Universidade Federal de Pernambuco

Departamento de Informática

Curso de Engenharia da Computação

Reduzindo o Atrito: Facilitando a Implementação e Visualização de Acessibilidade em Aplicativos

Trabalho de Conclusão de Curso de Graduação

por

Samuel Brasileiro dos Santos Neto

Orientador: Prof. Kiev Santos da Gama

Recife, Outubro / 2024

Samuel Brasileiro dos Santos Neto

Reduzindo o Atrito: Facilitando a Implementação e Visualização de Acessibilidade em Aplicativos

Monografia apresentada ao Curso de Engenharia da Computação, como requisito parcial para a obtenção do Título de Bacharel em Engenharia da Computação, Centro de Informática da Universidade Federal de Pernambuco.

Orientador: Prof. Kiev Santos da Gama

Recife

2024

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Santos Neto, Samuel Brasileiro dos.
Reduzindo o Atrito: Facilitando a Implementação e Visualização de Acessibilidade em Aplicativos / Samuel Brasileiro dos Santos Neto. - Recife, 2024.
37 p. : il., tab.

Orientador(a): Kiev Santos da Gama
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Engenharia da Computação - Bacharelado, 2024.
Inclui referências.

1. Acessibilidade Digital. 2. Desenvolvimento iOS. I. Gama, Kiev Santos da. (Orientação). II. Título.

000 CDD (22.ed.)

SAMUEL BRASILEIRO DOS SANTOS NETO

Reduzindo o Atrito: Facilitando a Implementação e Visualização de Acessibilidade em Aplicativos

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia da Computação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de bacharel em Engenharia da Computação.

Aprovado em: 17/10/2024

BANCA EXAMINADORA

_____

Prof. Dr. Kiev Santos da Gama (Orientador)

Universidade Federal de Pernambuco

_____

Prof. Dr. Leopoldo Motta Teixeira (Examinador Interno)

Universidade Federal de Pernambuco

Agradecimentos

Agradeço a meus pais, por me ensinarem, desde sempre, o valor de ajudar os outros. A Pablo, pelo apoio incondicional, amor e companheirismo. À minha irmã, por estar sempre por perto e me apoiar em todo os momentos. Aos meus amigos, que de tantas formas tornaram essa caminhada mais leve e feliz. Ao meu orientador, Kiev, pela paciência, conselhos e por acreditar no meu trabalho. E, por fim, a todas as pessoas que de alguma forma contribuíram para que eu chegasse até aqui e que, na minha vida, me ajudaram a ser uma pessoa melhor. Muito obrigado.

RESUMO

Em uma sociedade em que, durante o desenvolvimento de aplicativos, não são amplamente pensadas as necessidades de adaptação de pessoas que vivem com algum tipo de deficiência, se fez necessário uma pesquisa para compreender as dificuldades para tal implementação, em busca da contribuição para a melhoria da acessibilidade no ambiente digital, propondo uma solução viável e facilitadora. Esta solução visa auxiliar os desenvolvedores a criar meios mais eficazes para reduzir o atrito no desenvolvimento de aplicativos acessíveis, permitindo a simulação de cenários que atendam às necessidades de acessibilidade.

Palavras-chave: Acessibilidade Digital, Desenvolvimento de Aplicativos Móveis, SwiftUI, Previews no Xcode.

ABSTRACT

In a world in which, during the development of applications, the adaptation needs of people living with some kind of disability are not widely considered, research was needed to understand the difficulties of such implementation, in order to contribute the improvement of accessibility in the digital environment by proposing a viable and facilitating solution. This solution seeks to help developers in creating more effective ways to reduce friction in the development of accessible applications, allowing the simulation of scenarios that meet accessibility needs.

Keywords: Digital Accessibility, Mobile Applications Development, SwiftUI, Xcode Previews.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 Contexto

Cerca de 15% das pessoas no planeta vivem com algum tipo de deficiência [1], o que significa que a acessibilidade é essencial em todas as dinâmicas da vida cotidiana. Como os dispositivos móveis se tornaram o principal meio de comunicação no século XXI [2], é importante que as experiências digitais contemplem a todos. Em *Vivendo na Informação*, Jorge Arango acentua a importância de criar espaços digitais acessíveis e responsáveis, proporcionando benefícios a todos os usuários, independentemente de suas circunstâncias [3].

A inclusão digital sofre com a falta de acessibilidade nos aplicativos móveis, que muitas vezes apresentam não só problemas técnicos, mas também obstáculos de design. Uma pesquisa da WebAIM constatou que 98.1% dos sites tinham pelo menos um problema de acessibilidade [4]. Dificuldades semelhantes podem ser observadas em aplicativos móveis, como botões não rotulados para leitores de tela e contraste inadequado entre texto e plano de fundo. Esses problemas afetam diretamente a experiência do usuário, dificultando o uso eficiente dos aplicativos por pessoas com deficiências cognitivas, motoras e visuais.

Por exemplo, a análise de Yan et al. (2020) revelou que muitos aplicativos mundialmente conhecidos não suportam leitores de tela suficientemente bem e causam problemas de navegação para usuários cegos ou com deficiência visual [7]. Leite et al. (2021) observam que, embora os desenvolvedores brasileiros tenham um bom nível de conhecimento sobre acessibilidade, ainda há uma falta de adoção de práticas acessíveis [10].

Mesmo com a presença de regras de acessibilidade, como as *Web Content Accessibility Guidelines* (WCAG) [5] e as Diretrizes da Apple [6], muitos aplicativos conhecidos não atendem aos critérios de acessibilidade. Estudos indicam que a maioria dos aplicativos mais populares ainda tem muitos problemas não resolvidos, apesar da disponibilidade de padrões e diretrizes [7]. Embora a conscientização dos desenvolvedores e o treinamento contínuo sejam importantes, é preciso lidar com o atrito no processo de desenvolvimento de acessibilidade [8]. Esse atrito se manifesta como a dificuldade de testar interfaces para várias deficiências sem ferramentas integradas, o que leva a interfaces que não são apropriadas para usuários com necessidades especiais.

