Livro interativo *Generative Magic* para introdução de Arte Generativa

Alysson F. Moreira[1], **Windson V. de Carvalho**[1]

[1]Instituto Universidade Virtual (UFC Virtual)
Universidade Federal do Ceara (UFC) – Fortaleza, CE – Brasil

alyssonfacanhamoreira@gmail.com, windson@virtual.ufc.br

Abstract. *Generative art creates infinite creative possibilities to those that use its knowledge to produce in this area. Programmers have clear technical capabilities to create generative pieces, the challenge is introducing the subjects of this area in a easy to grasp way and really focusing at the main subject and not deviate attention to common development obstacles . The objective of this study is to create a interactive book to introduce generative art to programmers, discuss the re-design process of a previous version of the book and a posterior usability de evaluation of the product. The usability evaluation results show great results for the proposed product. It concludes with a discussion about the book's merits and potencial future work to be done.*

Resumo. *A arte generativa cria uma infinidade de possibilidades criativas para aqueles que utilizam de seus conhecimentos para produzir nessa área. Programadores tem plenas capacidades técnicas de criar obras generativas, o desafio é introduzir os assuntos da área de maneira desmistificada e focando de fato no assunto base e não desviar a atenção para obstáculos comuns de desenvolvimento. O presente estudo tem como objetivo criar um livro interativo que introduz arte generativa para programadores, discutir o processo de re-design de uma versão prévia do livro e uma posterior avaliação de usabilidade. O resultado do teste de usabilidade mostra bons resultados tangendo a usabilidade do produto proposto. Conclui-se discutindo os méritos do livro e os potenciais trabalhos futuros a serem desenvolvidos.*

1. Introdução

Arte generativa pode ser compreendida como qualquer forma de expressão artística que se utiliza de algum tipo de sistema, podendo este ser, mas não se limitando a, programas de computadores [Galanter 2003].

Dentro dessa forma de expressão, diferentes materiais são usados como insumo criativo das obras incluindo dados estruturados, músicas e algoritmos. É uma tendência desse ramo sofrer influência de outras áreas, como ciência cognitiva, matemática e música. Essa forma de expressão gera produtos variados, como músicas, imagens, vídeos, peças de mobília.

A arte generativa pode ser usada inclusive como fator motivante no aprendizado de programação, principalmente para pessoas mais visualmente orientadas, como expõe [Hansen 2017].

Introduzir a arte generativa como forma de expressão para programadores, que comumente são associados com produção mais técnica do que propriamente artística, se mostra como problema instigante. Até para expandir o leque de possibilidades de produção desses profissionais. No entanto, existem barreiras de complexidade que alguns ambientes de desenvolvimento impõem.

Nos últimos anos, várias ferramentas têm sido criadas e usadas para ensinar programação de maneira mais desmistificada. O objetivo consiste em deixar menos abstratos os conceitos de programação e não se ater em partes mais avançadas de configuração. Algumas dessas tiveram relativo sucesso, como o *App Inventor* [Gray et al. 2012].

Tendo em vista esse paradigma de ensino mais acessível, em 2016 a empresa *Apple Inc.* anunciou uma nova aplicação exclusiva para sua linha de *tablets*, onde propunha um ambiente de ensino e aprendizado de programação mais intuitivo e interativo. A intuito foi prover uma aprendizagem prática e visual, onde o usuário lê sobre os conceitos apresentados e pode testar os conhecimentos adquiridos no próprio livro interativo. A empresa abriu o formato dos livros interativos que são executados nesse ambiente e fez possível o desenvolvimento deles por terceiros.

Esse ambiente cria uma clara oportunidade a ser explorada para introduzir a arte generativa a programadores, sejam eles experientes ou não.

Este artigo tem como objetivo geral criar um produto multimídia no formato de um livro interativo para introduzir conteúdos sobre arte generativa, na plataforma *Swift Playgrounds*. Para atingir tal objetivo, foram elencados os seguintes objetivos específicos: (i) realizar um *re-design* da primeira versão do livro interativo e (ii) avaliar a usabilidade do livro.

Este artigo se em seções que discutem a fundamentação teórica dessa pesquisa, sua metodologia, seu desenvolvimento e por fim uma conclusão que discute resultados e propõe trabalhos futuros.

2. Fundamentação Teórica

Essa seção tem como objetivo apresentar e discutir a fundamentação teórica, visando um melhor entendimento da abordagem do presente artigo.

