar a comunicação do funcionamento do *script* a inicialização do *script* vai ser intitulado construtor, por servir objetivos semelhantes às de um construtor em outras linguagens de programação.

Ainda no construtor[3], é verificado se os parâmetros obrigatórios estão bem definidos e são definidos os parâmetros não obrigatórios, com os seus valore padrão. Mais à frente, serão listados todos os parâmetros possíveis a definir no construtor[3].

Após o *script* ser inicializado, a função "`runSuiType()`" é corrida. Se ocorrer algum erro no construtor[3], serão mostrados todos os erros, se não, será verificado se as fontes são validadas, e a seguir, o OpenType.js converte-as num objeto `Font` que, como já foi dito, vai permitir aceder em JavaScript a todas as informações necessárias das fontes.

Depois de serem verificadas as fontes, tendo em conta os parâmetros definidos (tamanho do CANVAS, texto, tamanho do texto), é calculada a percentagem (definida como limite[4]), entre as duas fontes master que o texto deve assumir no canvas, como anteriormente foi mostrado no gráfico da figura 61.

Quando os limites[4] para cada glifo forem calculados, é corrida a função de interpolação que, resumidamente, trata de desenhar as letras consoante a percentagem anteriormente calculada.

Este funcionamento está ilustrado no esquema da figura 70.

[4] - Valor de 0 a 1, que define a percentagem de cada master que cada glifo ou palavra vai assumir. Ver figura 61.

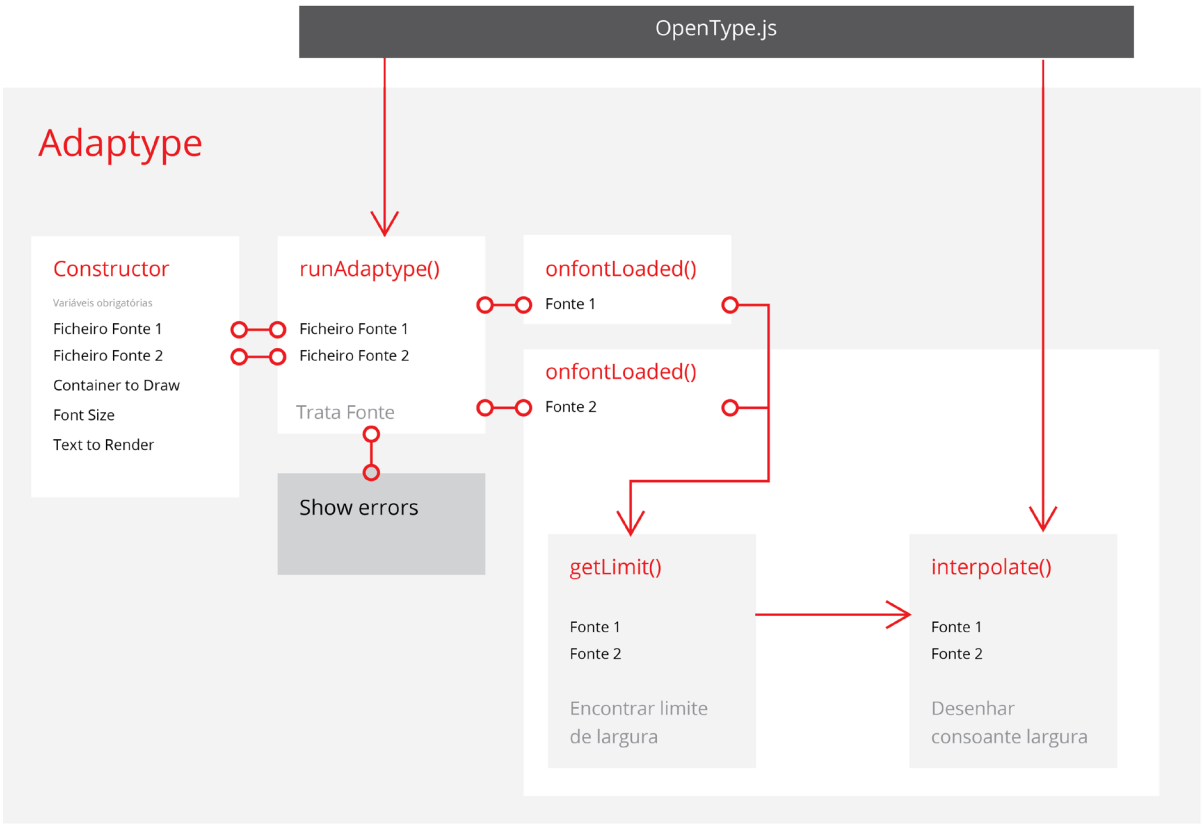


Figura 70 : Esquema elucidativo do funcionamento interno do *script*.

A função de `interpolate()` é onde o *script* muda a forma da fonte e desenha o resultado no CANVAS. Inicialmente, são obtidos os caminhos que definem o texto das duas fontes, ou seja, devolve uma lista de todos os pontos e das suas coordenadas. Isto é possível através da função do OpenType.js `Font.getPath()`. Com o limite[4] definido, é calculado ponto a ponto as novas coordenadas tendo em conta o limite[4], para serem desenhadas. O funcionamento interno da função `interpolate()` está ilustrado no esquema da figura 71.

A função `doSnap()` — é a função que vai alterar as coordenadas dos pontos de cada glifo de forma a que estes alterem a sua largura. Esta função, como está ilustrado no esquema da figura 72, recebe os dois caminhos das duas fontes e para cada ponto se as coordenadas são diferentes entre os dois caminhos, ou seja, o ponto é alterado entre as duas fontes e é calculado o novo valor do ponto, consoante o limite[4]. Como foi anteriormente referido, se a coordenada x for diferente, é feito um cálculo linear, se a coordenada y for diferente é feito um cálculo exponencial.

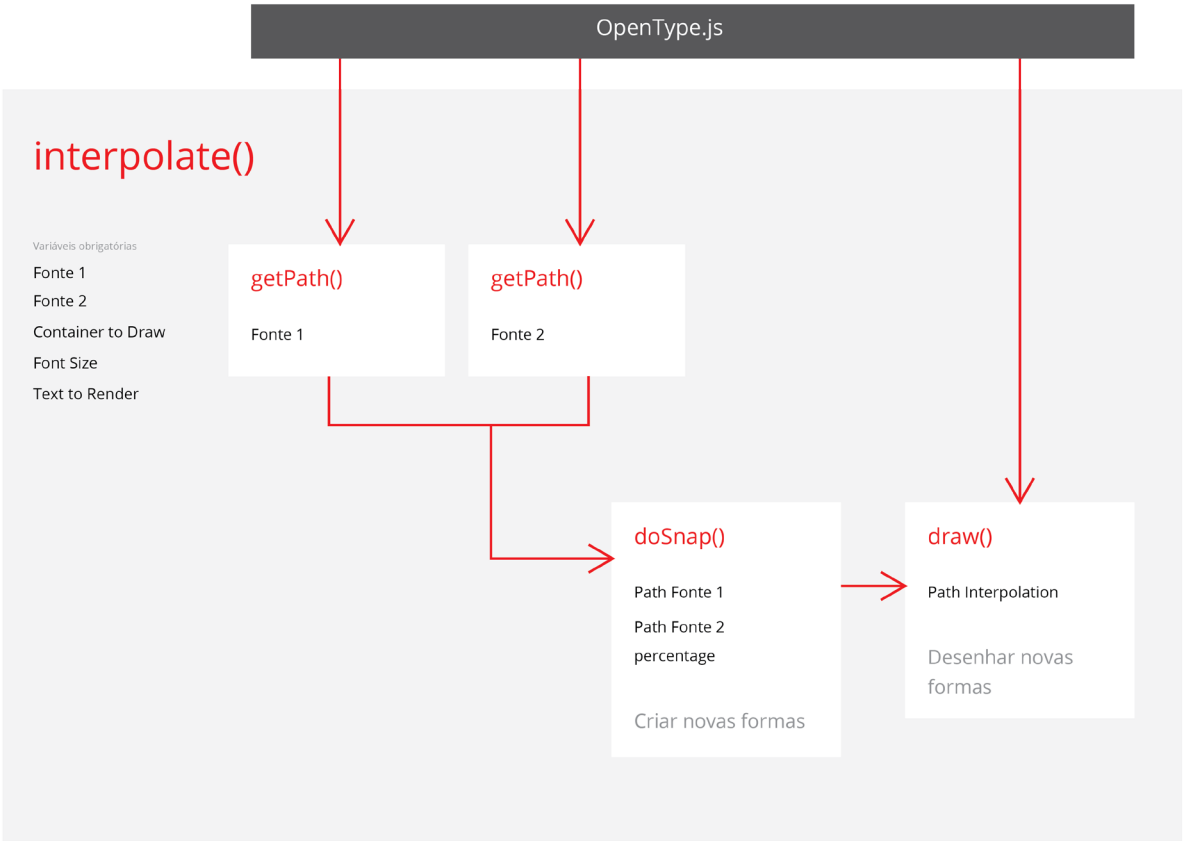


Figura 71 : Esquema elucidativo do funcionamento interno da função `interpolate()`.

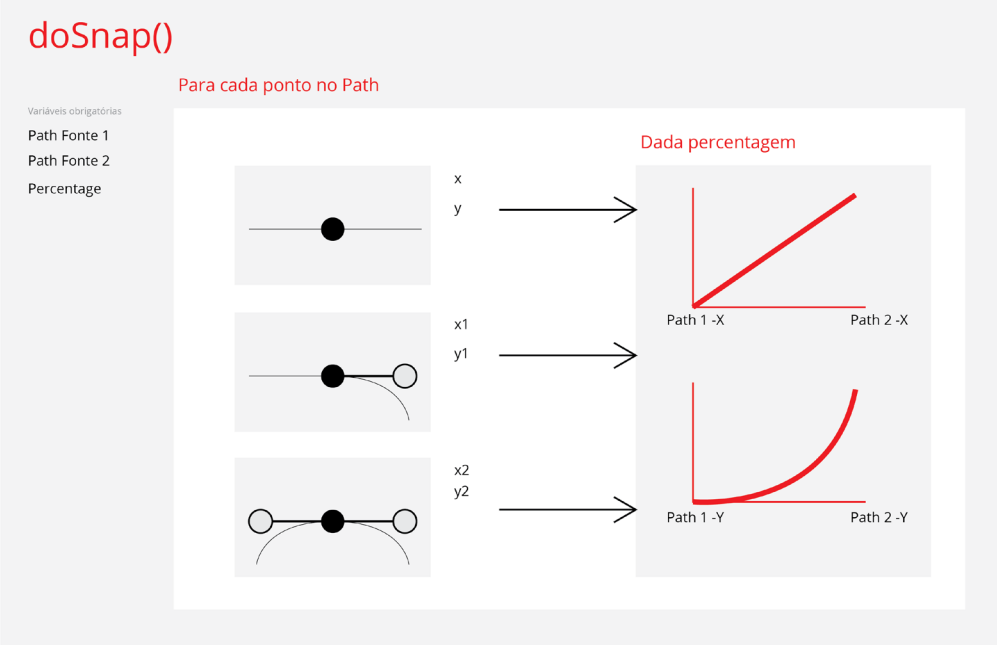


Figura 72 : Esquema elucidativo do funcionamento interno da função `doSnap()`.

4.5.4 CÁLCULOS

De modo geral, o cálculo da largura é feito da seguinte maneira, oferecidos todos os parâmetros, como o texto a desenhar e o tamanho da letra, é possível calcular a largura mínima que o texto vai ocupar, ou seja, a largura da fonte *Master* 1. Assim, ao subtrair o tamanho do ecrã ao tamanho mínimo do texto, é encontrado o excesso de largura do ecrã, que uma ou várias letras vão assumir na sua largura, como pode ser observado na figura 73.

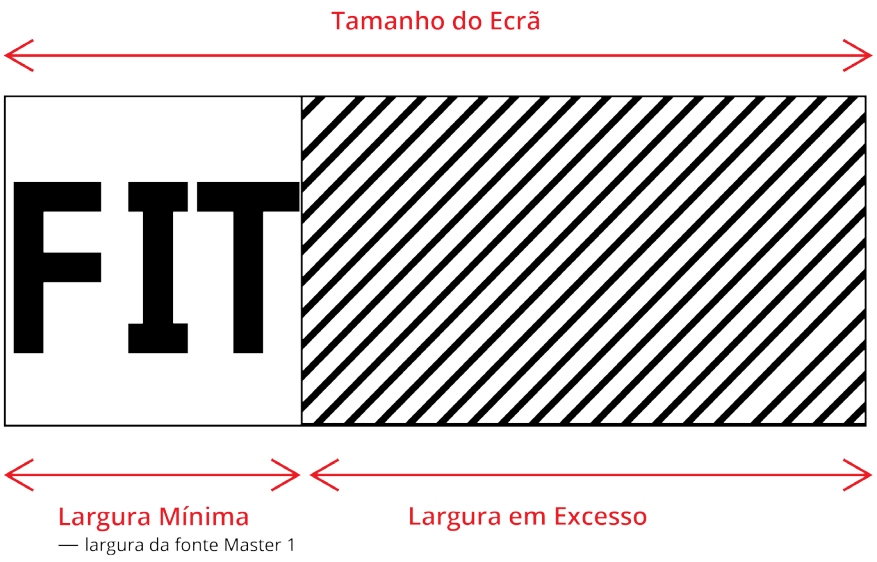


Figura 73 : Ilustração elucidativa do processo para alargar a fonte.

Ao encontrar o novo comprimento a aplicar às letras, é necessário converter essa medida em percentagem, ou seja, definir a percentagem que a largura de cada master representa, como é ilustrado nas figuras 74 e 75, sendo que o valor 0 representa a largura exacta da *Master* 1, o valor 1 a largura da *Master* 2.

A conversão da largura da fonte, para percentagem, como foi representado no gráfico da figura 75, é feita linearmente, ou seja, pode ser definida e calculada qualquer percentagem, a partir da equação: $y=mx+b$, que define funções lineares.