As ferramentas de Pré-Visualizações [9] de desenvolvimento iOS permitem a visualização de interfaces enquanto estão sendo desenvolvidas. No entanto, o SwiftUI, biblioteca de desenvolvimento de interfaces visuais da Apple [12], ainda não oferece suporte suficiente para testes visuais simples e acessíveis, exigindo modificações manuais trabalhosas [11]. A inclusão total pode ser prejudicada ao experimentar diferentes combinações de contraste e tamanho de fonte sem o uso de ferramentas especializadas.

O objetivo deste artigo é reduzir o atrito no desenvolvimento de aplicativos acessíveis, oferecendo uma maneira de aprimorar a criação e o teste visual de acessibilidade em aplicativos móveis. A implementação adequada da acessibilidade melhora não apenas a experiência dos usuários com deficiência, mas também a de todos os outros usuários. Todos têm o direito de participar plenamente da comunidade digital, e as práticas de acessibilidade ajudam a criar uma sociedade mais inclusiva.

1.2 Objetivos

Esta pesquisa busca contribuir para a melhoria da acessibilidade no mundo digital, analisando e criando uma solução viável para lidar com problemas de acessibilidade em aplicativos móveis e diminuir esse atrito no processo de desenvolvimento. Os objetivos são:

- Examinar as principais necessidades de acessibilidade que devem ser incluídas na criação de aplicativos móveis, fazendo referência aos padrões e diretrizes existentes, como o WCAG 2.2 e as diretrizes da Apple.
- Examinar os principais desafios que os desenvolvedores de aplicações iOS enfrentam ao colocar a acessibilidade em prática.
- Fornecer uma solução viável que ajude e incentive os desenvolvedores a criar aplicativos acessíveis.
- Verificar a eficácia da solução produzida, testando-a com desenvolvedores reais. A utilização de pesquisas, como o *Post-Study System Usability Questionnaire* (PSSUQ) [21], será parte do processo de validação.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Acessibilidade digital

O propósito da acessibilidade digital é garantir que todos os usuários, incluindo os com deficiências, sejam capazes de utilizar os produtos e serviços digitais. Dessa forma, o governo brasileiro destaca a acessibilidade digital como um direito fundamental [14].

Contudo, a acessibilidade digital ainda não é amplamente aplicada. A Figura 1 mostra que a grande parte das páginas iniciais da internet não seguem alguns requisitos básicos de acessibilidade [5], formando uma realidade onde a maioria dos sites não consegue oferecer uma experiência digital inclusiva.

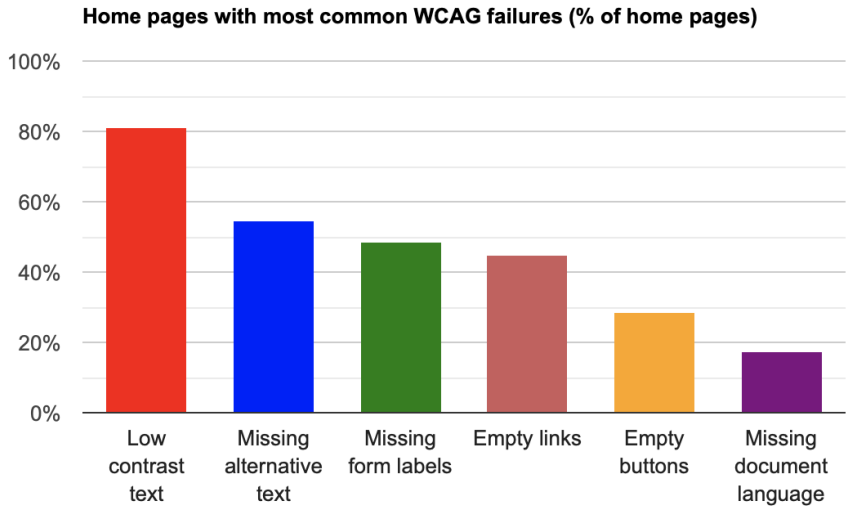


Figura 1: Homepages com falhas mais comuns da WCAG, conforme estudo da WebAIM [4].

A **Figura 1** apresenta os principais problemas de acessibilidade encontrados nas páginas iniciais, destacando *contraste baixo* (presente em mais de 80% dos sites) e *ausência de texto alternativo em imagens*. Essas falhas impactam diretamente em indivíduos com limitações visuais, que necessitam de recursos como leitores de tela para acessar esses sites.

No contexto de aplicativos móveis, esses desafios são igualmente presentes [16] e se tornam ainda mais latentes pela crescente adoção e dependência da tecnologia móvel [2]. Dessa forma, garantir a conformidade com padrões acessíveis se torna uma necessidade importante para apoiar a acessibilidade digital.

2.2 Diretrizes de acessibilidade

Um conjunto de recomendações denominado *Web Content Accessibility Guidelines* (WCAG), ou Diretrizes de Acessibilidade para Conteúdo da Web, foi criado para melhorar a acessibilidade do conteúdo para pessoas com deficiências. A versão mais recente, WCAG 2.2, estabelece quatro princípios fundamentais: perceptibilidade, operabilidade, compreensibilidade e robustez [5] e devem ser considerados no desenvolvimento de interfaces acessíveis.