2.1. Arte Generativa

Galanter define:

> A arte generativa se refere a qualquer prática artística que use de um sistema, como um conjunto de regras da linguagem natural, um programa de computador, uma máquina, ou outra invenção procedural, que é executado com algum grau de autonomia contribuindo ou resultando em um trabalho de arte completo. [Galanter 2003]

Essa definição engloba vários produtos e vários meios. O autor apresenta uma preocupação com a limitação que o conceito sofre, sendo geralmente limitado àquilo que os interlocutores produzem. [Galanter 2003]

Pearson cita Galanter, e adiciona que toda arte generativa tem o próprio sistema que o gera como autor, não o autor do sistema. Nessa perspectiva o autor do sistema age como um curador do que é esteticamente aceitável. [Pearson 2011]

A peça *AirCraft* é um exemplos de arte generativa que se enquadra na categoria de gráficos computadorizados e animação que Galanter define. Ela provê uma visualização de dados de tráfego aéreo, a figura 1 mostra um quadro da peça.

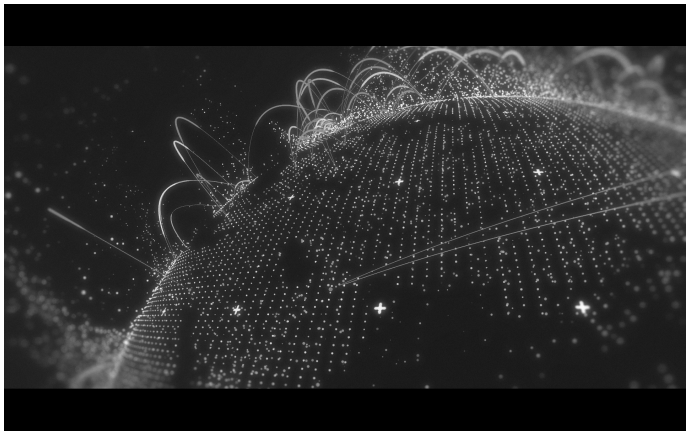

Figura 1. Quadro do vídeo generativo AirCraft [Guillou 2017].

2.2. *Playground Book*

Junto com o lançamento do *Swift Playgrounds*, foi tornado público também o formato *Playground Book*, específico para ele. Esse padrão era uma iteração, pois adicionava novas funcionalidades no formato *playground* que era destinado a outro ambiente da *Apple*: o *Xcode*.

O conteúdo do livro pode ser dividido em páginas e capítulos. O formato permite apresentar conteúdo textual formatado interpolado com código em *Swift* (linguagem de programação padrão para dispositivos *Apple*) na sessão *Contents* de cada página. O código é interpretado e executado, não limitando-se a ser uma ferramenta ilustrativa. É possível ainda explicitar áreas editáveis para direcionar a interação do usuário [Apple 2019a]. O *playground* pode opcionalmente apresentar uma interface gráfica adicional ao usuário, chamada de *Live View*. Ela pode dividir a tela com o conteúdo textual. É possível fazer com que o código digitado pelo usuário interfira diretamente no que é exibido na *Live View*, além dela ser interativa por meio de toques e gestos do usuário. Executando um *playground* no *Swift Playgrounds*, é possível usar vários dos recursos do *tablet* na interação, como sensores, câmera e *bluetooth*. A figura 2 destaca as sessões previamente citadas de uma página de *Playground Book*

Existem ainda funcionalidades que auxiliam durante a experiência de uso. A funcionalidade de glossário é uma delas, onde o desenvolvedor pode enumerar termos e suas definições de maneira estruturada e vincular essa definições às suas respectivas aparições no texto. Dessa forma o usuário pode acessar de maneira rápida as definições de termos encontrados no texto de maneira rápida.

Outra funcionalidade que contribui para a interação são os *assessments*, que é basicamente a possibilidade de mudar o estado da atual página para *pass* ou *fail*. Se o estado