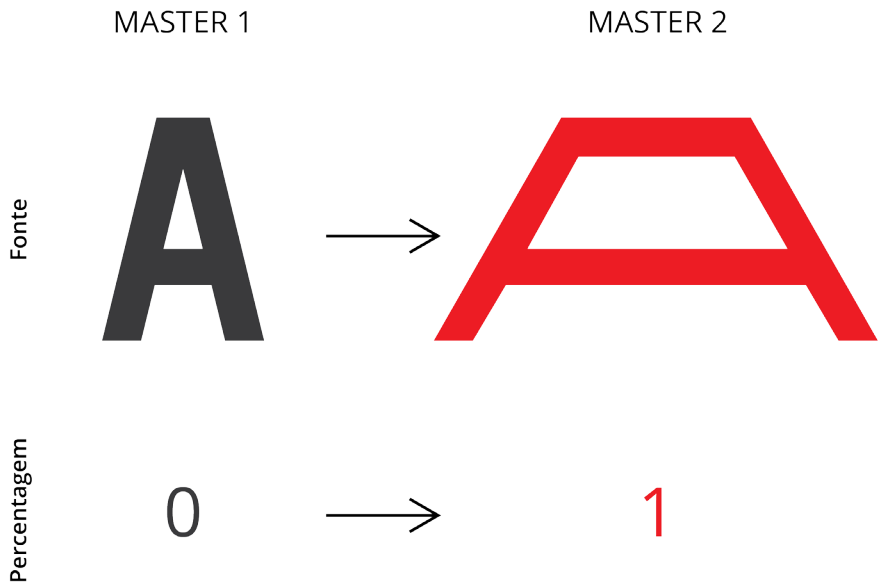


Figura 74 : Ilustração elucidativa do processo para alargar a fonte.

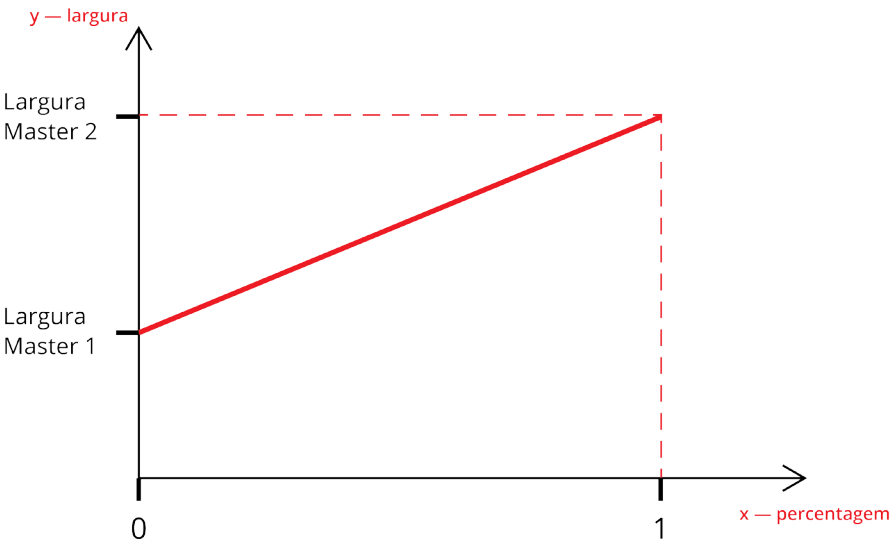


Figura 75 : Ilustração do gráfico, onde a relação entre largura e percentagem das duas *Masters*, é linear.

A partir dos dois pontos já obtidos (*(0, Largura Master 1)* e (*1, Largura Master 2)*), apresentados no gráfico da figura 75, é possível calcular as variáveis da equação que estão por definir, o *b* e o *m*, como é exemplificado nos cálculos seguintes. Isto para poder reunir todas a variáveis da equação necessárias para poder calcular a percentagem, para qualquer comprimento pretendido.

M1 —Master 1

M2— Master 2

Pontos:

P1(0 , M1)

P2(1 , M2)

$y = mx+b$

$M1 = m*0 +b$

$b = M1$

$M2 = m*1 +M1$

$m = M2 - M1$

Função Final:

$y = (M2- M1)*x+M1$

$x = (y - M1)/(M2 - M1)$

Quando a percentagem para uma dada largura, é encontrada, são calculadas as coordenadas para cada ponto que define cada glifo. Assim, se os valores das coordenadas do mesmo ponto nas duas *Masters*, forem diferentes, é calculado o novo valor, sendo que o cálculo é feito da seguinte maneira: inicialmente é calculada a diferença da coordenada de uma *Master* para a outra, e multiplicado esse valor pela percentagem obtida anteriormente; de seguida, a esse valor é somado a coordenada mínima, ou seja, a coordenada desse ponto da *Master* 1.

V — Valor

CM1 — Coordenada da Master 1

CM2 — Coordenada da Master 2

p — percentagem obtida anteriormente

Função:

$V = ((CM2 - CM1)* p)+CM1$

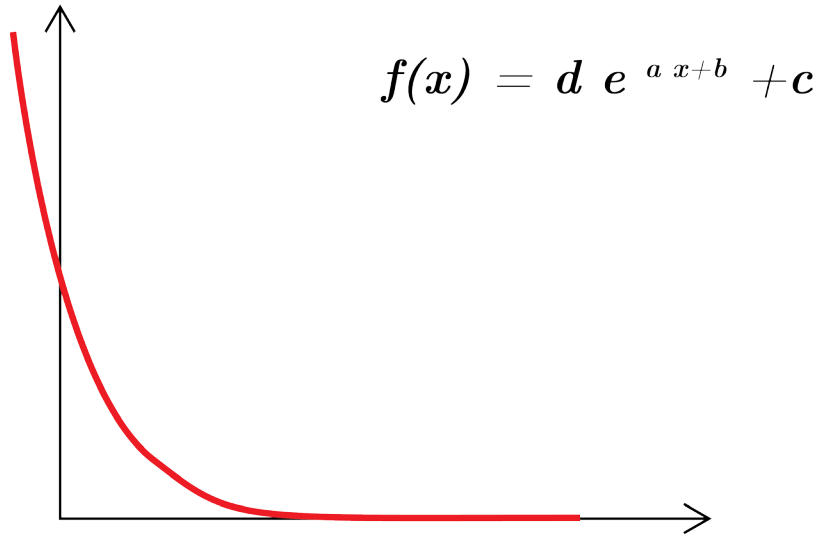
Resumindo, o cálculo para redesenhar a forma de cada letra de modo a mudar a sua largura, consiste em três passos. Primeiro, é encontrado o excesso de comprimento no tamanho da janela que, posteriormente, as letras têm de assumir; de seguida é calculada a percentagem entre as duas *masters* que esse comprimento representa e por último são calculados os novos valores para cada ponto que define cada letra.

Como foi anteriormente referido, de forma a que todos os intermédios de cada letra gerados entre as duas fontes, seja correta, chegou-se à conclusão que, se as coordenadas y de qualquer ponto são diferentes entre as duas fontes *master*, o valor do comprimento tem de ser calculado exponencialmente, em contraste com as coordenadas x, que são calculadas linearmente.

Em relação ao processo anteriormente referido, apenas um dos passos é ligeiramente alterado, sendo que a equação usada para calcular as percentagens de cada *master*, é alterada para uma função exponencial, que pode ser definida como
$f(x) = d\ e^{a\,x+b} + c$, como ilustrado na figura 76.

Dada a equação base que define uma função exponencial, começou-se por perceber o que cada variável na função representa e como estas se podem ajustar aos dados oferecidos pela fonte.

Figura 76 : Gráfico e equação que define uma função exponencial.



A variável c representa uma assíntota horizontal associada à função, ilustrada no gráfico da figura 77, ou seja, esta variável define o valor que a função ao longo do eixo x está a tender, sem nunca alcançar esse valor. Assim, a variável c, no processo de conversão da largura entre as duas *masters*, é definido pela coordenada da *Master* 2, extra larga.

Por outro lado, a variável b define um avanço no eixo de x, ilustrado no gráfico da figura 78, que no processo em concreto, não se sentiu necessidade de utilizar, por isso foi definida como 0.

A variável a representa a rapidez da transição entre valores, ou seja, se o valor de a for grande, mais rápida é a transição entre os valores de $f(x)$, mostrando uma curva mais acentuada, se o valor de a for pequeno, a transição entre os valores é mais lenta, mostrando uma curva mais suave, como é ilustrado no gráfico da figura 79. Por outro lado, o sinal de a, positivo ou negativo, define a orientação vertical da função.

A variável d, será definida a partir das variáveis anteriores, mas é possível concluir que o seu sinal, positivo ou negativo indica a orientação horizontal da função, como é ilustrado no gráfico da figura 80.

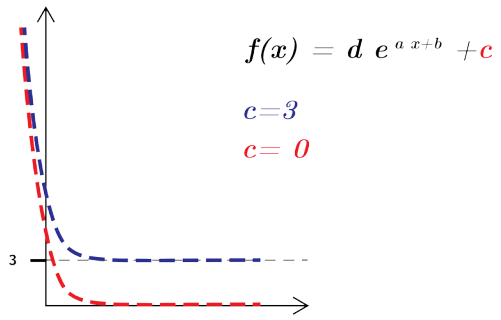


Figura 77 : Função exponencial, variável c.

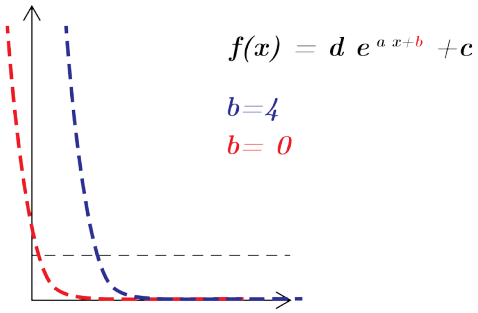


Figura 78 : Função exponencial, variável b.

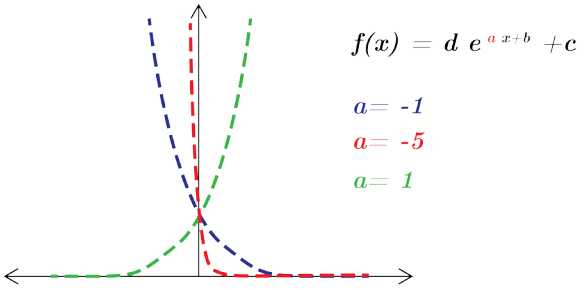


Figura 79 : Função exponencial, variável a.

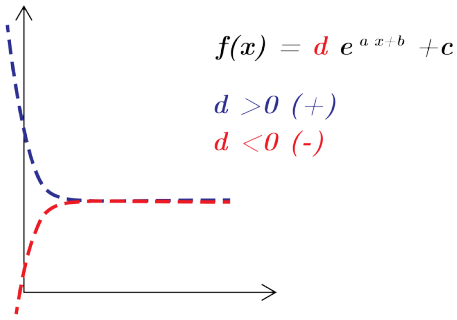


Figura 80 : Função exponencial, variável d.

Entendidas todas as variáveis, é possível definir a equação usada para calcular a percentagem entre *masters* (Figura 81). Assim, a coordenada a foi definida com um valor grande, para que a transição entre valores seja rápida, de forma a ajustar-se melhor à fonte e o seu sinal é negativo para que à medida que o valor x cresça, os valores tendam para uma assíntota e não o oposto. A variável b, como foi anteriormente referido é desnecessária, por isso é definida como 0.

A assíntota horizontal é facilmente definida, pois como que é pretendido que o valor das coordenadas tendam para os valores da Fonte *Master* 2, a variável c é definida pelos valores das suas coordenadas. A coordenada d é possível encontrar-se, através da decomposição da função, com as variáveis já definidas, como mostrado na figura 82.

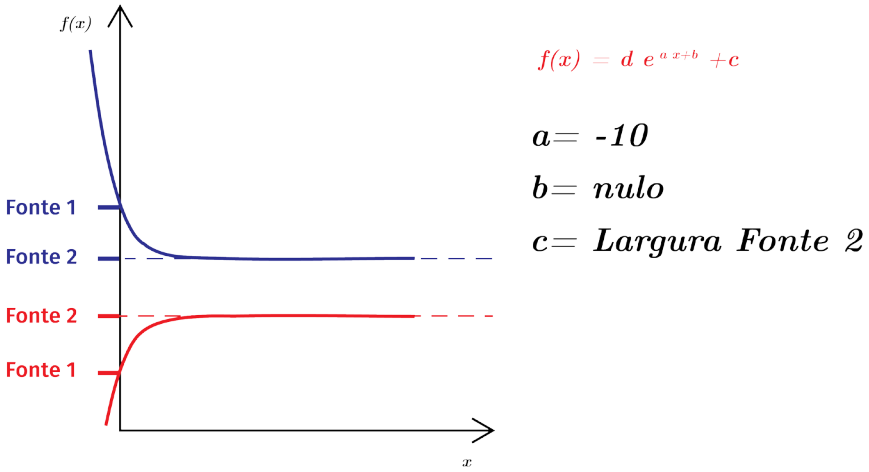


Figura 81 : Função exponencial, aplicada ao projeto.

$$f(x) = d\ e^{a\ x+b} + c$$

$$F1 = d\ e^{5*0} + F2$$

$$F1 = d\ e^{5*0} + F2$$

$$F1 - F2 = d\ e^{5*0}$$

$$d = (F1 - F2) * e^{5*0}$$

Figura 82 : Cálculos para encontrar função que define a variável d.

O cálculo realizado para encontrar as coordenadas x e y, permite que o processo de alargar a fonte, vá para além do máximo definido pela largura da fonte *Master* 2. Isto porque, dada a equação linear de cada coordenada x, facilmente é possível calcular os valores para além da *Master* 2 e como as coordenadas y são calculadas exponencialmente, dado o limite definido pela assíntota horizontal, a mudança das coordenadas a partir do máximo torna-se mínima, fazendo com que as diagonais não se deformem, como é ilustrado na figura 83. Ambas as figuras 83 e 84, mostram a preto o máximo definido pela *Master* 2 e a vermelho o cálculo previsto para uma largura superior à da *master*. Assim, se o cálculo das coordenadas y, for feito exponencialmente, os valores das coordenadas y estagnam, evitando deformações nas diagonais dos glifos, obtendo os resultados da figura 84.

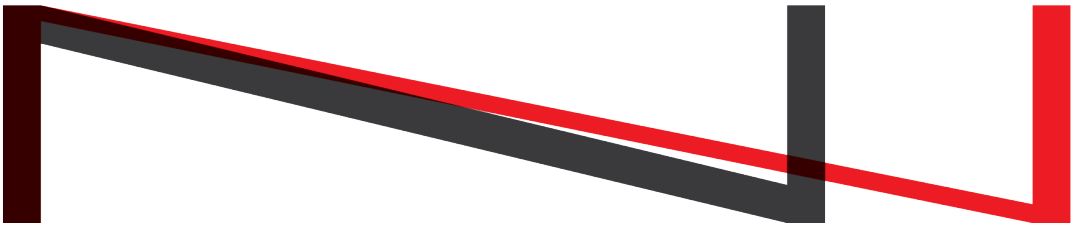


Figura 83 : Resultado da fonte para além da largura da *Master* 2, com cálculo das coordenadas *y*, linear.

