SIIMI/2016
IV simpósio internacional de inovação em mídias interativas
IV international symposium on innovation in interactive media
MAIO 4•6 UFG/BR
ISSN 2358-0488

Design Gráfico Computacional — DrawBot como ferramenta de produção visual.

Marcilon Almeida de Melo (UFG)

Palavras-chave: drawbot, arte generativa, programação, design computacional

Resumo

Este artigo analisa a influência do pensamento computacional aplicado ao design gráfico. Investiga de que maneira a programação pode ser utilizada no processo criativo e produção de artefatos visuais dinâmicos ou estáticos. Aprofunda a discussão ao verificar de que forma designers gráficos podem utilizar o *software* Drawbot para aprenderem conceitos fundamentais do pensamento computacional e de que forma esses fundamentos podem ser aplicados na produção de artefatos de design gráfico.

Abstract

This article analyzes the influence of computational thinking applied to graphic design. Investigates how the program can be used in the creative process and production of dynamic or static visual artifacts. Deepens the discussion to check how graphic designers can use Drawbot software to learn basic concepts of computational thinking and how these fundamentals can be applied in the production of graphic design artifacts.

Introdução

O design gráfico, suas práticas e métodos, sempre refletiram as características das mídias as quais seus artefatos visuais foram destinados. A cultura do objeto impresso, apesar da longa tradição caligráfica, acelerou-se de maneira sem precedentes a partir da popularização do tipo móvel e de toda a cadeia de produção proposta por Johannes Gutenberg, isso teve impacto profundo na aparência da página impressa e no desenho de tipos (elementos fundamentais do design gráfico). A partir dos anos 80 os computadores digitais e os softwares de produção visual permitiram que designers aplicassem toda a bagagem cultural e histórica da mídia impressa em softwares que simulavam a página impressa. Como também experimentar as novas possibilidades introduzidas por uma nova mídia, que diferente da mídia impressa, era a fusão de todas as outras mídias em uma e foi chamada por Alan Kay e Adele Goldberg (1977) de *metamídia*. Para entender e pensar criativamente nesse novo meio de comunicação os designers não podiam simplesmente utilizar o computador como um assistente de produção inferior, era necessário percebê-lo como um novo meio, uma nova linguagem com novos desafios.

Algumas décadas depois o computador já não é visto como novidade, mas sim como parte *sine qua non* da comunicação humana moderna. A não ser pelo que

chamo de uma espécie de fetiche pelo analógico — pinturas de parede, letreiros, desenho, ilustrações, etc. — , atualmente fazer design gráfico sem a mediação de *softwares* é virtualmente impossível, ou no mínimo pouco prático. Somo "reféns" do *"What You See Is What You Get (WYSIWYG)"*. O que vemos na tela é um antevisão do produto final em sua forma acabada, já não pensamos no código que está na base de tudo que produzimos e com isso talvez estejamos perdendo parte da autonomia e libertação que o computador potencialmente tem a nos oferecer. Apesar dessa aparente contradição o designer gráfico tem despertado interesse cada vez maior para uma habilidade que nunca consideramos essêncial para o fazer profissional: a programação. O designer programador vê além do permitido pelo software padrão de mercado. Ele automatiza, integra, anima, cria camadas de interatividade e visualiza informações.

1. Pensamento Computacional e Design Gráfico

Programar ainda é visto por muitos como uma espécie de camisa de força que impede a "real força expressiva" ao impor regras, códigos, procedimentos e lógica matemática na resolução de problemas. O filósofo Vilém Flússer investigando a natureza dos códigos e a capacidade humana de interpreta-los, referindo especificamente aos códigos de natureza tecnológica disse que "devemos aprendê-lo, senão seremos condenados a prolongar uma existencia sem sentido em um mundo que se tornou codificado pela imaginação tecnológica" (FLUSSER, 2007, p.137). Flusser nos lembra da importância de se compreender os novos códigos para que possamos dar sentido ao mundo que nos cerca.