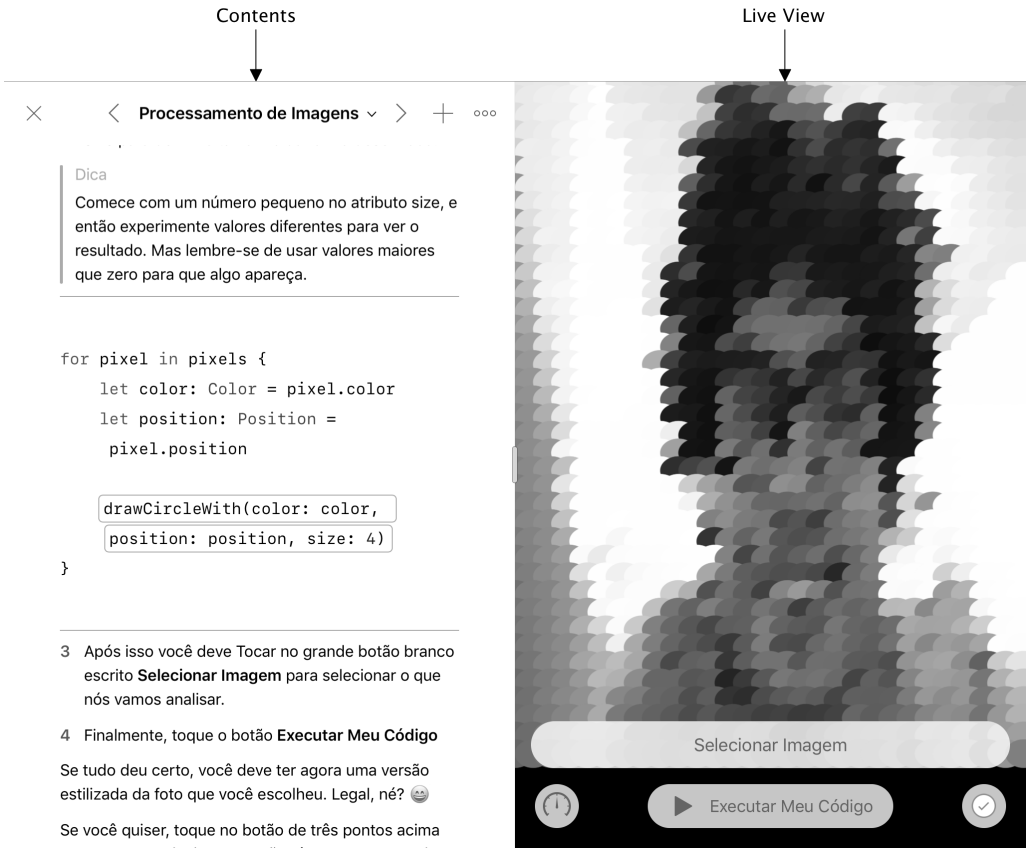

Figura 2. Foto que demonstra as divisões de uma página de um *Playground Book*.

da página muda para *pass* aquela tela é marcada de modo a demonstrar que o usuário conseguiu atingir os objetivos daquela interação, opcionalmente um texto de parabenização pode ser mostrado. Caso seja marcado como *fail* o aplicativo chama atenção para a sessão de dicas, caso o desenvolvedor tenha adicionado alguma.

2.3. *Generative Magic* - versão WWDC19

A World Wide Developer Conference - WWDC é um evento anual onde são apresentadas as novas tecnologias do ecossistema *Apple*. A empresa promove um concurso que premia projetos de estudantes ao redor do mundo com entrada para o evento e acomodação gratuita. A principal parte do processo de aplicação nesse concurso é desenvolver uma aplicação e desde 2017 essa aplicação tem sido um *Playground Book* para o *Swift Playgrounds* ou para *Xcode*.

Os requisitos do *playground* submetido para o concurso de 2019 foram:

1. O *playground* deve ser criado inteiramente pelo indivíduo que o submeteu;
2. Todo conteúdo textual deve estar em Inglês;
3. O arquivo submetido não pode ultrapassar 25 *megabytes* de tamanho;
4. O *playground* não deve depender de acesso à internet;
5. Deve ser executável no *Swift Playgrounds* 2.2 ou no *Xcode* 10.1

O prazo para entrega desde o anúncio do concurso (14/03/2019) eram de 10 dias. O resultado final saiu no dia 15/04/2019 [Apple 2019b].

Diante desse contexto foi criada a primeira versão do livro interativo *Generative Magic*. Essa versão continha uma introdução, duas páginas com conteúdo textual sobre arte generativa e interações, onde os usuários faziam experimentos que geravam intervenções gráficas em imagens escolhidas por eles, por fim uma conclusão que indicava outras ferramentas apropriadas para criação desse tipo de intervenção. O livro foi selecionado como um dos vencedores do concurso.

3. Metodologia

Essa pesquisa é divida majoritariamente em duas etapas: *re-design* do *Generative Magic* e avaliação da sua usabilidade. A seguir descreve-se essas etapas detalhadamente.

3.1. *Re-design* do *Generative Magic*

Uma aplicação digital moderna nunca está terminada, sempre há possíveis melhorias a serem feitas, adaptações para serem executadas em novas plataformas e a lista continua. *Generative Magic* não foge disso, a versão submetida para a seleção de bolsas para a WWDC estava estável e funcional, no entanto existiam claras oportunidades para melhorias que não foram exploradas pelo curto tempo para entregar o projeto.