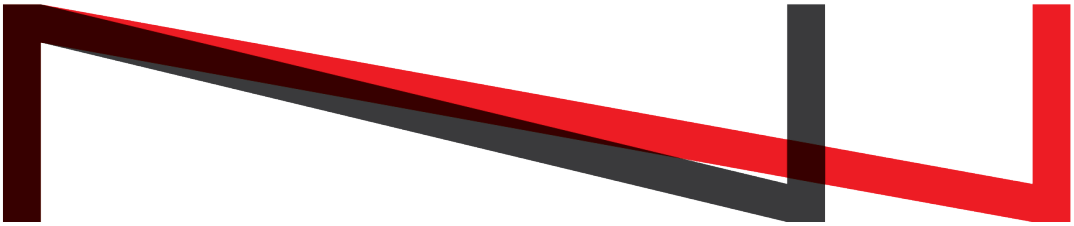


Figura 84 : Resultado da fonte para além da largura da *Master* 2, com cálculo das coordenadas *y*, exponencial.

4.5.5 DIFICULDADES ENFRENTADAS

— DEFORMAÇÕES

O *script* para calcular a nova largura do texto corre todos os pontos de cada glifo e compara os seus valores entre a fonte condensada e larga, se esses valores forem diferentes calcula o novo valor para o ponto. Como o OpenType.js retorna cada glifo, como uma sequência de pontos sem ordem definida, torna-se importante que ambas as fontes tenham exatamente a mesma quantidade de pontos, se não o *script* deforma a fonte, indiscriminadamente, como pode ser visto na figura 85 e 86.

Figura 85 : Deformações na texto, quando as duas fontes carregadas, não têm exactamente a mesma quantidade de pontos.

Figura 86 : Deformações na texto, quando as duas fontes carregadas, não têm exactamente a mesma quantidade de pontos.

— POSIÇÃO DAS LETRAS

Como foi indicado anteriormente, foi necessário acrescentar funções ao *script* OpenType.js, de forma a ter um maior controlo sobre cada letra numa palavra. Isto para permitir diferentes modos de dividir a largura pelas letras, deste modo, nem todos as letras são alongadas igualmente.

Isto fez com que as letras tomassem posições erradas, pois o OpenType.js calcula a posição de uma letra como se todas as letras tivessem a mesma largura, apresentando o resultado da imagem 87. É possível observar que a letra T tem uma percentagem perto de *0*, a largura da fonte *Master* Regular, ou seja, a sua posição é calculada como se todas as letras anteriores na palavra, W e I, tivessem também a largura da fonte *Master* Regular, fazendo com que a letra T esteja numa posição mais recuada do que o suposto.

Isto levou que fosse necessário fazer um cálculo extra para posicionar bem cada glifo no ecrã. Assim, cada glifo foi colocado na posição zero, como mostra a figura 88, e depois tendo em conta as larguras das letras anteriores, as suas margens e o *kerning* foi calculada a sua posição correta, como se pode observar na imagem 89.

Figura 87 : Posição de cada glifo mal calculada.

Figura 88 : Posição de cada glifo colocada no ponto zero.

Figura 89 : Posição calculada consoante a largura dos glifos anteriores, suas caixas delimitadoras e *kerning*.

— RECÁLCULO AO MUDAR LARGURA

O *script* foi implementado de forma a que quando o tamanho do ecrã ou contentor que delimita o espaço a ocupar pelo CANVAS é alterado, a largura das letras tem de ser novamente calculadas e mostradas no ecrã. Para a maioria dos modos, a tarefa foi simples, consistindo apenas em calcular a nova largura para cada letra, atualizando cada coordenada de cada ponto. Por outro lado, o modo "random" exigiu um maior trabalho e mais cálculos.

De forma a esclarecer a complexidade acrescida do modo "random", é explicado detalhadamente o processo utilizado para recalcular a largura das letras. Primeiro, é encontrada a largura que cada letra ocupa, excluindo o espaço entre as letras (Figura 90), encontrando o espaço total que todas as letras ocupam no CANVAS (Figura 91). Assim, com o novo tamanho de ecrã, ao comparar com o tamanho do ecrã anterior, é encontrada a diferença entre os dois e essa diferença é dividida por todas as letras igualmente (Figura 92).

Na maioria dos modos, quando a largura da letra é igual ao seu mínimo, quer dizer que todas as outras letras também têm a sua largura mínima. Por outro lado, no modo "random", como as letras têm diferentes percentagens a indicar a largura, quando uma das letras alcança a sua largura mínima, nem todas as letras a alcançaram, sendo necessário que a largura extra não retirada à letra, tenha de ser retirada a outras, como está ilustrado nas figuras 93 e 94.

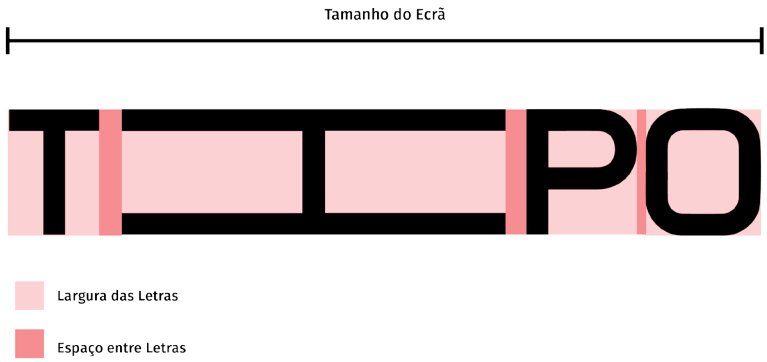


Figura 90 : Esquema do espaço usado por cada letra no CANVAS.

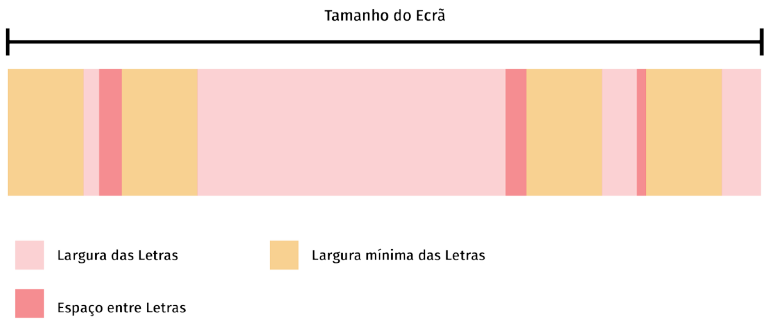


Figura 91 : Esquema do espaço usado por cada letra no CANVAS, ilustrando o espaço mínimo que as letras têm de ocupar.

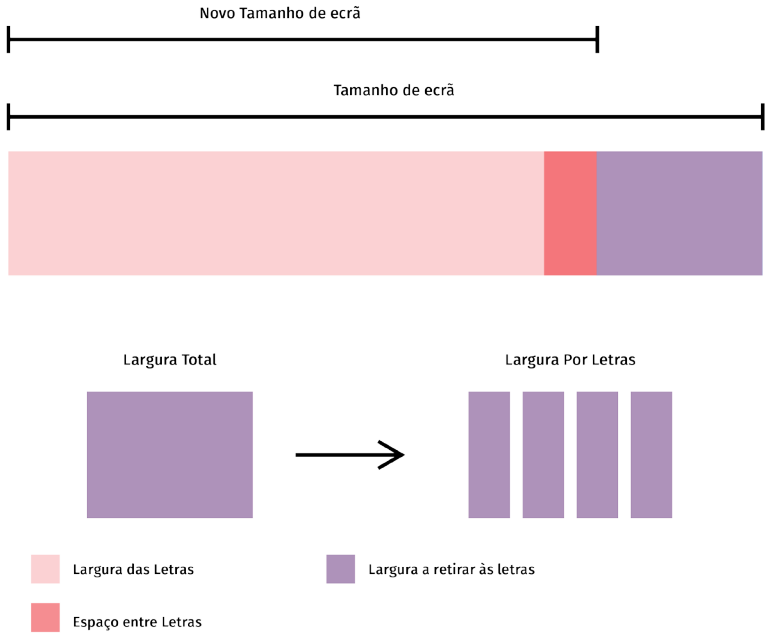


Figura 92 : Posição calculada consoante a largura dos glifos anteriores, suas caixas delimitadoras e *kerning*.

Figura 93 :Esquema de um glifo e quando a largura a retirar a uma letras, ultrapassa o permitido pelo mínimo da largura do glifo.

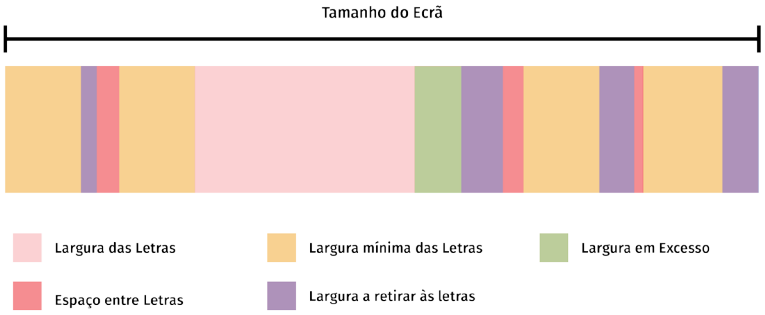


Figura 94 : Esquema que ilustra o processo aplicado no modo *random*, de forma a que o recalcular da largura do CANVAS seja correto.

4.6 Projeto Prático

Neste capítulo, são apresentados os vários conteúdos práticos desenvolvidos no âmbito desta dissertação prática. Resumidamente, consiste na construção de um fonte extra larga, que, como já foi explicado, tirando-se partido de uma fonte *opensouce* já existente, na construção de um *script* que calcula e apresenta a largura que uma fonte deve tomar, de forma a preencher o ecrã e por último, o trabalho de divulgação do *script*, que consiste na construção do *site* que apresenta a sua imagem.

4.6.1 FONTE

Como referido anteriormente, para os testes iniciais do *script*, foi usada a fonte *opensource* Reglo e esta foi ajustada e criada uma versão extra larga. A fonte completa pode ser observada na figura 95.

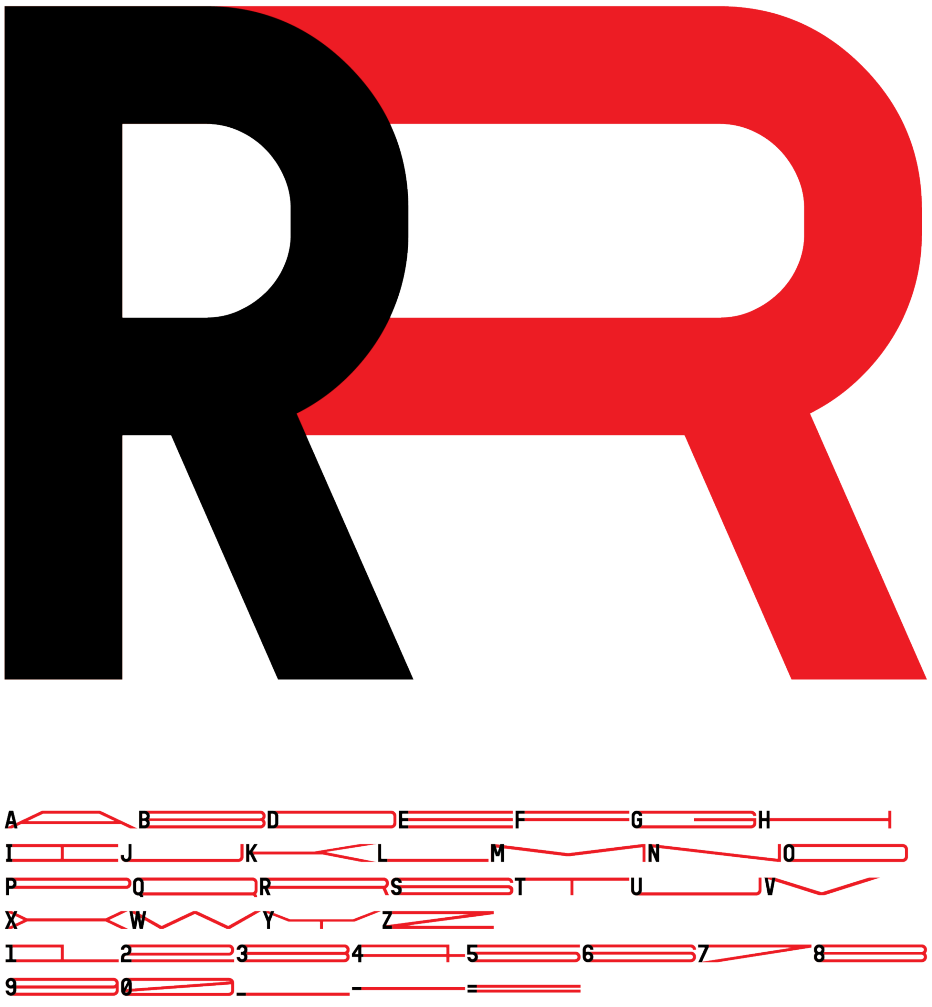


Figura 95 : Fonte Reglo com a versão original a preto e a vermelho a sua versão extra larga.

4.6.2 SCRIPT

— PARÂMETROS

O *script* foi construído de forma a que a sua posterior implementação seja o mais simples possível, sendo que a sua inicialização é feita com um construtor[3], ou seja, ao chamar a função são desencadeadas todas as funções necessárias para obter o resultado final desejado. A função pode ser inicializada da seguinte maneira:

```
var font= new AdapType({
    fonte1: "Path_to_Font_Regular",
    fonte2: "Path_to_Font_Expanded",
    textToRender: "Text",
    canvasID: "Canvas_Id",
});
```

Como foi explicado anteriormente, existem parâmetros obrigatórios e outros opcionais que têm valores padrão, evitando assim que o *web designer* tenha de definir uma grande quantidade de parâmetros desde início. Qualquer parâmetro pode ser acrescentado como é indicado no código seguinte. Todos os parâmetros podem podem ser observados na tabela seguinte.