Apesar de Flusser não referir especificamente aos códigos de natureza digital, aqueles formando com cadeias binárias representadas por 0's e 1's, podemos inferir que a compreensão desse tipo de código é tão importante quanto a compreensão da escrita alfabética. Compreender os códigos digitais significa elaborar pensamentos voltados a resolucão de um determinado problema em termos computacionais. Essa estratégia é chamada de pensamento computacional, *computational thinking* (CT), definido por (WING apud SANT'ANNA, 2012) e (GOOGLE, 2016) como um conjunto de competências e habilidades tradicionalmente restritas aos profissionais da Ciência da Computação, mas que poderiam ser utilizadas na resolução de problemas de qualquer área do conhecimento, incluindo matemática, ciência, artes e humanidades. Para o pioneiro cientistas e artista Alan Kay (1989) o desenvolvimento da capacidade de pensar em termos de mídia está relacionado com a capacidade de saber "ler" e "escrever", onde ler em uma determinada mídia significa acessar materiais e conteúdos criados por outros e escrever em uma mídia é capacidade de gerar materiais e ferramentas para outros. Tratando da diferença da mídia imprensa e da digital ele diz "na escrita impressa, as ferramentas são retóricas; elas demonstram e convencem. Na escrita computacional, as ferramentas que você gera são processos; eles simulam e decidem" (KAY,1989,p.125). Apesar do CT ser mais associado ao

desenvolvimento de aplicações computacionais, sua aplicação em áreas como matemática, humanidades, artes e design gráfico é vasta e vem ganhando espaço como estratégia de produção.

Em 1964, antes do alvorecer do uso do computador como ferramenta criativa para designers gráficos, o designer suiço Karl Gerstner em seu famoso livro "Designing Programmes" delineou o papel do designer com um sujeito capaz de analisar um problema de comunicação e propor um conjunto de soluções programáticas, um sistema, como método de trabalho e pensamento (muito próximo das competências do CT). Dizia ele:

> Descrever o problema é parte da solução. Isso implica: não tomar decisões criativas induzido pelo sentimento, e sim por critérios intelectuais. Quanto mais exatos e completos tais critérios, mais criativo torna-se o trabalho. O processo criativo deve reduzir-se a um ato de seleção. Projetar significa: escolher determinados elementos e combiná-los. Visto dessa perspectiva, o projeto requer um método.(GERSTNER, 1964, p.897)

Janet Murray (2012) diz que fazer design é dar forma a um artefato ou processo especifico ao escolher qual estratégia será utilizada para alcançar determinado objetivo. Dessa forma o designer deve ser capaz de considerar múltiplas abordagens para o problema de design, incluindo novas estratégias para explorar as *affordances* de novos materiais. O designer gráfico utiliza diversos elementos e técnicas para propor soluções para problemas de comunicação visual. Compreender e dominar princípios fundamentais do design tais como: tipografia, grid, cor, composição, hierarquia, etc., é parte essêncial na formação técnica e intelectual do designer gráfico, mas talvez esse conjunto pode ser expandido ao considerar a programação como capacidade fundamental para o desenvolvimento pleno de sua atividade.

2. DrawBot e seu papel na educação

O projeto DrawBot[1] teve inicio em 2003 por ocasião de um workshop de Python[2] na conferência tipográfica TypoTechnica. Criado por Just Van Rossum, Erik van Blokland e Frederik Berlaen, naquele momento o software que se chamava "DesignRobots", posteriormente foi renomeado para DrawBot, sendo atualmente um projeto open-source em ativo desenvolvimento. O DrawBot é uma aplicação poderosa e gratuita para MacOSX que permite o uso de Python para escrever simples scripts para gerar gráficos bidimensionais. O programa trás embutido suporte a primitivos básicos como retângulos, ovais, bezier, polígonos, texto,

[1] http://www.drawbot.com

[2] Python é uma linguagem de programacão criada em 1991 por Guido Van Rossum. É uma linguagem de programação de alto nível, que significa dizer que o seu código escrito é mais próximo de uma linguagem alfabética comum (no caso o inglês), do que de da linguagem de máquina (formada por zeros e uns). www.python.org

cor, transparência, além de uma poderosa API para exportar imagens em diversos formatos. Segundo descrição no website do programa:

DrawBot is an ideal tool to teach the basics of programming. Students get colourful graphic treats while getting familiar with variables, conditional statements, functions and what have you. Results can be saved in a selection of different file formats, including as high resolution, scalable PDF, sag, movie, png, jpeg, tiff…DrawBot has proven itself as part of the curriculum at selected courses at the Royal Academy in The Hague.(DRAWBOT, 2016)