Essas melhorias foram identificadas de uma maneira não estruturada, com testes e conversas informais com usuários, onde sugestões e problemas foram colhidos. As melhorias identificadas estão listadas a seguir:

- Desde a entrega do projeto uma nova versão do *Swift Playgrounds* foi lançada, e seria necessário portar todo o projeto para a nova versão.
- O livro foi desenvolvido inteiramente em inglês, pois era um dos requisitos do concurso que todo o conteúdo textual estivesse nessa língua. Portanto, seria necessário traduzir o livro.
- Havia mais uma interação envolvendo processamento de som planejada para estar na versão enviada, porém essa acabou sendo cortada por conta do escopo.
- Funcionalidades que proveriam auxílio para o usuário completar as suas atividades tinham sido pouco exploradas. Apenas a interação sobre ASCII *Art* (Uma forma de expressão que usa os caracteres e símbolos do *American Standard Code for Information Interchange*) possuía uma dica para interação, pois essa continha mais parâmetros e consequentemente mais possíveis pontos de falha para o usuário.
- Correção de uma imperfeição no sistema que providenciava sugestões de código de acesso rápido para o usuário.

3.2. Avaliação de Usabilidade

A norma ISO 9241-11 conceitua usabilidade como efetividade, eficiência e satisfação do usuário ao interagir com as funcionalidades de um sistema multimídia. Atingir bons índices nesses fatores apontam para facilidade de uso de um sistema. No presente artigo foi utilizado um método quantitativo de avaliação de usabilidade [ISO 9241-11:2018(en) 2018].

O objetivo é avaliar a usabilidade do *Playground* para o seu público-alvo. Busca-se entender se a aplicação está intuitiva e se os usuários conseguem captar os conceitos que a aplicação se propõe a passar. No teste com os usuários, deseja-se checar se estes conseguem realizar as interações propostas no *Playground* e entender seu processo de construção.

3.2.1. Procedimento

Foi aplicado um questionário para identificar o nível de proficiência de programação dos participantes. Esse questionário possuía uma pergunta objetiva e auto-avaliativa. Sendo as opções dessa pergunta baseadas na escala Likert [Likert 1933].

Após isso, foi feita uma breve introdução do aplicativo *Swift Playgrounds*, para que os participantes se familiarizassem com as telas e os controles do ambiente onde a aplicação era executada. Depois desse nivelamento, os participantes realizaram duas tarefas: criar uma intervenção gráfica com base em uma imagem usando formas básicas e a outra intervenção com caracteres.

3.2.2. Amostra

Dez estudantes do curso de Sistemas e Mídias Digitais escolhidos por conveniência, cursando o segundo semestre ou posterior, que entendam conceitos básicos de programação, como variáveis, estruturas de repetição, funções e estruturas de dados simples.

3.2.3. Instrumentos

Para esta avaliação foi aplicado um questionário que segue o SUS [Brooke 1996]. Este usando a escala Likert [Likert 1933] para medir a concordância dos participantes, sendo 1 para discordo totalmente e 5 para concordo totalmente. Após este, foi aplicado outro questionário com uma pergunta aberta, para receber críticas e sugestões dos participantes sobre do produto.

4. *Re-design* do *Generative Magic*

Essa seção discute o processo de *re-design* da versão prévia do *playground*.

Após a elencar os pontos a serem melhorados e expandidos no livro, estes foram dispostos em uma lista, onde foram dispostos em ordem de prioridade de execução. O critério para essa priorização foram:

- Tarefas que bloqueavam outras devem ter prioridade alta;
- Tarefas que aprimorem o conteúdo existente tem mais prioridade do que as de conteúdo novo (O intuito aqui era garantir qualidade acima de quantidade);
- Tarefas que demandassem pesquisa para serem executadas tinham mais prioridade.

Sendo esses critérios considerados na respectiva ordem que estão dispostos.

4.1. Conversão do projeto

A primeira tarefa a ser realizada foi portar todo o projeto para a nova versão do *Swift Playgrounds*. Para tal, foi baixado o novo modelo disponibilizado gratuitamente e *online* pela *Apple* para o desenvolvimento dos *playgrounds*.

Portar o código existente para a nova versão foi relativamente fácil, tendo em vista que não houve mudanças entre as versões do ambiente que demandassem mudanças no código antigo.

4.2. Localização

Logo em seguida o grosso do conteúdo textual foi localizado para português. Sendo assim, o conteúdo em inglês foi mantido junto com o português e aparece em dispositivos configurados para essa língua. Para realizar a localização foi necessário criar uma pasta chamada *"PrivateResources"* dentro dela foram criadas pastas com nomes que determinavam a qual linguagem aquele conteúdo se destinava. O nome consiste no código de identificação da língua segundos o padrão ISO 639.2, seguido da extensão *.lproj* [ISO 639-2:1998(en) 1998]. A figura 3 ilustra a estrutura de pastas de uma página do *playground*.

Figura 3. Estrutura de pastas de uma página do *Generative Magic*.

Todos os arquivos que contém texto localizado possuem a extensão ".strings" e contem pares de chave e valor de texto. Como ilustrado na figura 4.