Parâmetro	Padrão	Tipo	Descrição
`fontSize`	*100*	`number`	Define tamanho da letra a desenhar.
`mode`	*equal*	`string`	Define o modo como as letras vão ser alongadas, podendo ser: "`random`","`equal`", "`firstLetter`", "`lastLetter`", "`chosenLetter`", "`deform`."
`chosenLetter`	*null*	`array`	Define a posição das letras que vão ser alongadas no modo "`chosenLetter`".
`canvasWidth`	*100*	`number`	Define o tamanho do canvas.
`canvasWidthMode`	*percentage*	`string`	Indica o tipo de medida da largura do canvas, podendo ter os valores: *percentage, pixels.*
`color`	*black*	`color`	Cor para preencher a fonte.
`stoke`	*transparent*	`color`	Cor para preencher a linha delimitadora da fonte.
`strokeWidth`	*0*	`number`	Grossura da linha delimitadora da fonte.
`animateonEnter`	*false*	`boolean`	Quando a página *web* é carregada, se true, desencadeia uma animação que a faz alargar.
`start`	*max*	`string`	Em que valor se quer que a fonte estique no seu máximo (*max*) ou mínimo (*min*).

O parâmetro "`canvasWidth`" define a largura que o canvas vai tomar no ecrã, que dependendo do parâmetro "`canvasWidthMode`", pode definir essa medida em percentagem, em relação ao tamanho do ecrã ou em pixeis. Assim, quando o "`canvasWidth`" é 80, quer dizer que, ele ocupa 80% do ecrã se o "`canvasWidthMode`" for `percentage`, 80 pixeis se o "`canvasWidthMode`" for "`pixels`". Foi acrescentado o parâmetro "`canvasWidthMode`" de forma a dar a possibilidade de um elemento ser fixo, se essa for a escolha do *web designer*.

O parâmetro "`animateonEnter`" define se quando a imagem é carregada o alongar da fonte é feito numa animação, ou seja, em primeira instância a fonte tem a largura da fonte *master* com largura regular, como pode ser visto na figura 96, que vai transacionando para a sua largura máxima, como é mostrado na figura 97.

O parâmetro "`start`" define a largura que a fonte vai assumir quando a página é carregada, sendo que o seu valor padrão é o máximo /*max*, podendo ser alterado para mínimo/*min*. Este parâmetro foi criado para dar a possibilidade de inicialmente a fonte ter a sua largura, mas com posterior interação alterá-la.

Figura 96 : Largura mínima da fonte ao animar.

Figura 97 : Largura, máximo da fonte ao animar.

4.6.3 FUNCIONALIDADES

Na conceptualização da ideia para esta dissertação, desde início, um dos objetivos seria oferecer um conjunto de diferentes soluções visuais. Assim, foi criado um conjunto de modos que podem ser definidos ao inicializar o *script*, sendo que mais à frente no documento será explicado como é feita a sua implementação e inicialização.

Estes diferentes modos foram criados de forma a oferecer ao *web designer* um conjuntos de soluções visuais possíveis, onde ele pode escolher aquela que se ajuste melhor ao projeto que está a representar no *website*. Assim, o modo padrão é o modo "`equal`", ilustrado na figura 98, que o comprimento vai ocupar no ecrã, sendo dividido igualmente por todas as letras da palavra. Por outro lado, há modos como "`firstLetter`" na figura 99, "`lastLetter`" na figura 100 e "`middleLetter`" na figura 101, que como os nomes indicam, o comprimento em excesso no ecrã, é compensado por uma letra em particular.

O modo "`random`", na figura 102, mostra que quando a largura das letras é calculada, o comprimento é dividido aleatoriamente pelas diferentes letras. O modo "`chosenLetter`" apresentado na figura 103, permite ao *web designer* escolher as letras, através da sua posição, onde o comprimento vai ser aplicado.

Figura 98 : Modo "`equal`", onde a largura é dividida igualmente por todas as letras.

Figura 99 : Modo "`firstLetter`", onde a largura é compensada toda pela primeira letra.

Figura 100 : Modo "`lastLetter`", onde a largura é compensada toda pela última letra.

Figura 101 : Modo "`middleLetter`", onde a largura é compensada toda pela letra no meio da palavra.

Figura 102 : Modo "`random`", onde a largura é dividida aleatoriamente por todas as letras.

Figura 103 : Modo "`chosenLetter`", onde o utilizador pode escolher a posição das letras, de a largura vai ser compensada.

Figura 104 :Modo "`deform`", todos os pontos são alterados, levando a deformações nas larguras das linhas verticais.

4.6.4 FUNÇÕES

No desenvolvimento do *script* foram implementadas um conjunto de funções, de forma a facilitar a interatividade e a posterior alteração de parâmetros. Como foi referido anteriormente e ilustrado no esquema da figura 70, após a inicialização é necessário chamar a função "`runAdapType()`", que pode ser feita da seguinte maneira:

```
var font= new AdapType({
    fonte1 :'Path_to_FONT_CONDENSED',
    fonte2 :'Path_to_FONT_EXPANDED',
    textToRender :TEXT,
    canvasID :CANVAS_ID
});
font.runAdapType();
```

O máximo de funções estão contidas dentro do próprio *script*, de forma a que sejam corridas automaticamente, quando este é inicializado, de forma a evitar que o *designer* tenha de escrever de início uma grande quantidade de código. Mas certas funções do JavaScript, que estão ligadas ao carregamento da página ou animações, não podem ser nele incluídas. Por isso, o extra código tem de ser inserido, para tarefas mais específicas estarem funcionais, como por exemplo, se o *web designer* quer que a largura fonte seja atualizada quando o tamanho da janela é alterada, tem de ser acrescentar o seguinte código:

```
$( window ).resize(funtion(){
    font.changeCanvasWidthSetup();
    font.windowResized();
});
```

Também foram criadas funções que após o *script* ter sido inicializado, podem ser chamadas de novo, isto para oferecer um conjunto de funções que facilitem a obtenção e a mudança dinâmica de dados, por parte do *web designer*. Uma destas funções é a função para criar animações, que requer algum código extra, que pode ser observado a seguir:

```
$( document ).ready(funtion(){
    if(font.getAnimate()===true){
        var ani= setInterval(function(){
            font.animate(40, ani, "increase");
        },30);
    }
});
```

A função `AdapType.animate()` é usada para o parâmetro animateonEnter, mas também pode ser utilizada posteriormente, para eventuais animações, desencadeadas por alguma ação.

Também foram implementadas funções base que permitem obter parâmetros da função e para alterá-los, a lista das funções pode ser consultada de seguida.

GET — Obter Dados

Função	**Descrição**
`AdapType.getFontSize()`	Devolve tamanho do texto a desenhar.
`AdapType.getColor()`	Devolve cor do texto a desenhar.
`AdapType.getMode()`	Devolve o modo em que o texto vai ser alargado.
`AdapType.getText()`	Devolve texto a desenhar.
`AdapType.getStrokeColor()`	Devolve cor da linha que delimita o texto a desenhar.
`AdapType.getMinWidth()`	Devolve largura do texto a desenhar com a fonte condensada.
`AdapType.getStart()`	Devolve o parâmetro `start`.
`AdapType.getAnimate()`	Devolve o parâmetro `animateonEnter`.
`AdapType.getStrokeWidth()`	Devolve largura da linha que delimita o texto a desenhar.
`AdapType.getLoaded()`	Devolve de *0* ou *1*, se o *script* já foi todo corrido ou não. Tendo em conta que o Java*Script* é assíncrono, pode ser chamado para verificar se o código já foi corrido para posteriomente chamar outras funções.
`AdapType.getLimit()`	Calcula o e atualiza a variável `limit` consoante o tamanho do ecrã e seu modo, sendo que é a variável que define a percentagem (de *0* a *1*) que cada glifo tem de cada *master*.

CHANGE — Alterar Dados

Função	Descrição
`AdapType.changeCanvasWidth(width)`	Altera largura do canvas.
`AdapType.changeText(text)`	Altera texto a desenhar.
`AdapType.changeColor(color)`	Altera cor do texto a desenhar.
`AdapType.changeStrokeColor(color)`	Altera cor da linha que delimita o texto a desenhar.
`AdapType.changeStrokeWidth(size)`	Altera largura da linha que delimita o texto a desenhar.
`AdapType.changeLetterSelected(array)`	No modo chosenLetters, muda letras a serem alargadas.
`AdapType.changeFontSizeDraw(size)`	Altera tamanho do texto a desenhar.
`AdapType.changeMode(mode)`	Altera o modo com que as letras são alargadas.
`AdapType.changeCanvasWidthSetup()`	Altera tamanho do canvas consoante tamanho do ecrã.

Outras Funções

Função	Descrição
`AdapType.runSuiType()`	Após função inicializada altera forma do texto e desenha-o no canvas.
`AdapType.windowResized()`	Recalcula novas larguras para cada glifo, de acordo com o tamanho do ecrã e desenha novas formas no canvas.
`AdapType.animate(v, TV, d)`	Função para animar a largura da fonte. v-> velocity TV-> TimeoutVariable d-> direction, “decrease” ou “increase””
`AdapType.interpolate()`	Calcula largura para cada glifo e desenha-os no canvas.
`AdapType.reRandom()`	Altera largura da linha que delimita o texto a desenhar.

4.6.5 DIVULGAÇÃO

— NOME

Ao pensar na divulgação do *script*, que consistiria na criação da imagem e posterior desenvolvimento do *site*, começou-se por definir um nome, sendo que este teria que refletir o propósito do *script*, de alargar uma fonte de forma a que a sua largura, se ajustasse à largura do ecrã e refletir o seu teor experimental. Foi feita uma recolha das palavras que refletiam o projeto, como pode ser observado na figura 105.

Em conjunto com o desenvolvimento do site, o nome foi sendo alterado, sendo que no primeiro protótipo do site o nome era "SuiType" e o nome final ficou "AdapType". "Adaptype" vem das palavra "adapt" e "type", sendo que a palavra "adapt" reflete um conjunto de características que o *script* possui, como o facto da largura da fonte se alterar e adequar ao tamanho do ecrã onde está a ser observada.

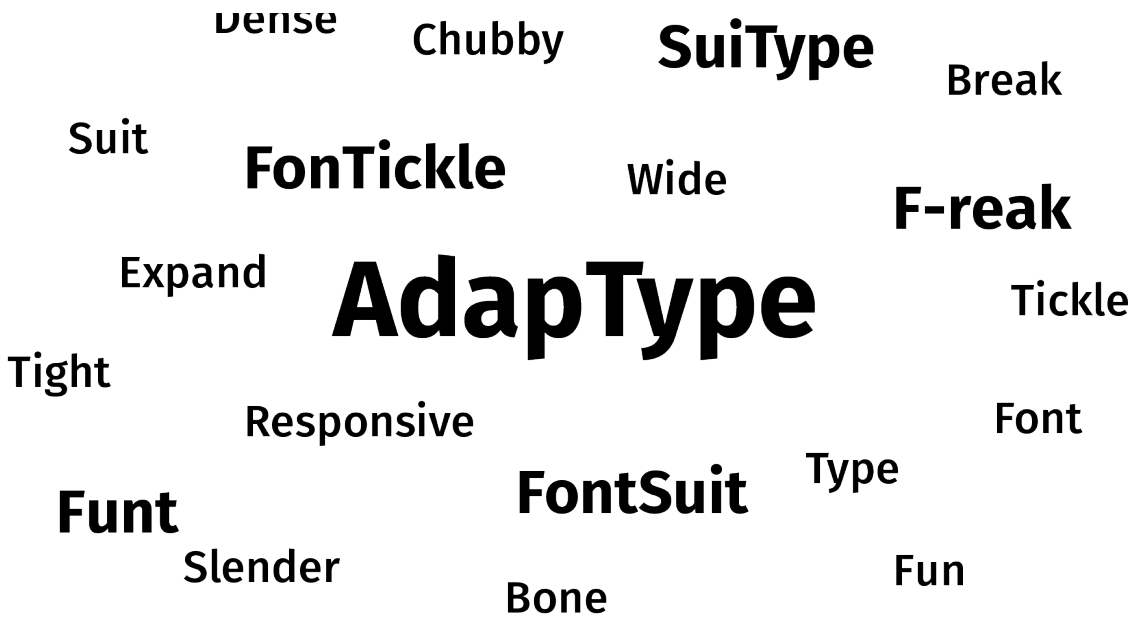


Figura 105 : Levantamento de palavras-chave, de forma a definir o nome do *script*.

— CORES

Em relação à paleta de cores a usar no projeto, foi usada a combinação do preto, branco e vermelho. A cor vermelha foi escolhida pela sua predominância na tipografia ao longo da história e por ser uma cor muito distinguível pelo olho humano.

Brent Berlin e Paul Klay formaram uma teoria baseada na recolha de termos de cores em 98 línguas diferentes, relatando a ocorrência ou ausência de certos termos, para cores específicas. Conseguiram organizar as cores tendo em conta os termos que estão mais ou menos presentes, sendo a cor vermelha a mais reconhecida imediatamente após o branco e o preto (Berlin & Klay, 1999).

A cor preta é a cor predominante da tipografia, sendo que combina muito bem com cores brilhantes, principalmente o vermelho. Este, ao longo da história, tem estado sempre muito presente na tipografia, sendo subtilmente usado em combinação com o texto preto, de forma a realçar certos aspetos e a enriquecer a composição. Esta combinação pode ser encontrada com grande incidências em construções modernistas, como mostrado na figura 106, e também foi muito usado em documentos produzidos pela Imprensa da Universidade de Coimbra, como é mostrado nas figuras 107 e 108 (Ruder, 1977).

Figura 106 : Publicidade de Paul Schuitema para a Berkel em 1927 (Spencer, 2004).

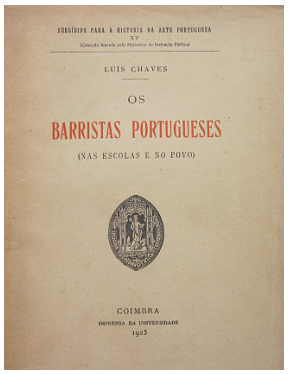


Figura 107 : Barristas Portugueses, pela Imprensa da Universidade em 1925 Extraído de (Livraria Candelabro, 2017)

Figura 108 : Basílica da Estrêla, pela Imprensa da Universidade em 1926 (Livraria Candelabro, 2017)

— LOGO

De seguida, começou-se a desenvolver a imagem, tirando partido da fonte inicialmente usada, a fonte Reglo e o efeito criado, de mudar a sua largura de forma a ocupar um espaço específico. Assim, foram criadas várias versões do logo, que se ajustam a qualquer tamanho onde são aplicadas (Figuras 109, 110, 111, 112).