Com uma interface e uma linguagem de programação relativamente simples, o DrawBot pode ser a porta de entrada ideal para o designer gráfico que deseja aprender como integrar conceitos fundamentais da programação na resolução de problemas. Uma das limitações do DrawBot, pelo menos até o momento, é a impossibilidade de utilizar 3d, áudio e vídeo. Para isso existem aplicações mais robustas como Processing[3], que é parte de um crescente número de ferramentas de programacão a disposição do designer gráfico. Apesar do DrawBot ser voltado essêncialmente para gráficos 2D, dimensão básica do design gráfico, atualmente ele é utilizado por muitos educadores como ferramenta introdutória de conceitos fundamentais de programação.

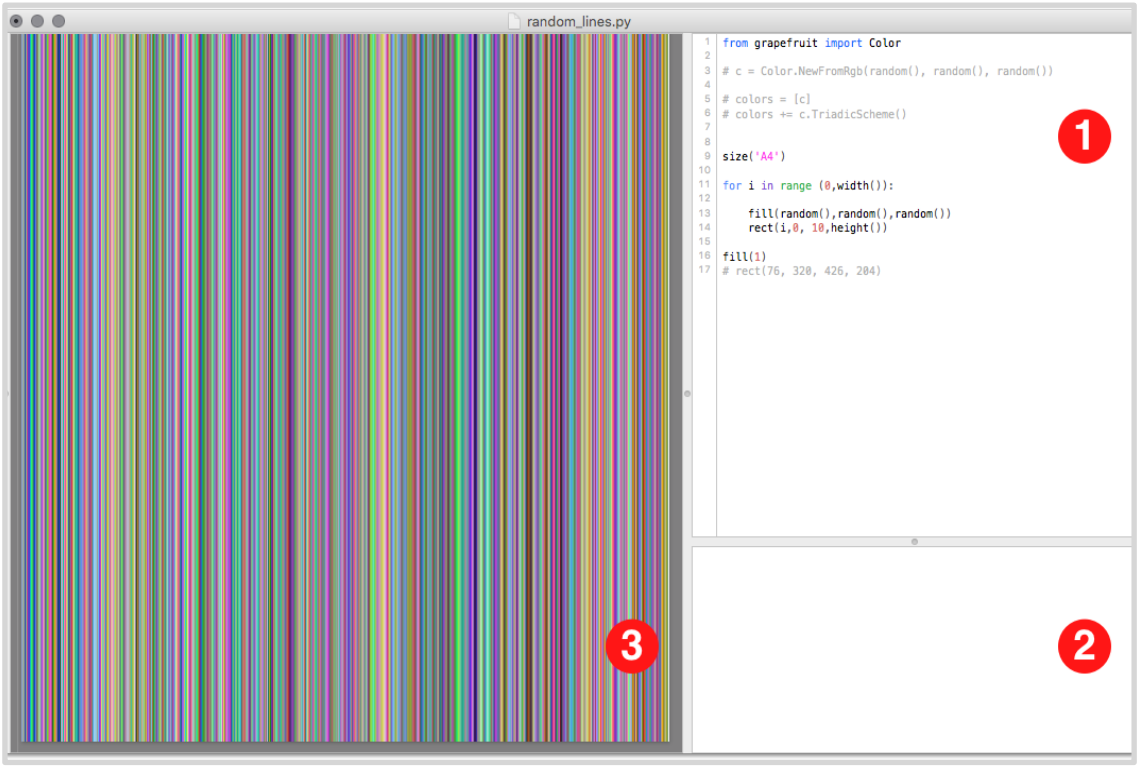


Fig.01 - Componentes básica da Interface do DrawBot: 1- Área onde o código é escrito; 2-Aréa onde o código é avaliado e 3- Área de resultado visual.