Além das WCAG, a Apple também oferece diretrizes específicas para a criação de aplicativos iOS acessíveis, através das *Human Interface Guidelines* (HIG) [15]. As diretrizes HIG ajudam os desenvolvedores a projetar interfaces de usuário confiáveis, compreensíveis e acessíveis. A Apple promove o uso de componentes de interface nativos, que devem estar disponíveis por padrão e incluir layouts responsivos e controles padrão. Além disso, as recomendações do HIG enfatizam a importância de usar cores e contrastes adequados, VoiceOver [26] e tipos dinâmicos como tecnologias de assistência para garantir que todos os usuários, independentemente da capacidade, possam utilizar os aplicativos de forma eficiente.

As diretrizes HIG incluem instruções abrangentes sobre como implementar e avaliar a acessibilidade em aplicativos. Por exemplo, elas aconselham o uso de legendas e rótulos descritivos para os componentes da interface do usuário, certificando-se de que todas as operações possam ser realizadas com comandos de voz ou teclado e adicionando descrições de acessibilidade para todos os elementos. [15]

2.3 Desafios da implementação da acessibilidade

A implementação de acessibilidade em aplicativos móveis enfrenta obstáculos que podem ser classificados em três categorias principais, no escopo do artigo: organizacionais, tecnológicos e interpessoais [10].

2.3.1 Obstáculos tecnológicos

A ausência de ferramentas integradas para testar e implementar acessibilidade é um dos maiores desafios tecnológicos enfrentados pelos desenvolvedores [11]. Ferramentas como o *Accessibility Inspector* do Xcode ajudam a verificar a conformidade dos elementos

da interface, mas possuem limitações, como a falta de suporte para testar combinações de contraste ou simular diferentes deficiências visuais de forma eficaz e rápida [25]. Além disso, as diretrizes, como as WCAG, podem ser vistas como complexas e difíceis de implementar corretamente, levando a soluções de acessibilidade inconsistentes [10].

2.3.2 Obstáculos organizacionais

A falta de prioridade da acessibilidade no início do desenvolvimento de aplicativos é outro problema recorrente [10]. Sob a pressão de prazos de entrega apertados e a falta de recursos dedicados, muitas equipes de desenvolvimento deixam a acessibilidade como um elemento secundário ou negligenciam completamente sua implementação. Adicionalmente, é recorrente que pequenas equipes de desenvolvimento não tenham as ferramentas necessárias para testar e implementar totalmente os recursos de acessibilidade [24].

2.3.3 Obstáculos interpessoais

É comum que os desenvolvedores não recebam treinamento adequado sobre acessibilidade, o que os deixa sem saber quais são as melhores maneiras de colocar em prática os métodos acessíveis [10]. Muitos desenvolvedores não têm conhecimento suficiente sobre as melhores práticas de acessibilidade ou não estão cientes das necessidades específicas dos usuários com deficiências. Além disso, embora o valor da acessibilidade esteja se tornando mais amplamente reconhecido, ainda há uma grande lacuna na adoção de técnicas acessíveis, principalmente entre os desenvolvedores que não estão familiarizados com os requisitos dos usuários com deficiências [10] [16].

2.4 Inspeção de acessibilidade

O processo de inspeção de acessibilidade envolve a utilização de ferramentas e estratégias para reconhecer e resolver problemas de acessibilidade durante o desenvolvimento de aplicativos.

Antes de publicar o produto final, os desenvolvedores podem examinar, em tempo de desenvolvimento, como suas telas operam em vários contextos com recursos de visualização como os incorporados no Xcode [9] para desenvolvimento do iOS. Quando se trata de acessibilidade, a visualização é essencial para testar determinados cenários e modificar

as interfaces de usuário para que possam ser usadas por todos. No entanto, pode haver lacunas de usabilidade para usuários com determinadas deficiências como resultado da frequente falta de suporte completo para todas as opções acessíveis dessas ferramentas. Por exemplo, o ajuste manual de tamanhos e contrastes de texto ao examinar várias combinações pode ser trabalhoso e propenso a erros [7].

Apesar de sua força como ferramenta para o desenvolvimento rápido de interfaces, o Xcode não tem a funcionalidade que permitiria aos desenvolvedores como suas interfaces funcionariam com várias configurações específicas de acessibilidade [9]. Devido a essa restrição, um grande número de desenvolvedores precisa modificar manualmente as configurações de acessibilidade, o que pode resultar em erros ou na exclusão de indivíduos com necessidades específicas. Uma grande desvantagem do Xcode é seu suporte insuficiente para visualizações de acessibilidade. Para avaliar a acessibilidade de suas interfaces, muitos desenvolvedores agora usam procedimentos manuais ou tecnologias de terceiros, que podem não ser confiáveis e gerar desperdício [7].

Uma ferramenta popular do Xcode que permite verificar e modificar as propriedades de acessibilidade dos componentes da interface do usuário é o Accessibility Inspector [25]. Ele verifica se todos os itens têm descrições suficientes, se o VoiceOver pode ser usado para interagir com eles e se a navegação é clara e fácil de usar [26]. Para garantir que as interfaces possam ser usadas por pessoas com deficiências visuais e de movimento, é necessário usar essa ferramenta. No entanto, o Accessibility Inspector tem várias desvantagens. É um desafio investigar determinadas situações e vários cenários ao mesmo tempo, pois está limitado a programas que estão em operação no momento [25]. O daltonismo é um dos problemas de acessibilidade que não é bem abordado [24]. Além disso, por ser uma ferramenta externa ao Xcode, ela coloca uma barreira entre os desenvolvedores e a implementação da acessibilidade, exigindo etapas extras no processo de desenvolvimento [10].

O uso de ferramentas de inspeção de acessibilidade dentro de IDEs de desenvolvimento, como o Xcode, que se integram diretamente ao ambiente de desenvolvimento, é essencial para contornar essas restrições. Já foi demonstrado por ferramentas integradas que permitir testes em várias configurações de acessibilidade com ajustes rápidos e precisos [11] [20], seja por meio de visualização em tempo de desenvolvimento, reduz o atrito do processo de desenvolvimento e garante que a implementação da acessibilidade esteja

em andamento e presente desde o início do projeto [27]. A adoção de tais ferramentas melhorará a experiência de todos os usuários, incentivando uma abordagem de design mais inclusiva e eficaz, além de ajudar as pessoas com deficiências [27].