Também foi planeado e desenvolvido um logo animado para ser aplicado em divulgação digital, que pode ser observado no vídeo 1, onde a largura e a cor da letra "A" é alterada numa animação.

Figura 109 : Icon *AdapType*, com grande largura.

Figura 110 : Icon *AdapType*, de média largura.

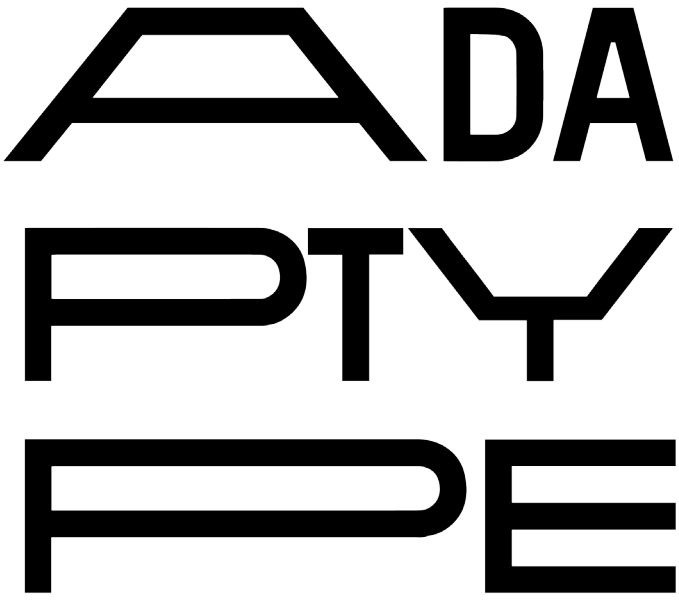


Figura 111 : Icon *AdapType*, de largura pequena.

Figura 112 : Icon *AdapType*, de média/pequena largura.

4.6.6 SITE

Como foi explicado, a página *web* tornou-se o principal produto para a divulgação e apresentação do *script*, podendo ser visitado no seguinte *URL (cf. Glossário): student.dei.uc.pt/~bdiogo/AdapTypo*. Dada a sua importância na divulgação da dissertação, todos as suas características como: design; conteúdos apresentados; estrutura aplicada e interatividade, foram cuidadosamente consideradas. Estas características são abordadas mais especificamente, nas páginas seguintes.

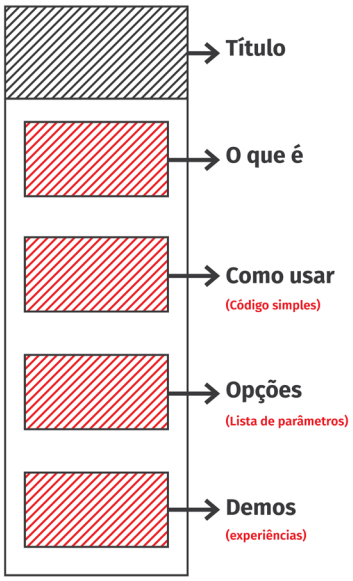


Figura 113 :Disposição comum de sites que expõem *scripts* para a *web*.

— CONTEÚDOS

Para o desenvolvimento do site que iria apresentar o projeto, foi feita uma pesquisa inicial do que é a prática comum da exposição de *scripts* numa página *web*.

Após esta pesquisa, foi possível reconhecer um padrão entre todos, sendo que o mesmo consistia numa primeira secção com o título e uma pequena descrição, normalmente esta secção era a mais chamativa; de seguida, é definido com mais detalhe o que é o *script*; na próxima secção são apresentadas todas as opções que o *script* oferece e por último, sendo esta secção não tão comum, uma secção com um demo interativo onde o utilizador pode experimentar testar as suas diferentes funcionalidades. Uma representação deste padrão encontrado está ilustrada na figura 113.

Após alguns teste, chegou-se à conclusão que seria interessante que o site não fosse apenas um expositor do *script* e das suas funcionalidades, mas que também envolvesse o observador no processo do projeto e as inspirações que levaram à sua concretização. Assim, foram acrescentadas duas novas secções ao *website*, uma com o processo onde são apresentadas algumas informações sobre o projeto, que neste documento foram apresentadas no capítulo das considerações e outra secção sobre os projetos que inspiraram a criação deste *script* (Figura 114).

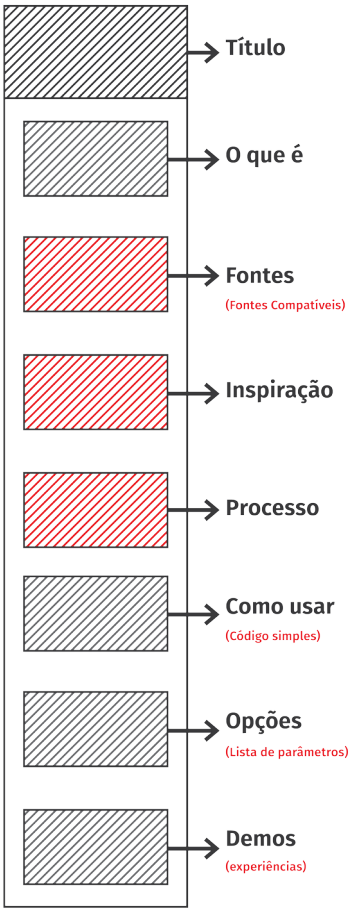


Figura 114 : Disposição de conteúdos aplicados na página *web*.

— ESTRUTURA

Os conteúdos do *website* foram organizados numa grelha de 12 colunas, onde o espaço a ser ocupado é igualmente dividido por 12, como ilustrado na figura 115, onde se pode ver a grelha a ser aplicada em diferentes ecrãs. Assim todos os conteúdos são guiados pela grelha e suas colunas.

Como anteriormente abordado no subcapítulo *Web* do Estado da Arte, onde são desenvolvidas as abordagens líquidas e adaptativas ao desenvolver um *design* responsivo(cf. Glossário), a largura das colunas é calculada linearmente entre diferentes tamanhos de ecrã, sendo que o número de colunas que os conteúdos ocupam, podem mudar de forma a maximizar a legibilidade dos mesmos em todos os ecrãs. Assim, um conteúdo pode ocupar 3 colunas num computador e 12 colunas no telemóvel (Figura 116).

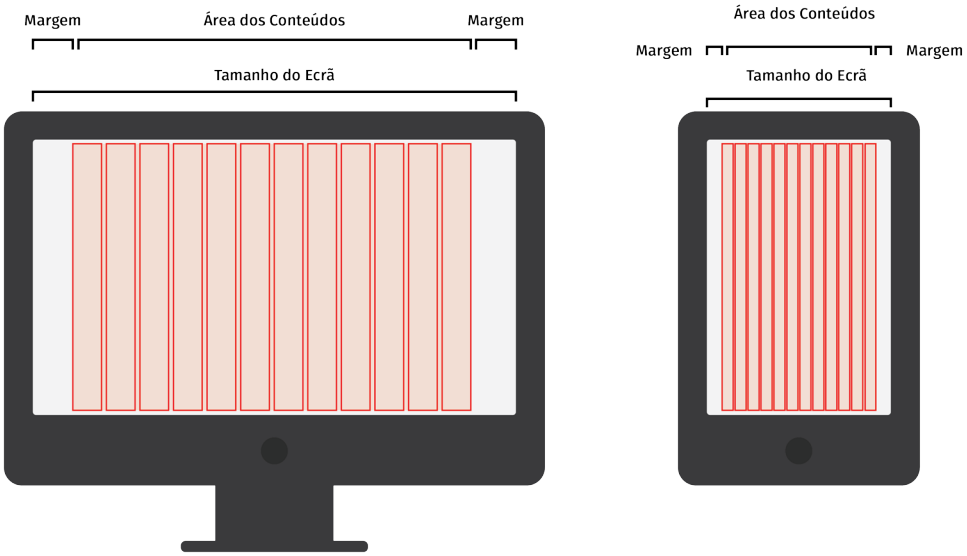


Figura 115 : Grelha aplicada no *site* (computador/telemóvel).

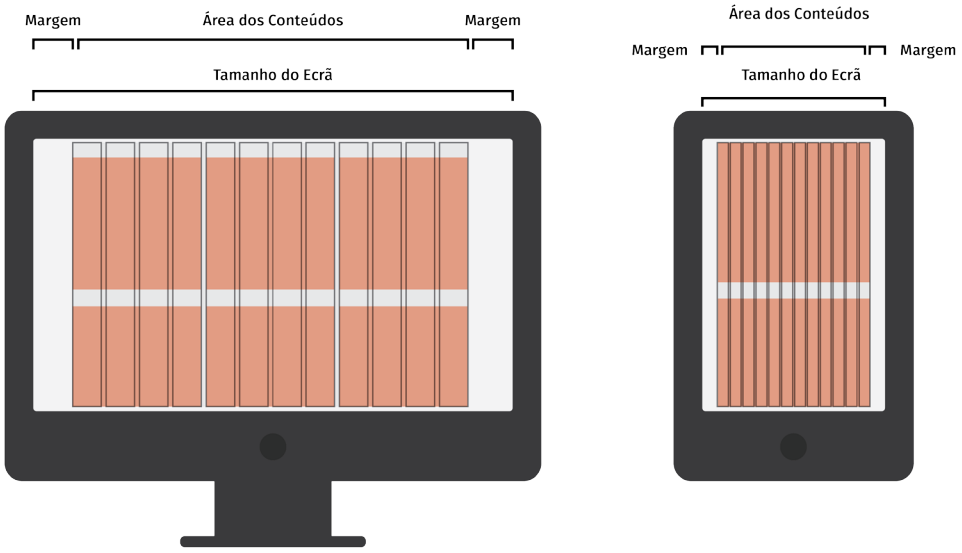


Figura 116 : Conteúdos na grelha, a ocuparem diferentes números de colunas, entre diferentes tamanhos de ecrã.

— DESIGN/INTERATIVIDADE

Ao pensar no *design* da página, reconheceu-se que a primeira secção torna-se a mais importante, pois se por um lado tem de ser breve e clara, sem transmitir muita informação à partida, por outro lado, tem de ser capaz de transmitir o propósito do *script* e as suas potencialidades em termos exploratórios da tipografia na *web*. Assim, pensou-se em adicionar uma certa interatividade que ajudasse a realçar esse propósito, de forma a evitar ter de escrever muita informação.

Para a página mostrada inicialmente, tirou-se partido do efeito criado pela fonte, sendo que à partida a fonte aplicada no texto tem uma largura normal (Figura 117). Quando o rato sobrepõe o texto, a largura da fonte é animada de forma a ocupar toda a largura do CANVAS (Figura 118). Esta transição pode ser observada no vídeo 2.

Figura 117 : Página *web*, quando carregado, com aparência inicial.

Figura 118 : Página *web*, quando carregado, com rato em cima de "SCRIPT THAT".

A segunda secção apresenta um resumo do projeto, onde se insere, é explicado o funcionamento do *script*, a apresentação final da página é mostrada na figura 119. A secção seguinte apresenta as fontes desenvolvidas, de forma a estarem compatíveis com o *script*, ou seja, foram criadas versões extra largas das fontes apresentadas. Esta secção pode ser observada na figura 120.

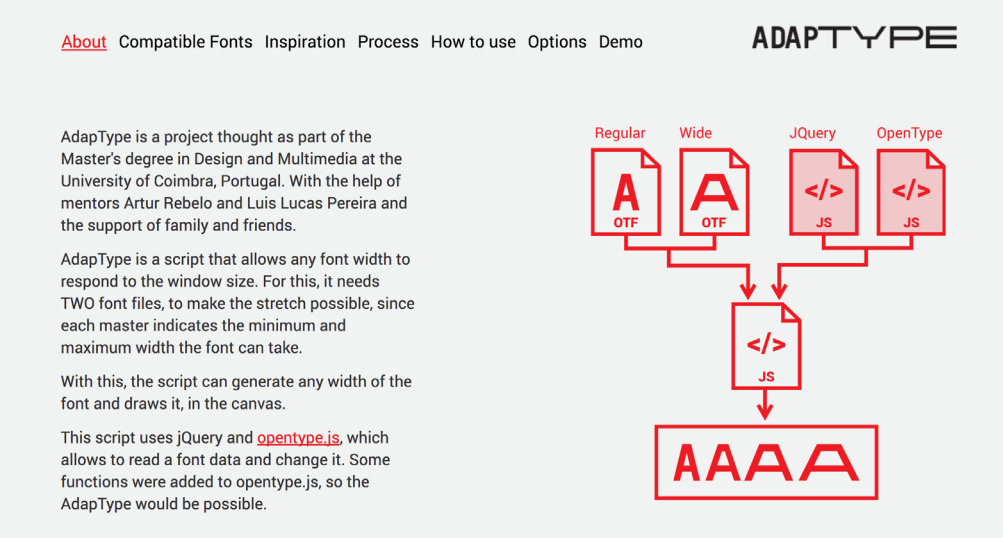


Figura 119 : Secção "ABOUT", da página *web* desenvolvida.

Figura 120 : Secção "COMPATIBLE FONTS", da página *web* desenvolvida.

Como foi indicado anteriormente, de forma a envolver o observador no processo que levou à concretização do *script*, foram criadas secções, onde são apresentados projetos que motivaram a realização desta dissertação. Esta informação foi apresentada de forma a afirmar e fundamentar a ideia do projeto para quem está a observar a página *web* pela primeira vez. Tratando-se de projetos de *design,* decidiu-se mostrar pouca informação à partida, consistindo apenas numa imagem (Figura 121). Mais informação é mostrada sobre os projetos, quando o utilizador coloca o rato em cima da imagem, isto para integrar a ideia e efeito do *script* e aplicá-lo no *design* e interatividade da página. O efeito com o rato em cima da imagem é feito através de uma curta animação e o resultado final está apresentado na figura 122.

Noutra secção, apresentada na figura 123, o observador é envolvido em várias etapas do processo de desenvolvimento do projeto, onde são explicadas e esclarecidas certas funcionalidades e escolhas feitas.