O DrawBot é utilizado como ferramenta pedagógica em alguns cursos na *Royal Art Academy in The Hague — Koninklijke Academie van Beeldende Kunsten*

[3] http://www.processing.org

ISSN 2358-0488

(KABK) in Den Haag — (Holanda). Depois de apresentados conceitos fundamentais da programação utilizando o DrawBot os alunos são encorajados a experimentar variações e combinações dinâmicas em seus projetos. A aceitação tem sido bastante positiva ao ponto que egressos do curso se tornam difusores do software, e mais importante, advogam por uma mudança na maneira como designers podem ampliar seus horizontes de possibilidades com a utilização de programação. Dentre os difusores cabe destacar Gustavo Ferreira, designer gráfico, programador, designer de tipos e educador, um entusiasta e dedicado promotor do Python e DrawBot através de workshops[4] voltados especificamente para designers.

3. Programação aplicada ao design gráfico

Em julho de 2015, na cidade de São Paulo, ocorreu o workshop "Design Tipográfico Computacional (dtgc)", voltado para designers com pouca ou nenhuma experiência em programação. O curso foi ministrado pelos instrutores Gustavo Ferreira e Eduardo Omine[5], ambos designers gráficos e progamadores.

Participei do workshop com o objetivo de aprender como integrar técnicas de programação no meu processo criativo. O principal objetivo do workshop era o de aplicar o conteúdo do curso — lógica, operadores, tipos de dados, funções e sintaxe básica da linguagem Python — a um projeto prático de design gráfico ou de desenho de tipos. Escolhi um projeto de identidade visual cuja as características eram adequadas para serem automatizadas via programação, o qual detalho a seguir.

3.1 MODUL — Identidade visual interativa

Modul é o nome de um espaço de co-working localizado na cidade de Goiânia. O projeto de identidade visual foi desenvolvido no final de 2014 tendo como proposta final um sistema modular tipográfico capaz de ser adequado para as mais diversas necessidades de uso. A marca é um tipograma formado pelos glifos "M","O", "D","U" e "L", onde o "M" é o único glifo fixo, enquanto as outras letras podem ser expandidas no sentido horizontal, vertical ou uma combinação de ambos.

[4] Design Tipográfico Computacional (dtgc) é um workshop ministrado por Gustavo Ferreira e Eduardo Omine voltado especificamente para designers. Eles utilizam DrawBot com ferramenta principal. http://dtgc-workshop.com.br/

[5] Eduardo Omine é Mestre em Design pela USP - https://br.linkedin.com/in/eomine

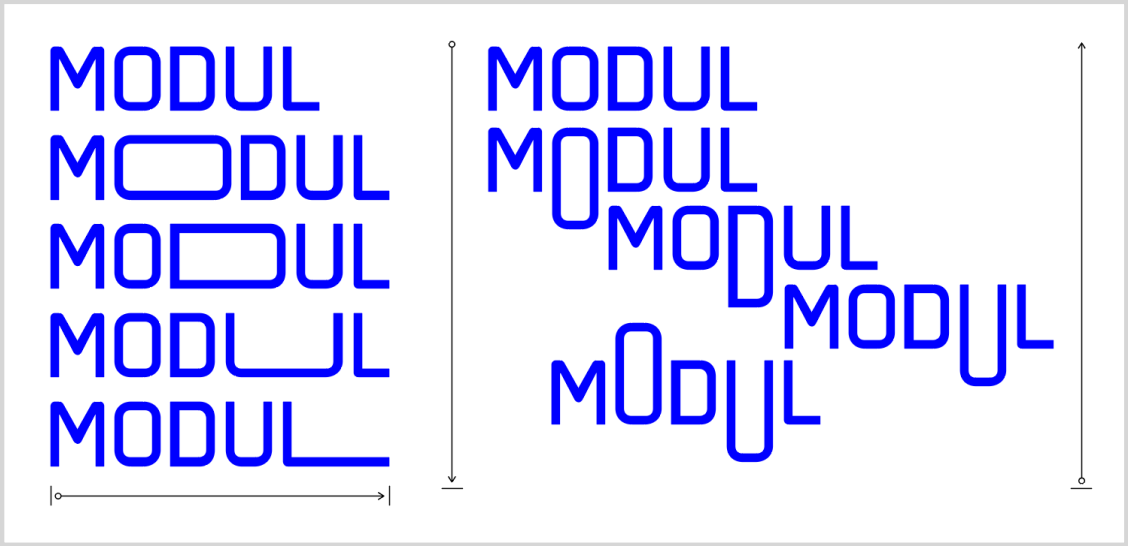


Fig.02 - Logotipo "MODUL" com variações no sentido horizontal e vertical.