Essas chaves são usadas no arquivo *Contents.swift* para identificar ao ambiente qual texto ele deve buscar para preencher aquele espaço, como é demonstrado na figura 5.

Foi necessário também, atentar para os nomes dos arquivos para que o *Swift Playground* os reconhecessem e usasse seus conteúdos. Por exemplo, o arquivo contendo o conteúdo textual de uma página deve ser nomeado *Prose.strings* para que o *Swift Playgrounds* o reconheça como tal.

Localizar o livro se mostrou uma das tarefas que demandou mais tempo, pois havia bastante conteúdo a ser localizado, todo conteúdo posteriormente criado tinha que ser localizado e o diferença do tamanho dos textos nas duas línguas geraram inconsistências visuais que tiveram de ser consertadas.

4.3. Implementação de *Assessments*

Implementar os assessments das interações foi bastante simples, já que as mesmas se apoiam em entradas de dados bem simples e de certa maneira já estava sendo feita uma validação dos dados inseridos antes de enviá-los para a *Live View*. Essa lógica foi isolada

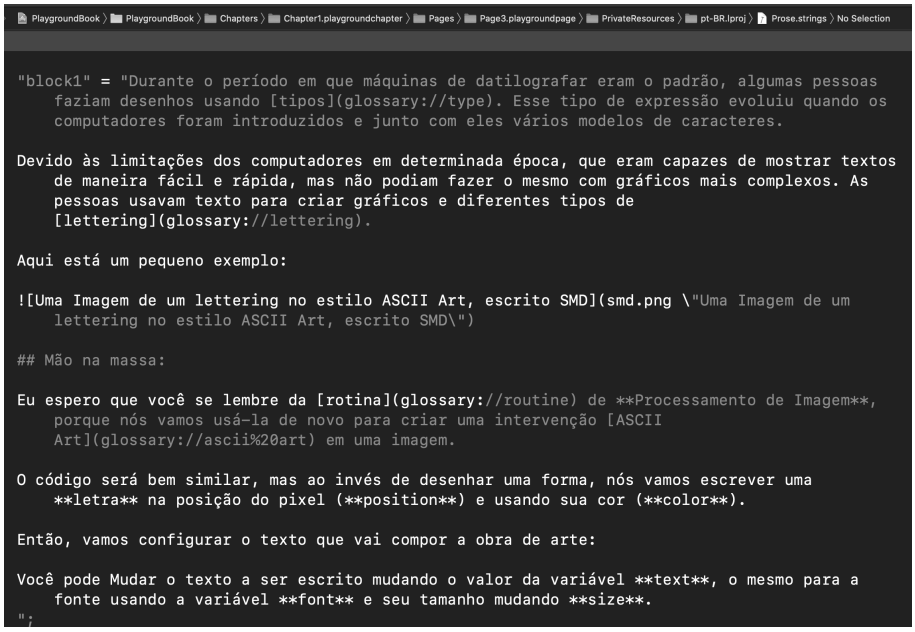

Figura 4. Sintaxe de um arquivo *.strings*.

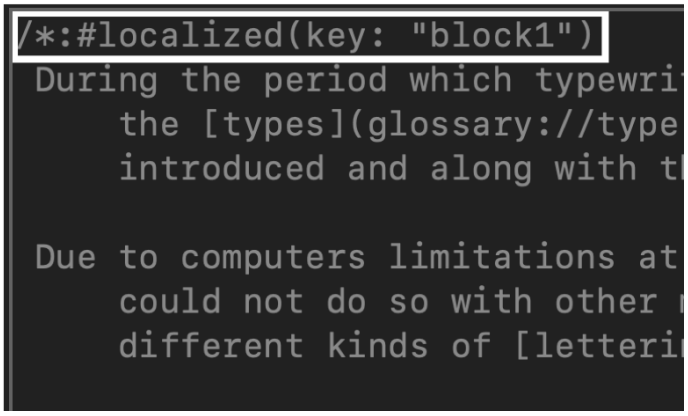

Figura 5. Demonstração de uso da chave de identificação do texto localizado.

em uma função em cada interação. Dentro dessas funções era testado se as entradas eram adequadas o estado era alterado para *pass*, caso contrário para *fail*. Um exemplo de função que isola essa lógica pode ser vista na figura 6

4.4. Criação da terceira interação

Uma terceira interação que lida com áudio foi implementada. Para tal, foi criado uma classe que analisa um arquivo de áudio presente no livro e produz um número referente ao tom do sinal da música enquanto ela toca. Esse número é então usado para determinar o tamanho e a cor de círculos que são desenhados quando o usuário tocar ou deslizar o dedo na *Live View*.