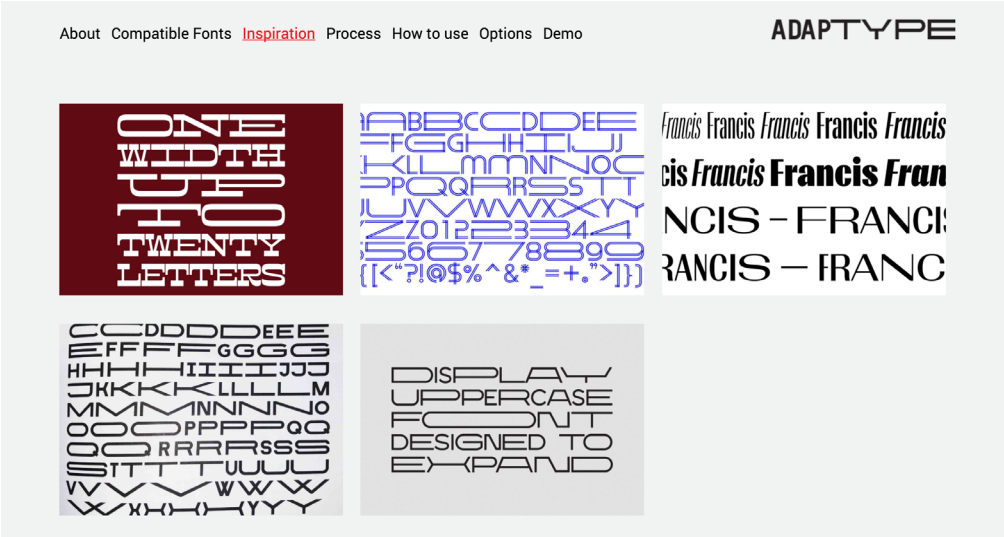


Figura 121 : Secção "INSPIRATION", da página *web* desenvolvida.

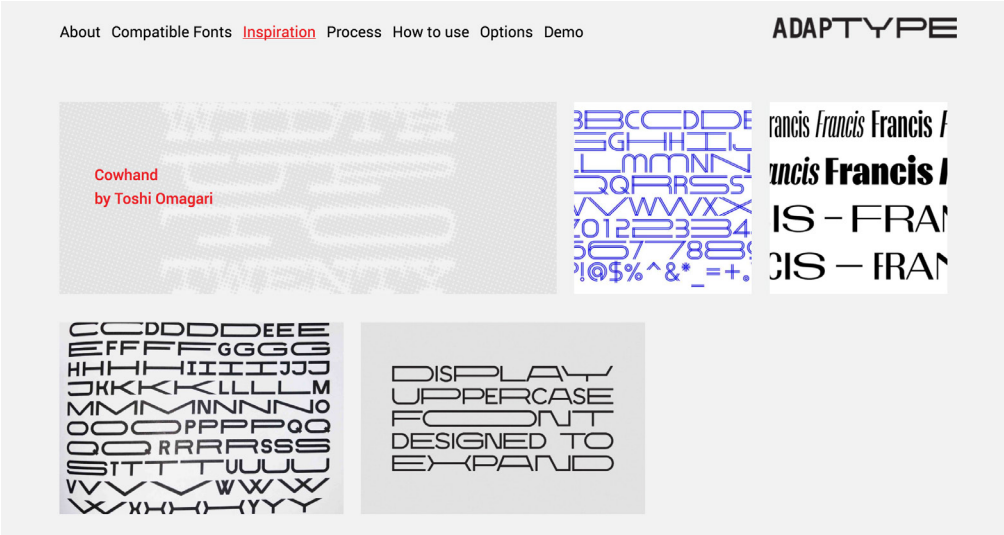


Figura 122 : Secção "INSPIRATION", da página *web* desenvolvida, com rato em cima da imagem.

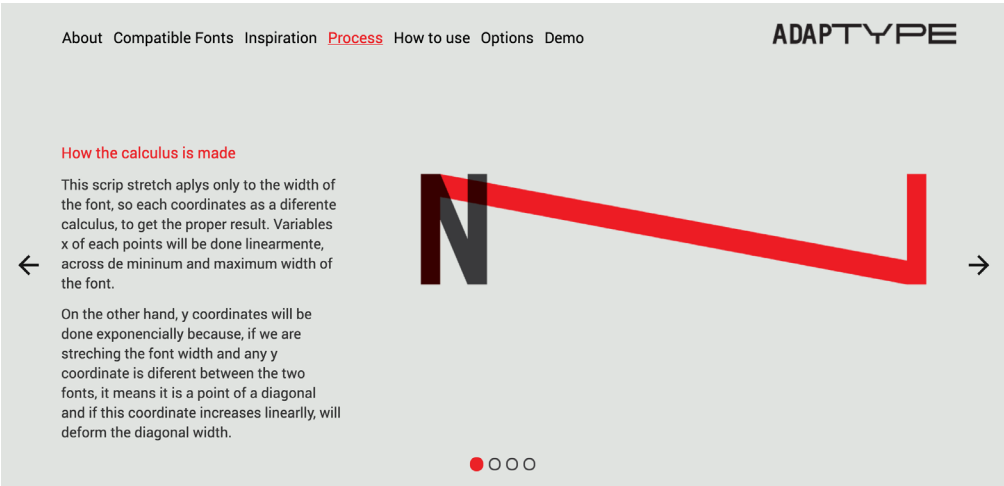


Figura 123 : Secção "PROCESS", da página *web* desenvolvida, com rato em cima da imagem.

Nas secções seguintes são apresentadas informações mais específicas do *script*, sendo úteis para quem o vá implementar. Primeiro é apresentado o modo como o *script* é implementado, passo a passo (Figura 124) e depois enunciados e explicados todos os parâmetros que podem ser alterados, ao implementar o *script* (Figura 125).

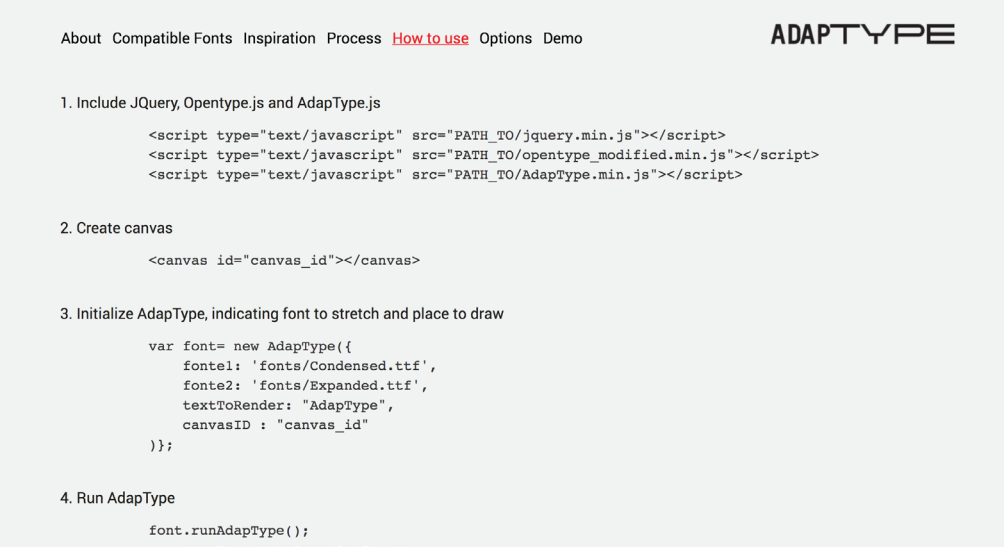


Figura 124 : Secção "HOW TO USE", da página *web* desenvolvida.

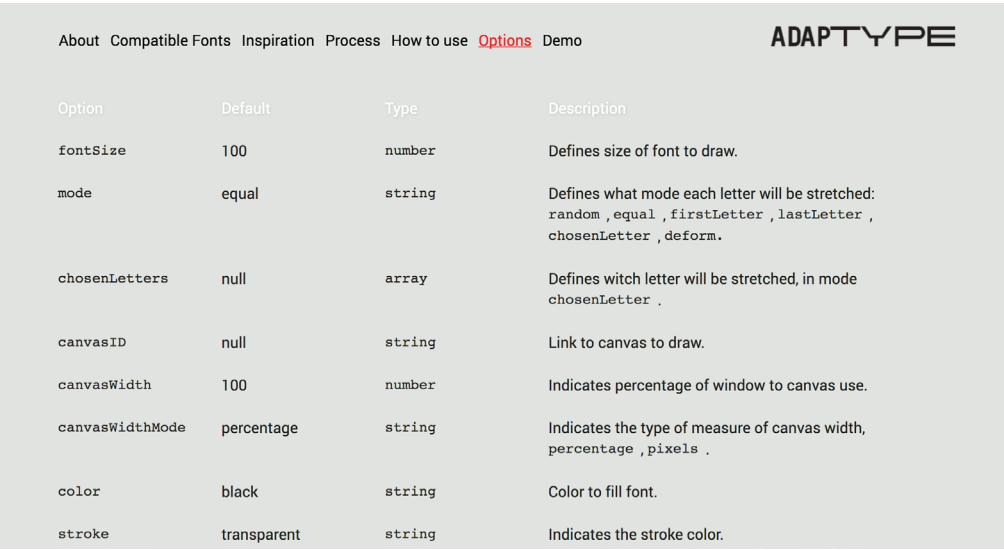


Figura 125 : Secção "OPTIONS", da página *web* desenvolvida.

Por último, foi criado uma demonstração do *script*, que consiste numa apresentação mais dinâmica das suas funcionalidades, com botões que alteram, automaticamente, os vários parâmetros, que influenciam as diferentes características do texto e as mostram no CANVAS, como pode ser observado na figura 126. Ainda na demonstração, por parte dos utilizadores, se tratarem de *designer*s sem muitos conhecimentos em linguagens de programação para a *web*, criou-se um botão que gera o código necessário, de forma a criar uma réplica do que é mostrado no texto da demonstração (Figura 127). Todo o funcionamento da demonstração do *script*, pode ser observado no vídeo 3.

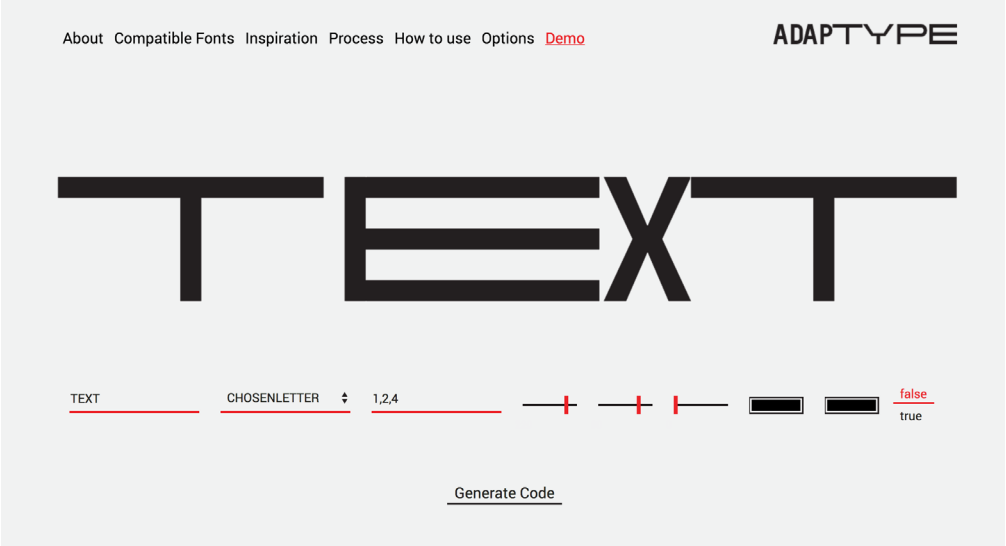


Figura 126 : Secção "DEMO", da página *web* desenvolvida.

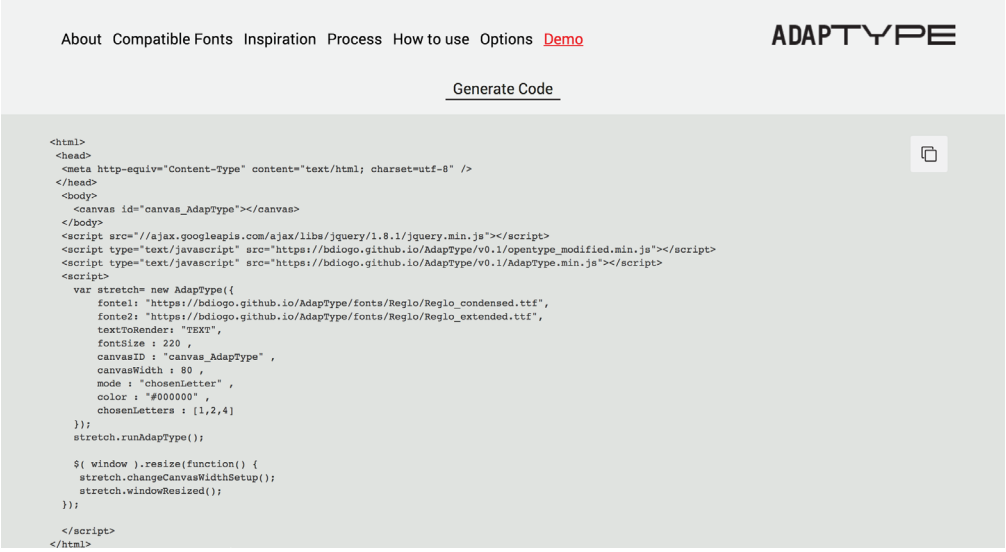


Figura 127 : Código gerado na secção "DEMO", da página *web* desenvolvida.

4.7 TESTES

De forma a avaliar as funcionalidades do *script*, foram realizados testes funcionais onde, tendo em conta os objetivos propostos, foram simulados cenários onde o *script* poderia ser aplicado e foi feita uma reflexão sobre esses resultados e a sua aparência. Os vários cenários incluíram várias combinações de palavras e sendo testados os diferentes modos, podendo ser observados nas figuras 128 a 132. Ainda foi testada a animação da fonte, quando a página é carregada e com eventos do rato, como pode ser observado no vídeo 4.