2.5 Revisão da Literatura

Inicialmente, foi realizada uma uma análise da literatura para embasar teoricamente o desenvolvimento da biblioteca de acessibilidade. Foram analisadas as Diretrizes de Acessibilidade para Conteúdo Web (WCAG) 2.2 [5] e as Diretrizes de Interface Humana (HIG) da Apple [6]. Essas diretrizes serviram de base para a definição dos requisitos da ferramenta e para garantir que a solução proposta estivesse alinhada com as melhores práticas em acessibilidade.

Estudos relevantes, como o de Leite et al. (2021) [10], que aborda as barreiras enfrentadas por desenvolvedores brasileiros na implementação de acessibilidade, e Oliveira & Filgueiras (2018) [11], que discutem as limitações das ferramentas de inspeção de acessibilidade, foram fundamentais para direcionar as funcionalidades a serem implementadas.

Para garantir que as melhores práticas e recomendações atuais fossem levadas em consideração, a pesquisa bibliográfica ofereceu uma sólida base teórica para a criação da solução sugerida.

2.6 Trabalhos relacionados

A comunidade científica abordou detalhadamente a investigação e a criação de métodos para melhorar a acessibilidade em aplicativos móveis. A ferramenta *ALib* [17] faz uma contribuição substancial. Ela tem o objetivo de dar aos elementos de um sistema de design acessibilidade inerente, tornando a interface mais amigável para todos os usuários.

Outro trabalho relacionado é o projeto *CodeForAll* [18] que propõe um conjunto de componentes modulares que facilitam a criação de aplicativos acessíveis utilizando SwiftUI [12]. O foco principal da biblioteca é reduzir as barreiras técnicas enfrentadas pelos desenvolvedores ao implementar práticas de acessibilidade, como as diretrizes da WCAG [5] e ARIA (Accessible Rich Internet Applications) [19], tornando o processo de desenvolvimento acessível mais intuitivo e ágil, sendo uma abordagem complementar ao objetivo deste trabalho.

Além disso, *Enhanced UI Automator Viewer with Improved Android Accessibility Evaluation Features* é outro estudo pertinente [20]. Esse estudo sugeriu aprimoramentos no UI Automator Viewer, incluindo novas funcionalidades para avaliar a acessibilidade em aplicativos Android. O estudo ilustrou como instrumentos de avaliação de ponta podem ajudar a encontrar e corrigir problemas de acessibilidade, resultando em mais experiências de usuário acessíveis.

3 METODOLOGIA

3.1 Desenvolvimento da Ferramenta

O desenvolvimento da biblioteca foi conduzido utilizando uma abordagem iterativa, com ciclos de design, implementação e testes. O processo foi guiado pelos seguintes passos:

3.1.1 Definição dos Requisitos Funcionais

Com base na revisão da literatura, foram definidos os requisitos da ferramenta de acessibilidade, focando em funcionalidades que facilitassem a visualização de problemas de acessibilidade durante o desenvolvimento de interfaces. O objetivo era permitir que os desenvolvedores testassem suas interfaces para diferentes cenários de acessibilidade sem a necessidade de modificar manualmente as configurações ou utilizar ferramentas externas.

3.1.2 Implementação Técnica

A ferramenta foi desenvolvida utilizando *Swift* e *SwiftUI*, aproveitando as funcionalidades de macros introduzidas no Swift 5.9 [22]. Essa abordagem permitiu criar uma extensão para o *Preview* do Xcode, onde os desenvolvedores podem ativar diferentes características de acessibilidade, como contraste de cores, tamanho de fonte e áreas de toque. A ferramenta foi projetada para ser facilmente integrável ao fluxo de desenvolvimento do Xcode, visando reduzir o atrito no processo de criação de interfaces acessíveis.

3.1.3 Testes Internos

Antes de ser avaliada por participantes externos, a ferramenta foi submetida a testes internos realizados durante o desenvolvimento. Esses testes permitiram identificar e corrigir falhas na implementação, além de refinar a interface da ferramenta para garantir sua usabilidade e eficácia.

3.2 Amostragem

Os participantes do estudo foram recrutados através de amostragem de conveniência, visando em pessoas desenvolvedoras com pelo menos um ano de experiência em SwiftUI,

para garantir que os participantes tivessem o conhecimento necessário para utilizar a biblioteca de forma eficaz. Foram realizados testes em projetos reais, onde os desenvolvedores utilizaram a biblioteca para identificar e corrigir problemas de acessibilidade em suas interfaces. No total, a amostra de participantes foi composta de 12 pessoas, abrangendo:

- **Experiência**: Pessoas desenvolvedoras com entre 1 a 10 anos de experiência em desenvolvimento de aplicativos móveis.
- **Familiaridade com Acessibilidade**: Participantes com diferentes níveis de conhecimento sobre práticas de acessibilidade, de iniciantes até especialistas.

3.3 Método de Análise de Dados

3.3.1 Questionário de Usabilidade (PSSUQ)

Após a utilização da ferramenta, os participantes preencheram o *Post-Study System Usability Questionnaire* (PSSUQ) [21], um questionário que avalia o sistema em termos de usabilidade do sistema (SYSUSE), qualidade da informação (INFOQUAL) e qualidade da interface (INTERQUAL). Foi escolhido o PSSUQ, ao invés de outras alternativas, como o System Usability Scale (SUS) [23], porque o PSSUQ permite uma análise mais aprofundada da usabilidade, pois avalia o sistema nestes três aspectos previamente citados, agregando para conclusões menos unilaterais e mais diversas. Já o SUS oferece uma avaliação mais geral, com uma única pontuação de usabilidade.