O primeiro desafio foi transferir as coordenadas dos contorno do Adobe®Illustrator CC, aplicação onde o projeto foi feito originalmente, para o sistema de coordenadas utilizado pelo DrawBot. O DrawBot tem como ponto de origem "x" e "y" na parte inferior esquerda do *canvas*, enquanto o Illustrator utiliza a parte superior esquerda como ponto de origem das coordenadas. Para acelerar o desenvolvimento, todos o glifos foram copiados do Illustrator e colados no Robofont[6] pelo fato deste utilizar o mesmo sistema de coordenadas do DrawBot. Para então extrair as coordenadas utilizei o seguinte código:

```
from robofab.pens.pointPen import *
g = CurrentGlyph()
pen = PrintingSegmentPen()
g.draw(pen)
```

[6] Editor de fontes baseado em Python. www.robofont.com

ISSN 2358-0488 | ISBN 978-85-495-0020-5 | ROCHA, Cleomar (Org). Anais do IV Simpósio Internacional de Inovação em Mídias Interativas. Goiânia: Media Lab / UFG, 2016.

Tomando por exemplo o glifo "M", ao executar o código acima o seguinte resultado é obtido:

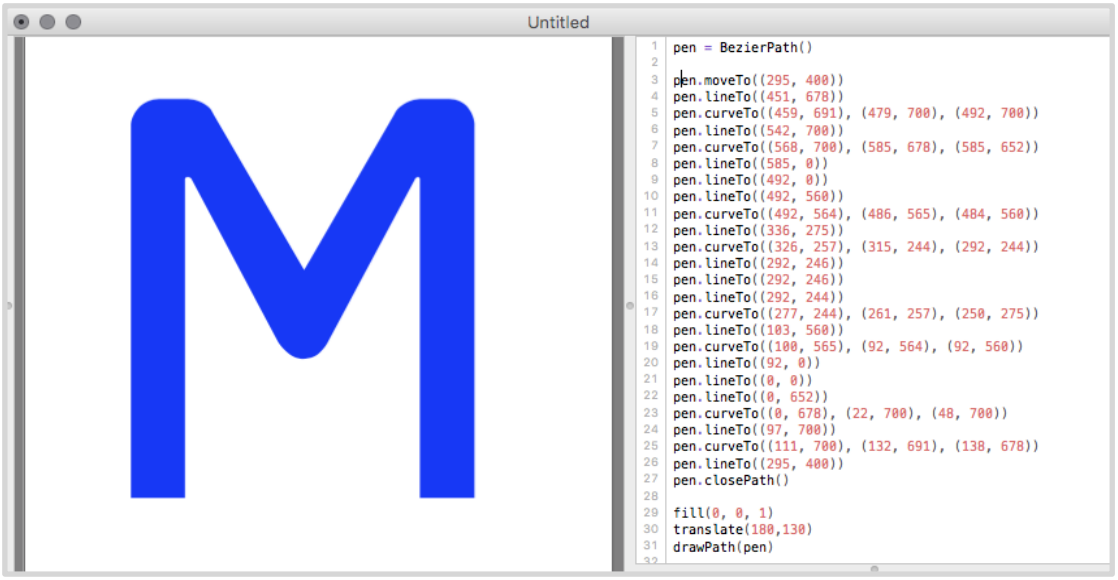
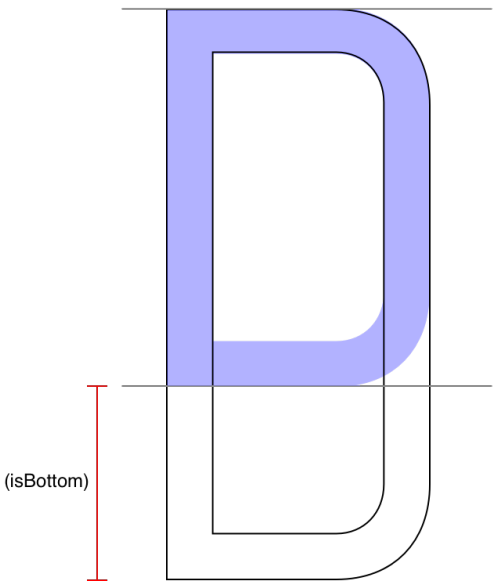


Fig.03 - Código em Python descrevendo os contornos do glifo "M".