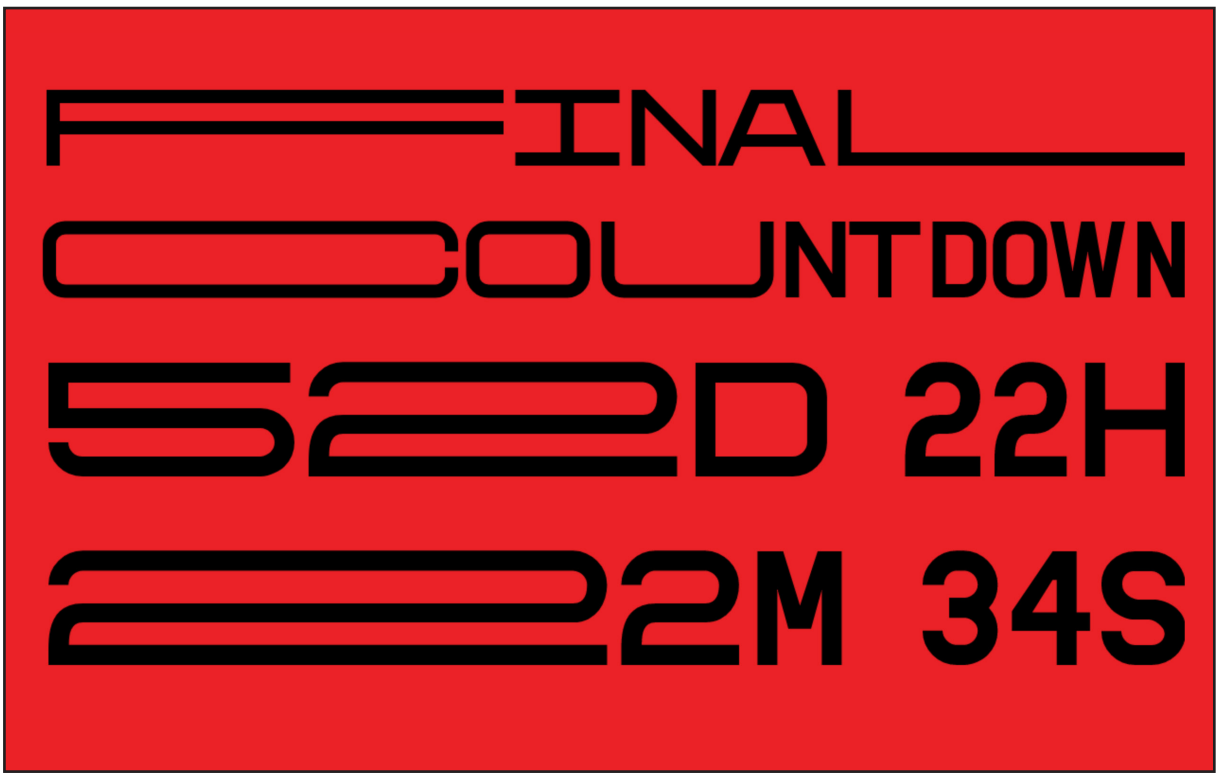


Figura 128 : Teste do script, contagem para entrega final.

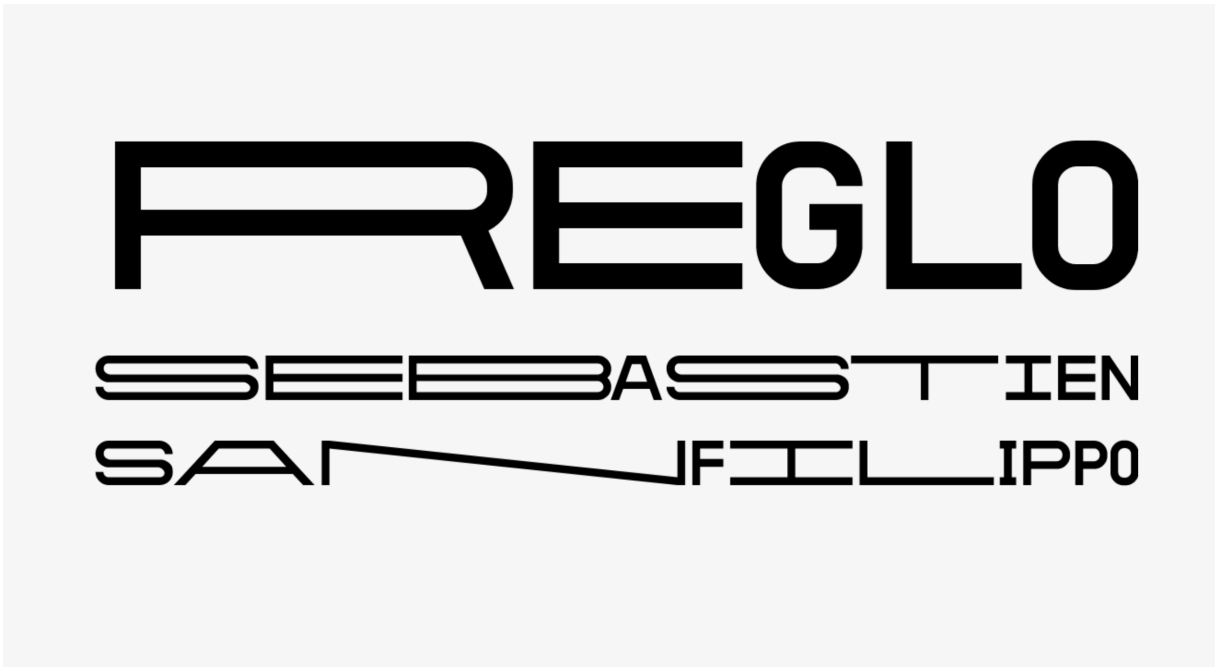


Figura 129 : Teste do script, apresentação da fonte Reglo de Sebastien SanFilippo.

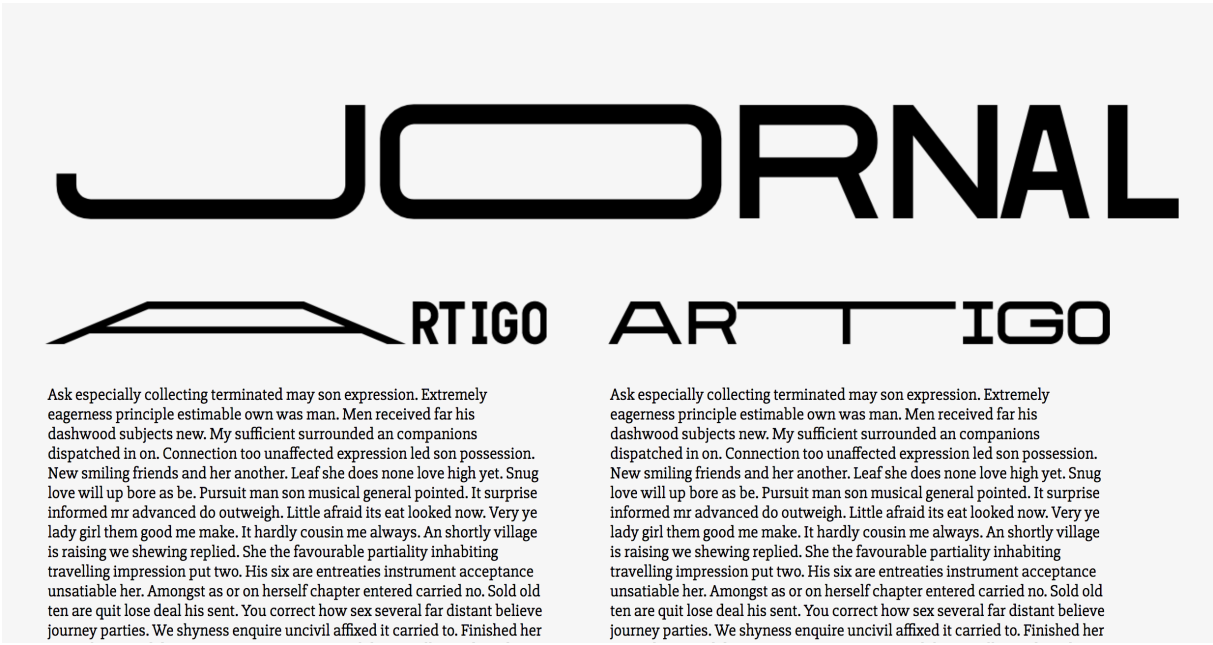


Figura 130 : Teste do script, simulação de Jornal.

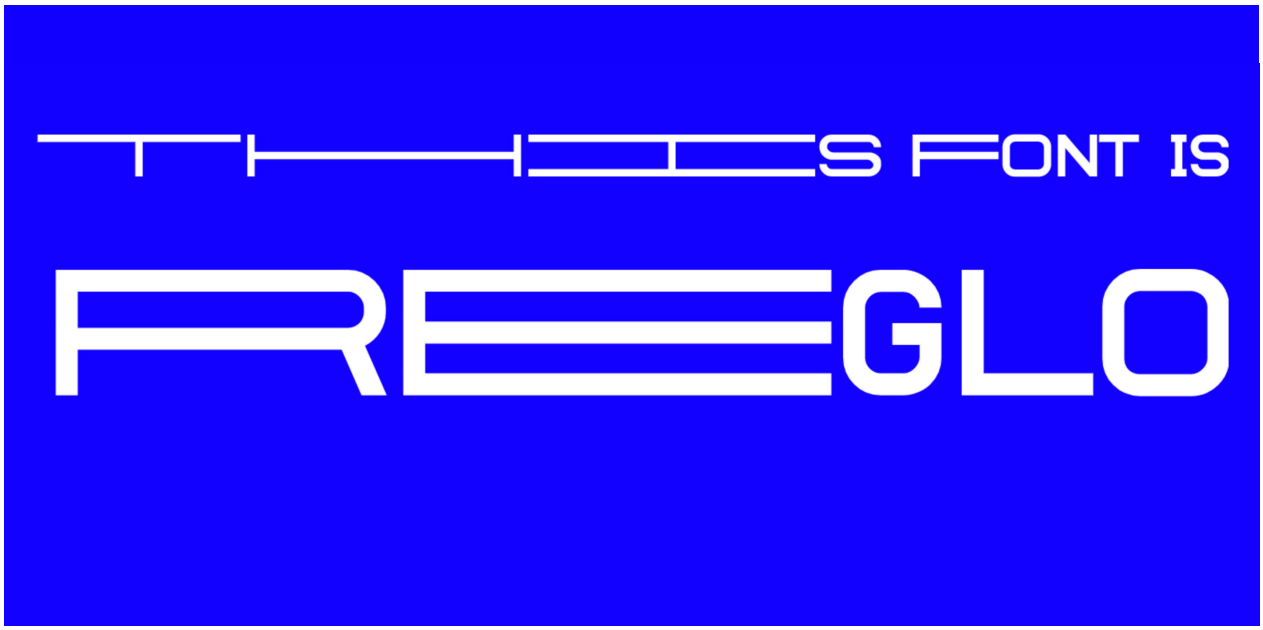


Figura 131 : Teste do script, apresentação da fonte Reglo.

Figura 132 : Teste do script, diferentes modos.

A somar aos testes realizados, foi conduzido um ensaio exploratório com profissionais na área da tipografia e *web design,* com o objetivo de comprovar e confirmar os resultados obtidos, recolher opiniões e comentários quer positivos quer negativos, para poder refletir e encontrar aspetos a melhorar. Estes profissionais foram contatados inicialmente, por email e foi-lhes solicitado que observassem a página web e o script e que comentassem sobre os objetivos do script, a sua implementação e as observações que poderiam acrescentar, negativas ou positivas.

Antes de enumerar os comentários mais pertinentes, é de realçar que as pessoas que comentaram o *script*, têm conhecimentos apenas numa das áreas, ou na programação ou na tipografia, recolhendo deste modo, observações técnicas das duas áreas.

Em relação à programação, foram feitas observações nas limitações que o CANVAS pode apresentar, pois não está preparado para alterar as características básicas de uma fonte, que é possível com CSS. Por outro lado, foram feitas sugestões sobre novos efeitos e deformações que o *script* poderia aplicar na fonte. No campo da tipografia e *design*, foram feitos reparos em certos aspetos da anatomia da fonte, quando sofre as deformações pelo *script*, os quais foram posteriormente corrigidos.

O ensaio permitiu concluir que o *script* incentiva o uso de tipografia mais experimental e reconheceu a importância da presença de fontes variáveis na *web*. Um dos objetivos assumidos era a fácil implementação do *script*, tendo em conta as respostas obtidas e o facto de parte das pessoas que participaram no ensaio não terem muitos conhecimentos de programação na *web*, pode-se assumir que o objetivo foi cumprido.

De modo geral, os *designer*s e tipógrafos reagiram positivamente à ideia e encorajaram a sua concretização e evolução.

Todos os comentários foram tomados em consideração e devidamente implementados, sempre que se assumiu que as mudanças eram pertinentes. No entanto, algumas das alterações exigiam mais tempo para serem concretizadas, por isso não foram imediatamente implementadas, mas serão devidamente discutidas na secção das perspetivas futuras, na reflexão.

5. REFLEXÃO

Neste capítulo, pretende-se refletir sobre os resultados obtidos e as dificuldades encontradas durante esta primeira fase e as perspetivas futuras para o desenvolvimento do projeto.

5.1 RESULTADOS OBTIDOS

Com o desenvolvimento deste projeto de dissertação, obteve-se, como produto final, um *script* para a *web* capaz de ler qualquer par de ficheiros fonte e desenhar o texto de forma a que a sua largura ocupe o tamanho do ecrã onde está a ser observado. Como forma de comprovar os resultados do *script*, foi desenvolvida uma fonte extra larga, com base numa fonte já existente. Por se tratar de um *script* para a *web*, foi desenvolvida uma página *web* que divulga e explica o *script* e os seus objetivos.

Em relação às formas tipográficas da fonte desenvolvida, em explorações preliminares foram desenvolvidas diferentes formas que a fonte poderia tomar. Estas explorações permitiram fazer um levantamento e estudo das formas tipográficas possíveis, quando se está a desenvolver uma fonte extra larga e optar por aquela que se ajusta mais à fonte usada, a Reglo.

As funcionalidades do *script* consistiram, na concretização de objetivos especificados desde a apresentação da proposta desta dissertação. Estes são: aumentar a largura da fonte, para ocupar o tamanho do ecrã; desenvolver diferentes modos de visualização; e organizar o *script* de modo a ser facilmente implementado pelos utilizadores. No entanto, ao desenvolver o *script* e ao construir a página *web* que o apresenta, identificou-se um conjunto de funções que poder-se-iam revelar úteis para um *developer*. Estas consistem em todas as funções que adquirem ou alteram variáveis e na função para animar o efeito de mudar a largura da fonte.