O questionário foi utilizado para permitir uma análise da percepção dos desenvolvedores sobre a ferramenta. A partir das respostas fornecidas, é possível obter pontuações para cada métrica do sistema e, consequentemente, obter conclusões.

4 RESULTADOS

Este capítulo apresenta e detalha os resultados obtidos a partir da implementação e avaliação da ferramenta proposta.

4.1 Escolha das Diretrizes de Acessibilidade

A escolha das diretrizes de acessibilidade que fundamentaram o desenvolvimento da ferramenta foi baseada na relevância e aplicabilidade das WCAG 2.2 e das Diretrizes de Interface Humana (HIG) da Apple. As WCAG 2.2 foram selecionadas por sua abrangência em estabelecer critérios claros e mensuráveis para garantir que interfaces digitais sejam acessíveis a todos os usuários. A inclusão de diretrizes específicas para contraste de cores e uso de cor, por exemplo, é interessante para atender às necessidades de usuários com deficiências visuais, como o daltonismo, que afeta uma parcela significativa da população mundial [5].

Além das WCAG 2.2, as Diretrizes de Interface Humana da Apple desempenharam um papel central na definição dos critérios adotados. Essas diretrizes oferecem recomendações específicas para o desenvolvimento de interfaces no ecossistema iOS, com ênfase na criação de experiências de usuário que sejam tanto esteticamente agradáveis quanto acessíveis. A escolha de diretrizes como o tamanho mínimo de alvos de toque (HIG) foi feita para garantir que a ferramenta ofereça suporte aos desenvolvedores na criação de interfaces que sejam utilizáveis por pessoas com deficiências motoras, garantindo que todos os elementos interativos sejam facilmente acessíveis [15].

A decisão final sobre quais diretrizes implementar também levou em consideração as limitações técnicas e as possibilidades oferecidas pelo SwiftUI e pelo Xcode. A capacidade de realizar simulações em tempo de desenvolvimento, como a aplicação de escala de cinza e a modificação de contrastes, foi priorizada para maximizar a eficácia da ferramenta dentro do fluxo de trabalho dos desenvolvedores. Essa abordagem não só facilita a verificação da conformidade com as diretrizes, mas também promove uma maior conscientização sobre a importância da acessibilidade desde as fases iniciais do design e desenvolvimento de aplicativos [24].

A seguir, as diretrizes escolhidas são detalhadas em subseções, abordando necessidades específicas de usuários com deficiências visuais, motoras e labirintopatias.

4.1.1 Critérios para Daltonismo

Usuários daltônicos frequentemente encontram dificuldades ao interagir com interfaces que dependem exclusivamente de cores para transmitir informações. O daltonismo afeta a capacidade de discernir entre determinadas tonalidades, o que pode comprometer a eficácia de uma interface visual. Para mitigar esses desafios, foram escolhidos os seguintes critérios:

- **Critério de Sucesso 1.4.1: Uso da Cor**

  De acordo com esta diretriz, a cor não deve ser a única forma de transmitir informações, sinalizar uma reação, indicar uma ação ou destacar um elemento visual. Essa recomendação é fundamental para garantir que usuários daltônicos possam acessar e compreender informações sem depender da diferenciação de cores [5].

- **Critério de Sucesso 1.4.3: Contraste (Mínimo)**

  Esta diretriz exige que haja contraste suficiente entre o texto e o plano de fundo para que pessoas com daltonismo ou outras deficiências visuais possam ler o texto confortavelmente. A ferramenta implementa funcionalidades que simulam condições de baixo contraste, permitindo aos desenvolvedores ajustar a paleta de cores da interface para atender a esses requisitos [5].

4.1.2 Critério para Deficiências Motoras

As interfaces digitais devem ser projetadas para suportar a interação de usuários com deficiências motoras, como tremores ou dificuldades em realizar movimentos precisos. Esses usuários enfrentam desafios específicos ao tentar interagir com elementos de interface que são pequenos ou que requerem alta precisão. Para atender a essas necessidades, o critério selecionado foi:

- **Critério de Sucesso 2.5.5: Tamanho do Alvo (Nível AAA)**

  Esta diretriz recomenda que os alvos de toque, como botões e links, sejam suficientemente grandes para que possam ser utilizados com facilidade por pessoas com mobilidade limitada. A ferramenta oferece funcionalidades que destacam os elementos da interface que não atendem a esse requisito, permitindo ajustes em tempo de desenvolvimento para garantir a acessibilidade [5].

4.1.3 Critérios para Labirintopatias

Usuários com labirintopatias, ou distúrbios vestibulares, são particularmente sensíveis a movimentos excessivos ou inesperados em interfaces digitais, o que pode provocar náusea, vertigem e desorientação. Para minimizar esses efeitos negativos e tornar as interfaces mais confortáveis para esses usuários, as seguintes diretrizes foram selecionadas:

- **Critério de Sucesso 2.3.3: Animação Causada por Interações**

  Esta recomendação sugere que os usuários com problemas vestibulares devem ser capazes de evitar ou minimizar as animações causadas pelas interações do usuário, reduzindo o risco de desconforto. A ferramenta desenvolvida permite que os desenvolvedores simulem diferentes velocidades de animação e ajustem a interface de acordo [5].

- **Critério de Sucesso 2.2.2: Pausar, Parar, Ocultar**

  Esta diretriz permite que os usuários pausem, parem ou ocultem qualquer informação em movimento para criar uma experiência de usuário mais gerenciável e agradável. A ferramenta inclui a opção de congelar todas as animações da interface, garantindo que as funcionalidades essenciais permaneçam acessíveis mesmo quando as animações são desativadas [5].