Posteriormente foi criada uma função para cada glifo. Essa função continha as coordenadas dos pontos do contorno da letra, além de parâmetros para permitir que pontos especificos fossem modificados de maneira dinâmica. Para tanto foram definidos os seguintes parâmetros: "dx" para modificação horizontal, "dy" para vertical, além de "isBotton" indicando se a modificação vertical seria negativa ou positiva em relação a linha base. Exemplificado aqui pela função correspondente ao glifo "D":




```
def drawD(dx, dy, isBottom):
    pen = BezierPath()
    pen.moveTo((0, 700+dy))
    pen.lineTo((0, 0))
    pen.lineTo((314+dx, 0))
    pen.curveTo((420+dx, 0), (484+dx, 70), (484+dx, 176))
    pen.lineTo((484+dx, 523+dy))
    pen.curveTo((484+dx, 629+dy), (420+dx, 700+dy), (314+dx, 700+dy))
    pen.closePath()
    pen.moveTo((305+dx, 608+dy))
    pen.curveTo((356+dx, 608+dy), (392+dx, 568+dy), (392+dx, 515+dy))
    pen.lineTo((392+dx, 176))
    pen.curveTo((392+dx, 124), (356+dx, 92), (305+dx, 92))
    pen.lineTo((92, 92))
    pen.lineTo((92, 608+dy))
    pen.closePath()
    if isBottom:
        save()
        translate(0, -dy)
    drawPath(pen)
    if isBottom:
        restore()
```

Fig.04 - Variação vertical negativa no glifo "D" indicado no parâmetro "isBottom".

Com o código de todos os glifos no programa o passo seguinte foi criar uma interface de controle através do comando "Variables" que permite acessar uma

biblioteca de elementos básicos de interface gráfica tais como sliders, checkboxes, etc. Esse conjunto de controles permite que o usuário do sistema modifique os elementos dinâmicos da marca (inclusive de maneira aleatória). Uma vez que o resultado seja satisfatório o designer/usuário pode salvar uma versão em formato ".pdf" do logotipo ao selecionar a opção "Save_PDF". Dessa forma temos um design de identidade com com características dinâmicas e controles interativo construídos, de maneira relativamente simples, utilizando técnicas básicas de programação utilizando DrawBot.