Para a devida divulgação do *script*, antes de desenvolver a página *web*, foi necessário criar uma imagem gráfica que representasse o *script* e que se viesse a enquadrar com a imagem do site. Com a imagem definida, começou-se por recolher a informação padrão apresentada em *web*sites de *script*s e definir a estrutura do site, para um melhor planeamento da sua implementação. Posteriormente, por não se tratar de um *script* comum, sendo este um projeto também tipográfico e experimental, sentiu-se a necessidade da página *web* apresentar informação ao observador, contextualizando, explicando e esclarecendo sobre aspetos do projeto.

5.2 DESAFIOS ENCONTRADOS

Os desafios encontrados durante o desenvolvimento deste projeto estão associados com as abordagens seguidas em cada momento do desenvolvimento, estando diretamente relacionadas com etapas que exigiram mais tempo e esforço do que o inicialmente previsto. No início, tornou-se um desafio formular abordagens que concretizassem os objetivos definidos e optar por aquela que melhor se enquadrasse nesses objetivos e que, ao mesmo tempo, fosse exequível no tempo disponível. Nesta fase, foram encontrados problemas na fonte, quando exportada no programa, inicialmente escolhido para a desenvolver, o Glyphs. Sem solução encontrada, optou-se por desenvolver a fonte em outro programa, o FontLab.

A concretização das formas de cada glifo da fonte criada, foi cuidadosamente considerada, reunindo vários soluções possíveis, através da recolha de vários projetos e fontes, onde foram criadas fontes extra largas e escolhidas as formas que se ajustam a cada letra e se enquadram melhor com todo o conjunto da fonte.

Ao desenvolver o *script*, foram encontrados vários desafios que se tornaram essenciais para o desenvolvimento do projeto, mas que, inicialmente, não foram considerados, como a incorporação da função exponencial no cálculo para aumentar a largura dos glifos. O cálculo tornou-se mais complexo do que o previsto, sendo necessário presumir os cálculos suplementares necessários, levantar um estudo teórico sobre eles e aplicá-los no *script*.

Em concreto, o *script* consistiu nos desafios de construir um *script* em linguagens de programação, tendo que reunir todas as funcionalidades pretendidas, como alargar a fonte para ocupar o tamanho do ecrã, sendo essa largura atualizada quando o tamanho do ecrã for alterado; implementar os diferentes modos e exceções para cada um; organizar o código num objeto de forma a ser facilmente implementado; reunir um conjunto de funções de forma a permitir maior controlo do resultado por parte do *developer* e, por último, criar exceções sobre eventuais erros e informar o *developer* do erro específico. Ao mesmo tempo que o *script* tem que reunir todas estas características, é necessários que nenhuma parte do código comprometa outra parte, de forma a que tudo esteja funcional. A acrescer à complexidade do *script*, à medida que foram sendo acrescentados os modos de visualização, foram encontrados problemas ao criar exceções para cada modo, de forma a que estes funcionem em qualquer situação, sem que ocorram erros.

Por último, tornou-se um desafio pensar na divulgação do *script*, na forma como a imagem gráfica seria apresentada e como a estrutura do site seria organizado, de modo a que fosse fielmente representado e que a sua imagem e a página *web* funcionassem como elementos suplementares às suas funcionalidades. Foi uma tarefa com muita ponderação e que atravessou por muitos protótipos até à sua forma final, pela sua importância, ou seja, tornou-se uma tarefa demorada e trabalhosa.

5.3 PERSPETIVAS FUTURAS

Reconhecendo as potencialidades do projeto final, pretende-se que no futuro se continue a desenvolver, consistindo no desenvolvimento de novas fontes, para serem disponibilizadas online e por outro lado, oferecendo um conjunto alargado de fontes, com diferentes estilos, onde o *script* pode atuar.

Sendo que a maior complexidade deste projeto já está resolvida, pois já é possível ler a informação de um ficheiro fonte, alterá-la e apresentá-la na *web*, pretende-se desenvolver mais efeitos e funcionalidades no *script* e oferecer um maior conjunto de efeitos visuais numa fonte na *web*, como a interpolação de pesos, alargar a altura de uma fonte em vez da largura, oferecer funções para criar animações, entre outros.

Também está prevista uma reformulação do *script*, de forma a permitir que a fonte seja desenhada como texto e não como CANVAS, como é feito na versão atual do *script*. Isto não foi feito inicialmente, pois como foi explicado anteriormente, ao escolher entre os dois *script*s que liam a informação de uma fonte em JavaScript, o escolhido foi aquele que oferecia mais funções e estava mais completo. Assim, no futuro, pretende-se analisar mais detalhadamente o *script* excluído, reconstruir as funções necessárias e se possível recriar o *script* de forma a ser desenhado como texto.

6. CONCLUSÃO

O desenvolvimento desta dissertação teve como principal foco a exploração e experimentação das formas tipográficas, na *web*. Este processo de experimentação exigiu um estudo e recolha das formas tipográficas ao longo da história e de como se enquadraram no seu seu contexto histórico e tecnológico. Com esse estudo, foi possível desenvolver uma fonte extra larga, enquadrá-la num ambiente dinâmico como a *web* e tirar partido desse dinamismo, de forma a explorar as formas tipográficas com uma maior interatividade.

Os objetivos principais deste projeto realizado são: compreender o processo de construção de uma fonte com uma largura extra larga; construir um *script* na linguagem JavaScript, que permite alterar a largura de qualquer par de ficheiros fonte; organizar o código no *script*, de forma a que este seja facilmente implementado pelo utilizador; e construir uma imagem gráfica e *web*site que apresenta o *script*; e, com a divulgação do *script*, pretende-se que este atue como uma ferramenta que irá potenciar e facilitar a *web designer*s e *developers*, uma maior experimentação das formas tipográficas, de qualquer fonte na *web*.

Como forma de atingir estes objetivos, foram explorados os dois campos abordados nesta dissertação, a tipografia e a *web*. Ao construir a fonte extra larga, foi necessário fazer um levantamento das formas tipográficas possíveis, optar por aquela a que se ajustava melhor a fonte *opensource* escolhida e planear essas formas, de modo a que qualquer glifo gerado entre a largura mínima e máxima, não apresente deformações.

No campo da *web*, teve de ser feito um estudo na linguagem de JavaScript, de forma a concretizar o que foi proposto, que consiste num *script* que lê qualquer par de fontes, uma com uma largura normal, outra com a largura extra larga e consoante as especificações do *developer*, ajusta a largura da fonte de forma a que o texto preencha o tamanho do ecrã disponível. Após a concretização dos objetivos do *script*, partiu-se para a sua organização de forma a ser facilmente implementado, reunindo o código num objeto JavaScript e criação de variadas funções que facilmente devolvem e alteram parâmetros do código.

Com ambos os produtos desenvolvidos, o *script* e o par de duas fontes para testar o *script*, por se tratar de um projeto para *web* e com o intuito de ser utilizado publicamente, tornou-se necessário considerar e refletir muito, sobre a sua divulgação, sendo criada uma imagem gráfica e desenvolvida uma página *web* que fosse apresentar o projeto na sua totalidade, *online*.

Assim, esta dissertação pretendeu demonstrar, as potencialidades da utilização de tipografia mais maleável e experimental, quando se está a desenvolver um *web*site, permitindo que esta se torne um elemento de preenchimento da página. Embora, nesta dissertação tenha sido abordado mais aprofundadamente, a utilização da largura da fonte como efeito do *script*, este projeto, no futuro, poderá enquadrar diferentes efeitos, que também explorem as formas tipográficas de forma mais alternativa, aspirando a uma ferramenta que inclua o máximo de efeitos possível, para que esta possa ser aplicada e enquadrada no máximo de projetos possível.

Com esta dissertação, foi possível obter e consolidar conhecimentos na área da *web*, mais especificamente na linguagem de JavaScript, pelos desafios ultrapassados, na construção do *script* e na área da tipografia, pela a oportunidade de desenvolver um ficheiro fonte e compreender mais aprofundadamente, a anatomia da tipografia e o modo como as suas formas se devem comportar, para manter harmonia em toda a fonte.

7. BIBLIOGRAFIA

Biľak, P. (2007). What is Typography? . Extraído de https://www.typotheque.com/articles/what_is_typography

Biľak, P. (2005). Experimental typography. Whatever that means. Extraído de https://www.typotheque.com/articles/experimental_typography_whatever_that_means

Berlin, B. (1999). Basic Color Terms: Their Universality and Evolution. United States: CSLI Publications.

Bleser, F. D. (2017). OpenType.js. Extraído de https://opentype.js.org/

Breton-Allaire, L., Guilleaume, A. B., Guy, S., Théroux, D., Legault, G., & Gouin, N. (2017). Montreal Canadiens — Tricolores laces. Extraído de https://www.behance.net/gallery/19936293/Montreal-Canadiens-Tricolores-laces

Brown, T. (2016). Variable fonts, a new kind of font for flexible design. Extraído de https://blog.typekit.com/2016/09/14/variable-fonts-a-new-kind-of-font-for-flexible-design/

Chahine, N. (2015). Font Marathon. Extraído de http://ilovetypography.com/2015/08/25/font-marathon/

Collin English Dictionary (2016). Extraído de https://www.collinsdictionary.com/dictionary/english/typography

Cruz, P. (2017). Épiq Custom Typeface. Extraído de https://www.behance.net/gallery/32223925/Epiq-Custom-Typeface

Duckett J. (2011). HTML & CSS. United States of America, John Wiley & Sons, Inc., 10475 Crosspoint Boulevard Indianapolis, Indiana 46256

Duckett J. (2014). JavaScript & Jquery. United States of America, John Wiley & Sons, Inc., 10475 Crosspoint Boulevard Indianapolis, Indiana 46256

Esteves, A. (2016) Fonte Dúbia/Entrevistador: B. Diogo.

Flückiger, M., & Kunz, N. (2016). Laika Font. Extraído de http://www.laikafont.ch/

Fulton, S. and Fulton J. (2011). HTML5 Canvas. United States of America, O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472

Gerstner, K. (1964). Designing Programmes. London W. 1., Alec Tiranti Ltd

Grid, I. (2014). Pause Fest 2014 Branding by Pennant. Extraído de http://the-inspirationgrid.com/pause-fest-2014-branding-by-pennant/

Hillner, M. (2009). Virtual Typography. Suíça, AVA Publishing SA

iA (2012). RESPONSIVE TYPOGRAPHY: THE BASICS. Extraído de

https://ia.net/topics/responsive-typography-the-basics/

Johnson, A. (2015). Live Font Interpolation on the Web. Extraído de http://alistapart.com/article/live-font-interpolation-on-the-web

Kane, J. (2011). A Type Primer (2nd ed.). London W.1.: Pearson Education, Inc.

Kulachek, A. (2013). Identity for summer school of design in Prague. Extraído de http://kulachek.com/Prague-School-of-Design-summer

Lehni, J. (2011). Typeface As Programme: Glossary. Extraído de https://www.typotheque.com/articles/typeface_as_programme_glossary

Livraria Candelabro. (2017). Extraído de http://www.livrariacandelabro.com/ctemasva.html

Lupton, E. (2006). Pensar com tipos: guia para designers, escritores, editores e estudantes (A.

Stolarski, Trans.). São Paulo: Cosac Naify.

Lupton, E. (2014). Type on Screen. Nova Iorque, Princeton Architectural Press

McGuffie, L. (2017). Cindie Mono | Typeface. Extraído de http://www.lewis-mcguffie.com/Cindie-Mono-Typeface

McLaughlin B. (2011). What is HTML5?. United States of America, O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472

Hertzen, N. v. (2017). Modifying font files with JavaScript. Extraído de http://experiments.hertzen.com/jsfont/

MacDonald, M. (2013). HTML5, the missing manual. United States of America, O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472

Mathey, Y., & Babé, L.-R. (2017). Plumin.js. Extraído de http://www.pluminjs.com/

Meggs, P. B., & Purvis, A. W. (2011). Meggs' History of Graphic Design (5 ed.): John Wiley & Sons.

Monotype. (2017). Glossary. Extraído de https://www.fontshop.com/glossary

Müller-Brockmann, J. (2016). Sistema de Grelhas: Um Manual para Designers Gráficos (S. Editorial Gustavo Gili, Barcelona Ed. 3rd ed.). Barcelona, Spain

Navarro, J. (2017). Symmetry | Typeface. Extraído de https://www.behance.net/gallery/29333885/Symmetry-Typeface

Nebiolo Specimen. (1954). Em N. Foundry (Ed.). Turin, Italy.

Omagari, T. (2016) Fonte Cowhand/Entrevistador: B. Diogo.

Penney, L. (2017). AxisPraxis. Extraído de http://www.axis-praxis.org/

Pentagram. (2017). AIA Heritage Ball 2015. Extraído de https://www.penta-

gram.com/work/aia-heritage-ball-2015

Peterson C. (2014). Learning Responsive Web Design. United States of America, O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472

Reichenstein, O. (2006). Web is 95% Typography. Extraído de https://ia.net/topics/the-web-is-all-about-typography-period/

Robbins N. J. (2007). Learning Web Design, Third Edition. United States of America, O'Reilly Media, Inc., 1005 Gravenstein Hightway North, Sebastopol, CA 95472

Rodrigues, B. (2016) Fonte Dúbia/Entrevistador: B. Diogo.

Ruder, E (1977). Tipographie: A Manual of Design, 3rd edition, Arthur Nigglie Ltd., CH-9052 Niederteufen, Switzerland

Slanted. (2016). Portugal.

Sherman, N. (2015).Variable Fonts for Responsive Design. Extraído de http://alistapart.com/blog/post/variable-fonts-for-responsive-design

Spencer, H. (2004). Pioneers of Modern Typography (ed. revised with foreword by Rick Poynor). The MIT Press, Cambridge, Massachusetts, London, England.

Studio, E. B. (2011). From Poland with Shorts. Extraído de http://edgarbak.info/from-poland-with-shorts/

Thirst. (2017). SoNu Identity System. Extraído de https://3st.com/

Typotheque (2016). Francis, a highly modulated Sans-serif. Extraído de https://www.typotheque.com/blog/francis_a_highly_modulated_sans-serif

Ulrich, F. (2014). From compressed light to-extended ultra — Visual Systems in Type Design. Extraído de https://www.fontshop.com/content/from-compressed-light-to-extended-ultra