4.2 Detalhamento da Ferramenta

A ferramenta desenvolvida, nomeada ***AccessiblePreview***, foi projetada para ser uma solução integrada diretamente no ambiente de desenvolvimento Xcode, utilizando o SwiftUI como base. Essa integração permite que os desenvolvedores realizem verificações e ajustes de acessibilidade em suas interfaces de forma dinâmica. A escolha por uma integração direta no Xcode foi motivada pela necessidade de minimizar a dependência de ferramentas externas, o que garante que as diretrizes de acessibilidade sejam incorporadas desde as primeiras fases do processo de design e desenvolvimento. Isso não apenas simplifica o workflow dos desenvolvedores, mas também promove uma cultura de acessibilidade contínua durante todo o ciclo de vida do desenvolvimento do software.

Através de uma macro [22] que pode ser utilizada diretamente no SwiftUI, a ferramenta permite que os desenvolvedores visualizem e ajustem as interfaces de acordo com as necessidades de acessibilidade, assegurando que os produtos finais sejam inclusivos e usáveis por todos, utilizando apenas as Pré-Visualizações do Xcode [9].

Cada *trait*, ou característica da ferramenta, foi cuidadosamente projetada para abordar necessidades específicas de acessibilidade, conforme as diretrizes WCAG 2.2 e as Diretrizes de Interface Humana (HIG) da Apple. No contexto desta ferramenta, uma *trait* refere-se a uma funcionalidade específica que pode ser ativada durante o desenvolvimento para simular como a interface se comportaria em condições específicas de acessibilidade. Essas *traits* permitem que os desenvolvedores visualizem e modifiquem a interface de maneira a atender melhor as necessidades de usuários com diferentes tipos de deficiências, como daltonismo, limitações motoras e labirintopatias.

Além disso, a ferramenta foi desenvolvida com um foco especial na usabilidade. As *traits* são intuitivas de utilizar, ativadas diretamente no código SwiftUI, o que facilita a adoção por desenvolvedores de todos os níveis de experiência. Ao integrar essas funcionalidades diretamente no ambiente de desenvolvimento, a ferramenta oferece uma maneira eficiente de garantir que as interfaces sejam acessíveis, melhorando a experiência do usuário final. Essas *traits* incluem opções para testar contraste, modificar áreas de toque, ajustar animações, entre outras, todas visando garantir a conformidade com os padrões de acessibilidade.

A seguir, são detalhadas as *traits* principais da ferramenta, organizadas de acordo com os critérios de acessibilidade abordados.

4.2.1 Funcionalidades para Daltonismo

Para abordar os critérios de sucesso de daltonismo definidos na seção 4.1.1, a ferramenta inclui duas *traits* principais:

- ***.grayscale***

  Esta funcionalidade permite que os desenvolvedores visualizem suas interfaces em uma escala de cinza, removendo a diferenciação de cores. Isso assegura que as informações transmitidas por meio de cor sejam complementadas por outros elementos visuais, como ícones ou texto. Essa funcionalidade garante a conformidade com o Critério de Sucesso 1.4.1 (Uso da Cor) [5].

- ***.lowContrast***

  Através desta *trait*, os desenvolvedores podem simular condições de baixo contraste na interface, replicando a experiência de usuários com deficiências visuais que dificultam a distinção entre cores de baixa saturação. Essa funcionalidade auxilia na adequação ao Critério de Sucesso 1.4.3 (Contraste Mínimo), permitindo ajustes precisos para melhorar a legibilidade do texto e outros elementos visuais [5].

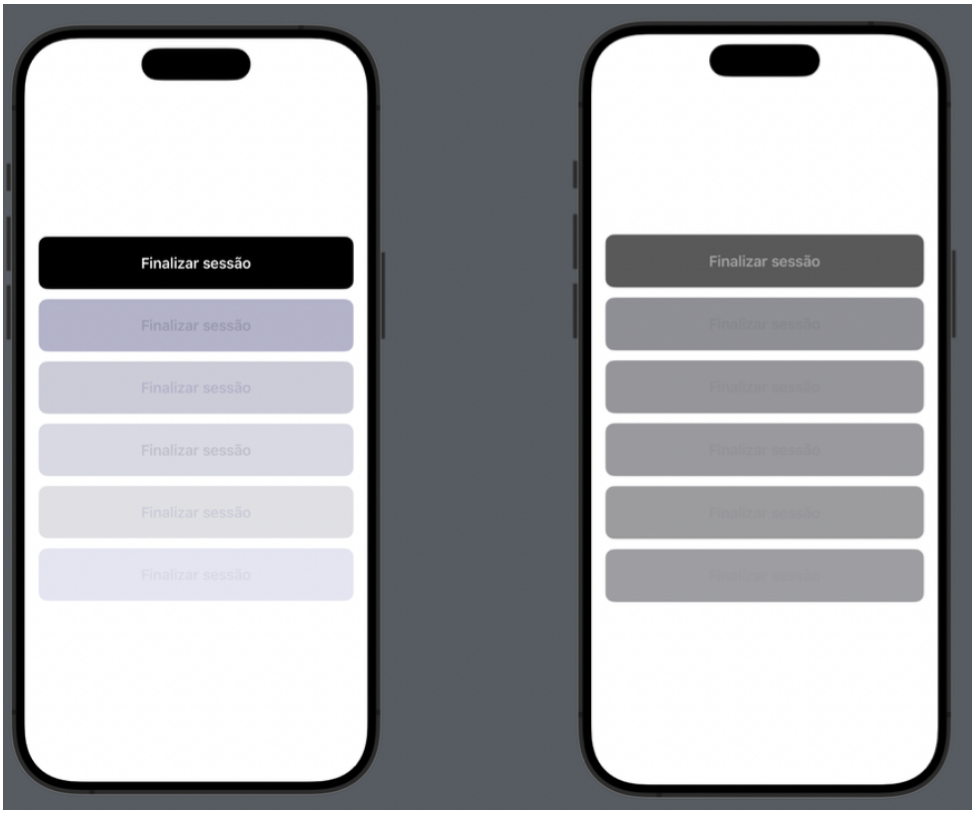


Figura 2: Exemplo da execução do *trait .lowContrast*.