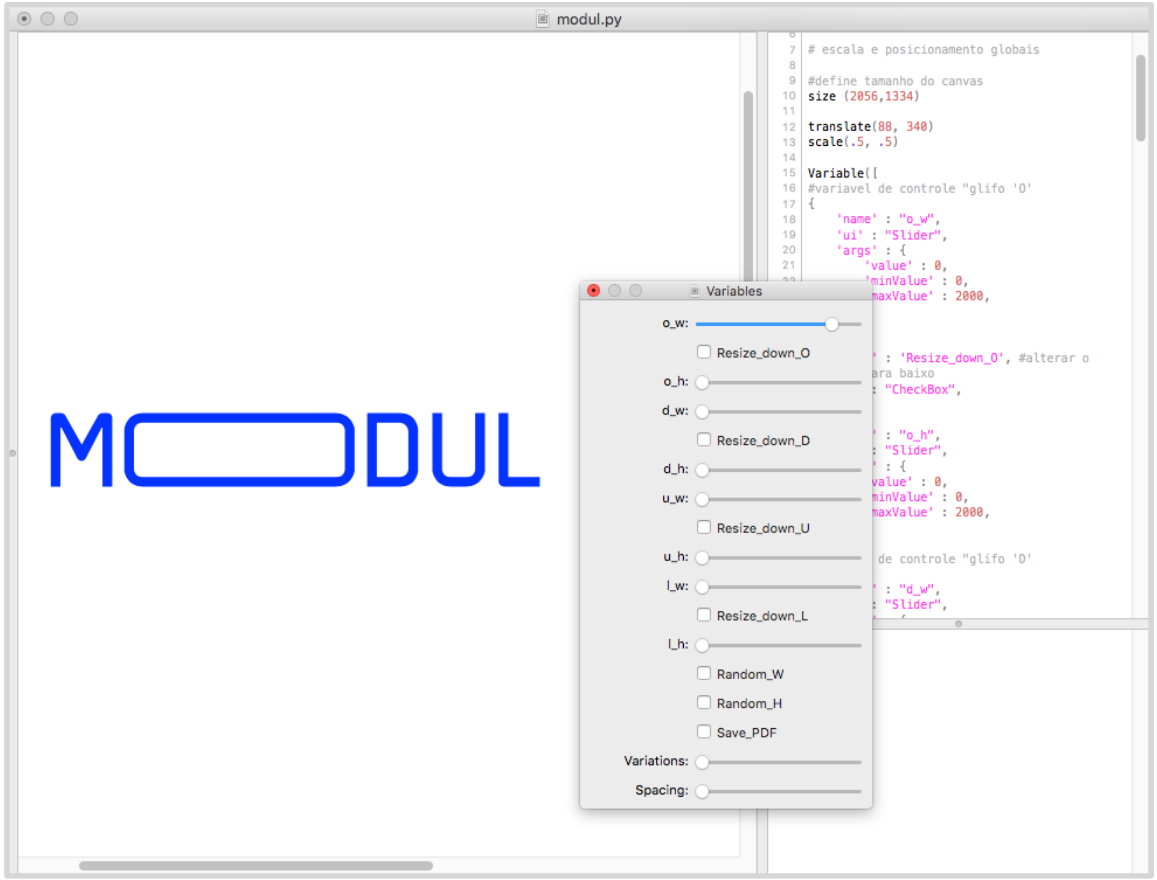


Fig.05 - Programa completo: código; resultado visual (logotipo) e interface de controles do aspectos dinâmicos dos glifos.

4.Conclusão

Apesar da programação ainda enfrentar resistências em áreas do conhecimento fora do campo das ciências exatas é clara a influência positiva que o pensamento computacional pode oferecer para outras áreas de conhecimento. O designer gráfico busca elaborar artefatos de comunicação visual que considerem as características técnicas e simbólicas das mídias os quais se destinam. Em um ecossistema cada vez mais digital, saber lidar com os códigos próprios do meio se mostra essêncial. Como apontado por Kay (1989), para aprender a ler e escrever nas novas mídias devemos não apenas usar as ferramentas mas também produzí-las. O DrawBot é uma ótima ferramenta de introdução de conceitos fundamentais de programação para designers, além de

se mostrar como uma opção muito interessante para ser utilizada em projetos de design gráfico com características dinâmicas.

5.Bibliografia

DRAWBOT. **DrawBot Documentation — DrawBot 3.81**. Disponível em: http://www.drawbot.com. Acesso em: 15 mar. 2016.

FLUSSER, Vilém. **O mundo codificado: por uma filosofia do design de comunicação.** Organizado por Rafael Cardoso. São Paulo, Cosac Naify, 2007.

GERSTNER, Karl. **Projetando Programas.** In: ARMSTRONG, Hellen. Teoria do Design Gráfico. São Paulo: Cosac Naify, 2015. Edição Kindle, posição 897.

GOOGLE. **Exploring computational thinking: what is CT?** 2011. Disponível em: http://www.google.com/edu/computational-thinking/what-is-ct.html. Acesso em: 15 mar.. 2016.

KAY, Alan. **User Interface: A Personal View**. In: Multimedia: from Wagner to virtual reality. Nova Iorque, W,W Norton & Company, 1989. p.121-131.

KAY, Alan. GOLDBERG, Adele. **Personal Dynamic Media.** In:IEEE Computer, v 10, n.3, p.31-41, mar. 1977. Reimpresso em: WARDRIP-FRUIN, Noah; MONTFORT, Nick. The New Media Reader. Cambridge: The MIT Press 2003. p.391-404.

MURRAY, Janet H. **Inventing the Medium: Principles of Interaction Design as a Cultural Practice.** Massachussetts: MIT Press, 2011.

SANT'ANNA, Hugo Cristo ; CARMO, J. C. ; GATTI, F. ; ROCHA, M. A. ; RANGEL, S. R. O. ; Neves, V. B. . **Da Arte Generativa ao Pensamento Computacional - Uma análise comparativa das plataformas de aprendizagem.** In: #11.ART Encontro Internacional de Arte e Tecnologia, 2012, Brasília. 10.art, 2012.