Toda a parte mais complexa é transparente para o usuário e apenas lhe é dado o poder de modificar poucos dados na interação, para que esta seja mais simples. O usuário deve implementar o conteúdo de duas funções, elas tem por nome *"didUpdatePitch"* e *"handleTouch"*.

```
func assessStatusFor(_ color: Color, _ position: Position, _ size: Double) -> Bool {
    if functionCalled {

        if color == initialColor && position == initialPosition && size > 0 {

            let passMessage = NSLocalizedString("PassI1", comment: "I Tried")

            PlaygroundPage.current.assessmentStatus = .pass(message: passMessage)
            return true

        } else {

            let hint1 = NSLocalizedString("Hint1I1", comment: "I Tried")
            let hint2 = NSLocalizedString("Hint2I1", comment: "I Tried")

            let hints = [hint1, hint2]

            PlaygroundPage.current.assessmentStatus = .fail(hints: hints, solution: nil)
            return false

        }


    } else {

        let hint = NSLocalizedString("HintGI1", comment: "I Tried")

        PlaygroundPage.current.assessmentStatus = .fail(hints: [hint], solution: nil)
        return false
    }


}
```

Figura 6. Demonstração de uso da chave de identificação do texto localizado.

No texto é dito para o usuário que a função *"didUpdatePitch"* é chamada toda vez que um novo valor é registrado pela classe que analisa o áudio, sendo passado o novo valor como parâmetro da função. A tarefa do usuário é dentro da função usar o valor passado como fator de interpolação entre as duas cores declaradas anteriormente, para isso ele deve usar uma função já implementada: *"getIntermediateColor"*. O resultado da chamada de *"getIntermediateColor"* deve ser armazenado na variável *"currentColor"* que é declarada no anteriormente na página. Logo após o usuário deve usar o valor passado para a função para interpolar entre o tamanho máximo e mínimo declarados anteriormente e salvar na variável *"currentSize"*. A figura 7 mostra uma possível entrada válida do usuário na função *"didUpdatePitch"*.

```
func didUpdatePitch(with value: Double) {

    currentSize = minimumSize + (maximumSize*value)
    currentColor = getIntermediateColor(from: initialColor, to: highestColor, by: value)

}
```

Figura 7. Exemplo de entrada válida na função *"didUpdatePitch"*.

O texto também informa o usuário que, toda vez que um toque é registrado no componente gráfico de borda preta presente na *Live View* é tocado a função *"handleTouch"* é chamada e é passado um objeto informando a posição do toque. O usuário deve então usar a função *"drawCircleWith"*, que foi previamente usada na primeira interação, e passa para a mesma a variável com a cor interpolada, a variável *"position"* que a função recebe e o valor interpolado de tamanho. A figura 8 mostra um exemplo de entrada válida na função *"handleTouch"*.

```
func handleTouch(at position: Position) {

    drawCircleWith(color: currentColor, position: position, size: currentSize)

}
```

Figura 8. Exemplo de entrada válida na função *”handleTouch”*.

Ao apertar o botão ”Executar o Meu Código” (Figura 2), caso os requisitos citados acima sejam satisfeitos uma mensagem de sucesso será mostrada ao usuário onde o mesmo é encorajado a apertar o botão *”Play”* para tocar a música e começar a ”pintar”no local delimitado na *Live View*. A seguir na figura 9, possível ver o resultado gráfico da interação.

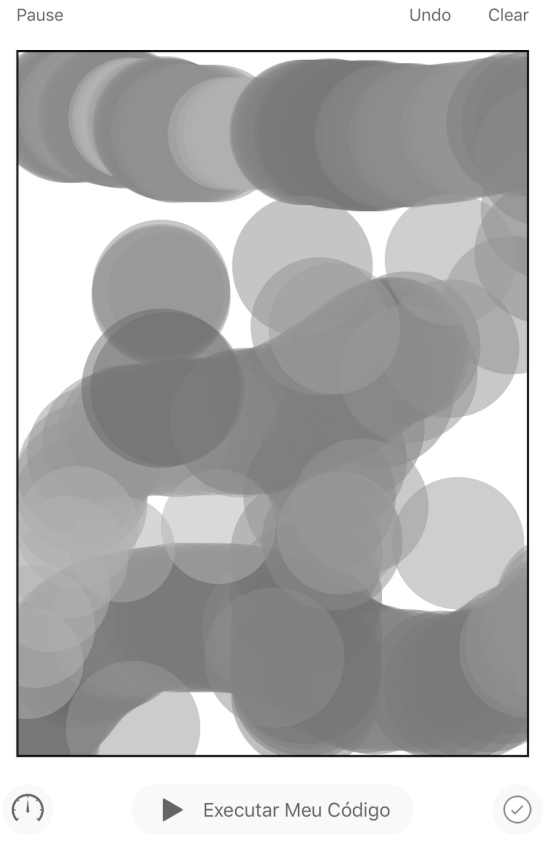

Figura 9. Exemplo de produção realizada na interação de processamento de áudio.