A Figura 2 mostra duas telas iguais com botões, a primeira sem aplicar contraste baixo e a segunda aplicando contraste baixo. Dessa forma os desenvolvedores conseguirão observar tais diferenças mais nitidamente se não identificadas anteriormente.

4.2.2 Funcionalidades para Deficiências Motoras

A ferramenta também inclui uma *trait* para atender às necessidades de usuários com deficiências motoras, conforme definido na seção 4.1.2:

- ***.buttonTapArea***

Esta funcionalidade destaca os botões e outros alvos de toque na interface, mostrando contornos coloridos que indicam se os elementos atendem ao tamanho mínimo recomendado (44x44 pixels). Elementos que cumprem esse requisito são marcados com um contorno verde, enquanto aqueles que não o cumprem são marcados com um contorno vermelho. Isso permite que os desenvolvedores identifiquem rapidamente os elementos que precisam de ajustes para atender ao Critério de Sucesso 2.5.5 (Tamanho do Alvo) [5].

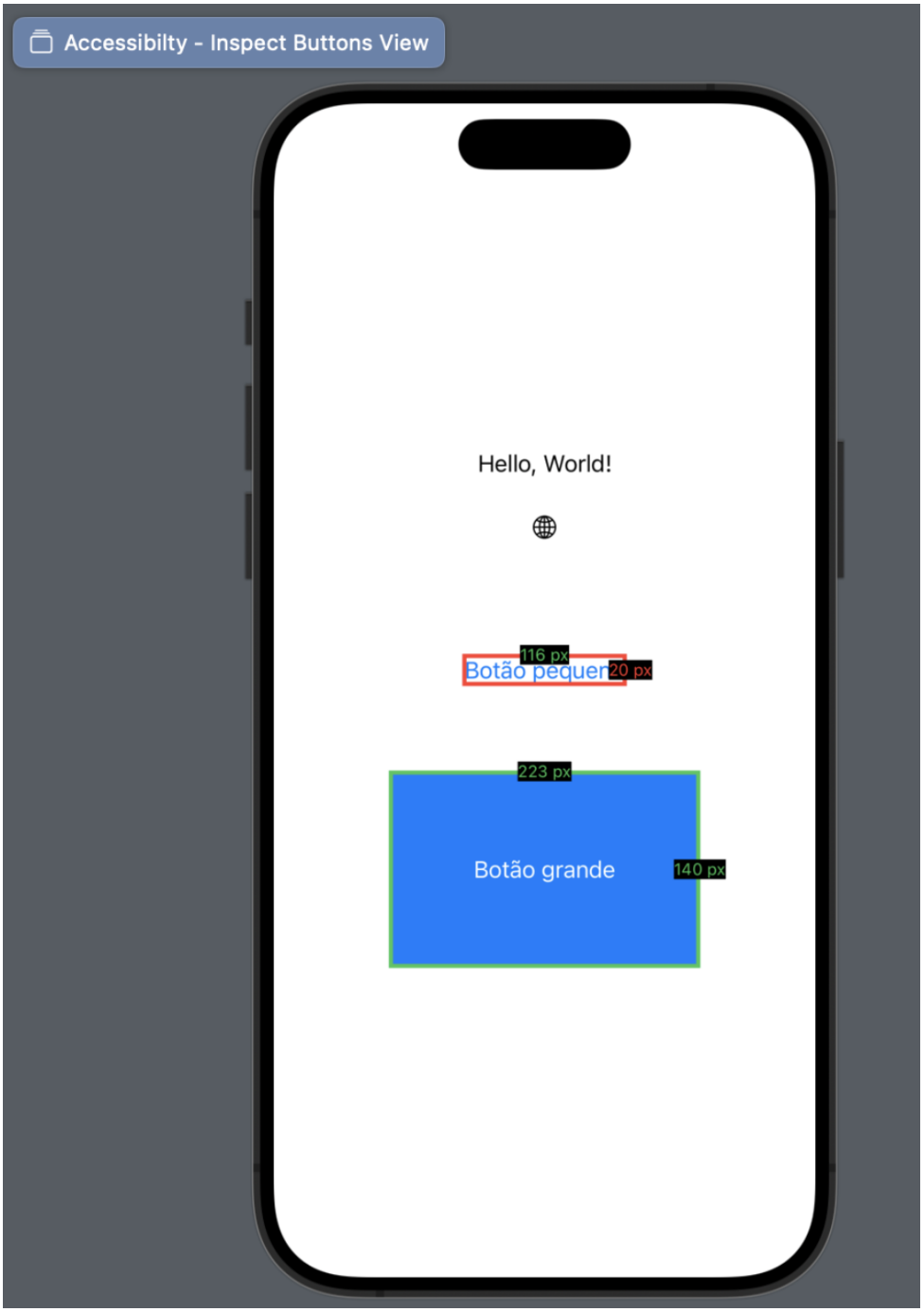


Figura 3: Exemplo da execução do *trait .buttonTapArea*.

4.2.3 Funcionalidades para Labirintopatias

Para usuários com problemas vestibulares, a ferramenta oferece *traits* que permitem ajustar ou eliminar animações na interface, em conformidade com os critérios discutidos na seção 4.1.3:

- ***.lowAnimation***

  Esta *trait* permite que os desenvolvedores simulem a interface com animações desaceleradas, verificando se a funcionalidade e a usabilidade permanecem adequadas mesmo com a redução de movimentos. Isso ajuda a assegurar a conformidade com o Critério de Sucesso 2.3.3 (Animação Causada por Interações), garantindo que as animações não causem desconforto a usuários sensíveis [5].