4.5. Outras Adições e correções

A adição do glossário foi bem trivial. Foi uma questão de levantar os termos, escrever suas definições nas duas línguas e vincular as aparições dos termos no texto com o glossário. O mesmo vale para as dicas para as interações, que foi mais um trabalho de escrita do que de programação em si. As figuras 10 e 11 mostram a funcionalidade de glossário em ação.

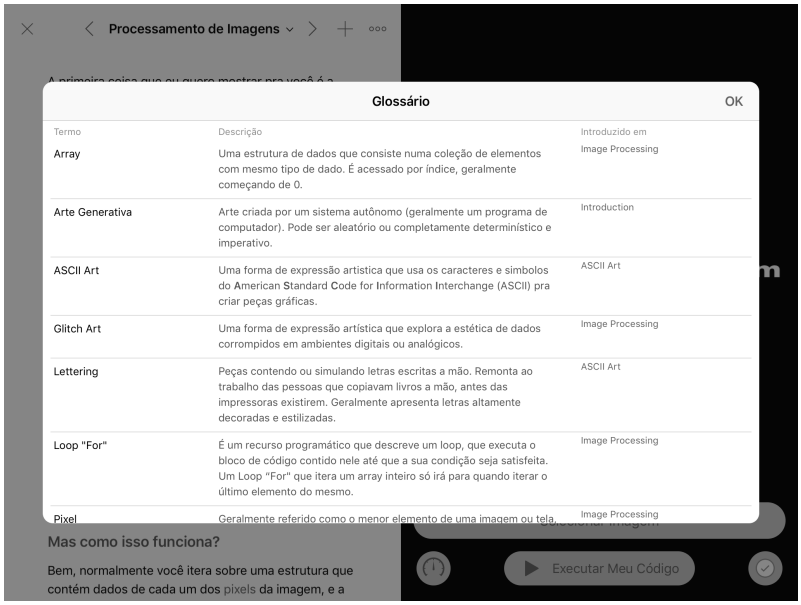

Figura 10. Tela de Glossário.

Por último um pequeno problema que o livro tinha para mostrar sugestões de acesso rápido a símbolos e assinatura de funções foi corrigido. Após isso foi possível usar nomes de função mais longos e as sugestões apareciam mais rapidamente.

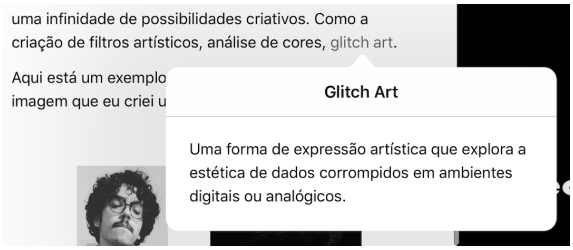

Figura 11. Funcionalidade de acesso rápido a definições do glossário.

5. Avaliação de Usabilidade e discussão de resultados

Antes que o processo de avaliação propriamente dito fosse iniciado, todos os participantes foram apresentados a um termo de consentimento de participação e uso dos dados produzidos por eles nesse artigo. Todos foram orientados a ler o documento e caso concordassem com os termos assinassem e só então iniciassem o processo de avaliação.

Logo após isso, todos preencheram um formulário para confirmar se estavam no perfil da entrevista: Alunos da graduação de Sistemas e Mídias Digitais, com conhecimentos básicos ou superior em programação.

Após a introdução do *Swift Playgrounds*, propõe-se uma avaliação com intuito de quantificar a usabilidade do *Generative Magic*. A avaliação é baseada no sistema SUS, que consiste em dez perguntas que visam identificar quanto o sistema avaliado é efetivo, eficiente, e satisfatório para o usuário [Brooke 1996].

Define-se uma amostra de dez estudantes composta por alunos do curso de Sistemas e Mídias Digitais, com conhecimentos básicos ou superior em programação. Após

isso, os participantes são orientados a utilizar o livro com os seguintes objetivos:

- Criar uma intervenção gráfica com formas geométricas;
- Criar uma intervenção gráfica com letras.

Após a execução das atividades, os participantes respondem a uma questionário em três etapas:

1. Perguntas referentes ao SUS;
2. Colhimento de *feedbacks* e sugestões.

As duas primeiras consistindo de perguntas objetivas e possuíam respostas baseadas na escala Likert: 1 - Discordo completamente; 2 - Discordo parcialmente; 3 - Indiferente; 4 - Concordo parcialmente; 5 - Concordo completamente [Likert 1933].

Cada teste demorou cerca de trinta minutos e todos foram executados no mesmo dispositivo (*iPad* 6ª geração).

A metodologia de avaliação escolhida propõe um calculo para gerar um número que varia de 0 à 100, sendo 68 a média. Esse calculo da seguinte forma:

- Subtrair 1 das respostas das questões 1,3,5,7 e 9;
- Para as questões 2,4,6,8 e 10, o valor será 5 menos a resposta.

Sendo assim os valores variam de 0 à 4, após esse calculo, soma-se todos os valores do questionário sus e multiplica-se por 2.5. O resultado dessa conta é a nota final para aquela avaliação. Somou-se as notas para realizar um calculo de média simples.

O resultado desse calculo para a avaliação do produto apresentado foi de 76,5. No artigo *Determining what individual sus scores mean*, um critério é proposto para classificar resultados de avaliações SUS de maneira qualificar os diferentes níveis de bons e maus resultados.[Bangor et al. 2009] O produto proposto está classificado nessa escala como Bom, e apenas a 3,5 pontos de ser classificado como Excelente.

Por último havia uma seção onde os participantes podiam escrever sugestões de melhorias e comentários sobre a experiência com o *Generative Magic*. Dentre as melhorias sugeridas destacam-se as seguintes:

- Melhorar o *feedback* ao usuário quando uma imagem é selecionada nas interações
- Aumentar o número de interações.

O usuários também destacaram como ponto positivos a linguagem simples e o resultado das interações.

6. Conclusão

Nesse artigo foi apresentado o livro *Generative Magic* e o seu potencial para introduzir arte generativa para programadores de uma maneira interativa. Durante o desenvolvimento da pesquisa todos os objetivos foram cumpridos. O *re-design* e a avaliação de usabilidade, ambos com bons resultados.

O livro demonstrou, de modo geral, bons índices na avaliação de usabilidade, comprovando sua facilidade de uso. Além disso, por meio da avaliação também foram identificadas possíveis melhorias a serem desenvolvidas no sistema.

Diante desse cenário, propõe-se como trabalhos futuros a disponibilização de todo o código fonte atualizado da presente versão do *Generative Magic* em um repositório público na plataforma *GitHub*, com documentação visando explicar seus componentes; Disponibilizar de maneira mais simplificada o arquivo executável do livro, para que mais pessoas tenham acesso facilitado; Melhorar o *feedback* das interações atuais, como o uso de ícones para deixar mais claro para o usuário que uma imagem foi selecionada em interações em que isso é necessário; Portar o livro para a nova versão do *Swift Playgrounds* que saiu durante o desenvolvimento do presente artigo e realizar novos testes de usabilidade para avaliar a nova interação criada.

Referências

Apple (2019a). Swift Playgrounds documentation. `https://developer.apple.com/documentation/swift_playgrounds`.

Apple (2019b). WWDC19 scholarshipsn. `https://developer.apple.com/wwdc19/scholarships/`.

Bangor, A., Kortum, P., and Miller, J. (2009). Determining what individual sus scores mean: Adding an adjective rating scale. volume 4, pages 114–123.

Brooke, J. (1996). Sus - a quick and dirty usability scale. 189(194):4–7.

Galanter, P. (2003). What is generative art? complexity theory as a context for art theory. In *GA2003 – 6th Generative Art Conference*.

Gray, J., Abelson, H., Wolber, D., and Friend, M. (2012). Teaching cs principles with app inventor. In *Proceedings of the 50th Annual Southeast Regional Conference*, ACM-SE '12, pages 405–406, New York, NY, USA. ACM.

Guillou, A. L. (2017). Realtime air traffic visualisation prototype. `https://vimeo.com/215513115`.

Hansen, S. M. (2017). Deconstruction/reconstruction: A pedagogic method for teaching programming to graphic designers.

ISO 639-2:1998(en) (1998). Codes for the representation of names of languages — Part 2: Alpha-3 code. Standard, International Organization for Standardization, Geneva, CH.

ISO 9241-11:2018(en) (2018). Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts. Standard, International Organization for Standardization, Geneva, CH.

Likert, R. (1932–1933). A technique for the measurement of attitudes. volume 22, pages 5–55.

Pearson, M. (2011). *Generative Art*. Manning Publications.

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M836l Moreira, Alysson Façanha.
Livro interativo Generative Magic para introdução de Arte Generativa / Alysson Façanha Moreira. – 2019.
13 f. : il.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Instituto UFC Virtual, Curso de Sistemas e Mídias Digitais, Fortaleza, 2019.
Orientação: Prof. Dr. Windson Viana de Carvalho.

1. Arte Generativa. 2. Livro Interativo. 3. Swift Playgrounds. I. Título.

CDD 302.23