- ***.noAnimation***

  Esta funcionalidade congela todas as animações da interface, permitindo que os desenvolvedores vejam como a interface funciona sem qualquer movimento. Isso é essencial para verificar se a interface permanece funcional e acessível quando as animações são desativadas, em conformidade com o Critério de Sucesso 2.2.2 (Pausar, Parar, Ocultar) [5].

4.2.4 Aplicação de *traits*

A execução de pré-visualizações possui o mesmo processo de utilizar as Previews [9] padrões da plataforma Xcode, a fim de não introduzir uma visualização fora do habitual para os desenvolvedores. Conforme a Figura 4, é possível notar que a escrita e a aplicação da ferramenta se assemelha ao padrão definido pelo SwiftUI [12] presente no primeiro item da figura, em comparação com os outros casos, que são variantes de aplicações de *traits*.

```
// Default Preview
#Preview {
    ContentView()
}

// Example 1: Preview with slower animations
#Preview(trait: .lowAnimation) {
    ContentView()
}

// Example 2: High contrast with extra-large dynamic text
#Preview(traits: .highContrast, .extraLargeText) {
    ContentView()
}

// Example 3: Preview with detailed button tap area,
    showing if it respects minimal 44x44px limit
#Preview(trait: .buttonTapArea, name: "Accessibility") {
    ContentView()
}
```

Figura 4: Aplicação da ferramenta no código.

4.3 Análise dos Dados Coletados

4.3.1 Detalhamento dos Participantes

Os participantes foram divididos em grupos com base em dois critérios principais: **experiência em desenvolvimento mobile** e **conhecimento em acessibilidade**. A tabela a seguir apresenta o agrupamento de acordo com esses dois critérios:

Tabela 1: Perfil dos participantes agrupados por experiência e conhecimento de acessibilidade

Experiência em Desenvolvimento Mobile	
Experiência	**n(%)**
0-2 anos	3 (25%)
2-3 anos	2 (17%)
3-5 anos	3 (25%)
5+ anos	4 (33%)
Conhecimento em Acessibilidade	
Nível de Conhecimento	**n(%)**
Iniciante	5 (42%)
Intermediário	4 (33%)
Avançado	3 (25%)

Notou-se, a partir desse detalhamento, que era possível separar os participantes em três grupos:

- **Grupo 1** - Estes participantes têm uma abrangente experiência no desenvolvimento iOS e, em sua maioria, possuem um nível avançado de conhecimento sobre acessibilidade, tendo trabalhado diretamente com práticas e projetos relacionados ao tema.
- **Grupo 2** – Os participantes deste grupo têm um nível intermediário a baixo de conhecimento de acessibilidade, com alguns tendo experiência prática limitada, enquanto outros possuem maior familiaridade teórica e experiência prática moderada.
- **Grupo 3** – Este grupo inclui desenvolvedores mais novos, com pouco ou nenhum conhecimento prático de acessibilidade, ainda em fase de aprendizado sobre o tema.

4.3.2 Análise do PSSUQ

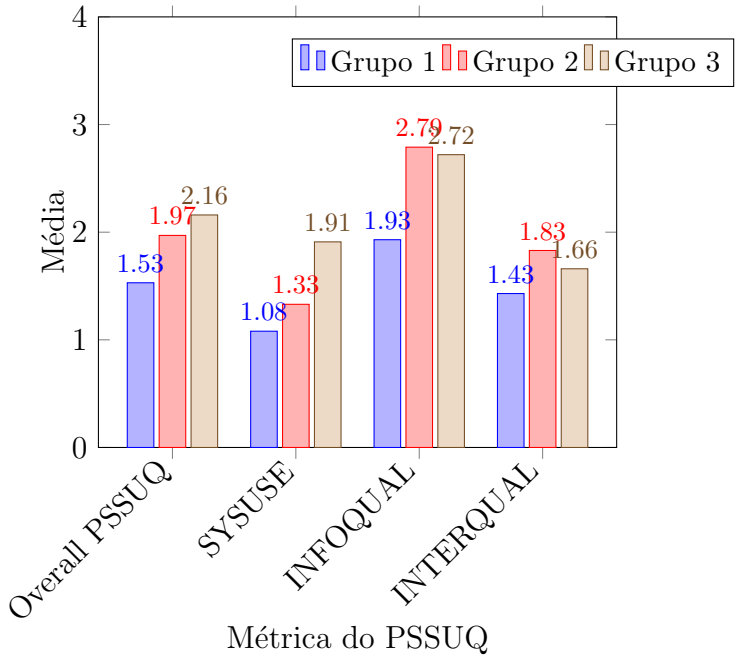
Os resultados do Post-Study System Usability Questionnaire (PSSUQ) foram agrupados em base dos grupos previamentes definidos. A pontuação 1.0 é a melhor pontuação possível e a pontuação 7.0 é a pontuação menos receptiva possível.

As métricas fornecidas pelo método e, consequentemente, analizadas, incluem a *Pontuação Geral do PSSUQ*, *System Usefulness* (SYSUSE), *Information Quality* (INFOQUAL) e *Interface Quality* (INTERQUAL).

Tabela 2: Média dos Resultados do PSSUQ por Grupo

Grupo	**Overall PSSUQ**	**SYSUSE**	**INFOQUAL**	**INTERQUAL**
Grupo 1	1.53	1.08	1.93	1.43
Grupo 2	1.97	1.33	2.79	1.83
Grupo 3	2.16	1.91	2.72	1.66

Figura 5: Gráfico da Média dos Resultados do PSSUQ por Grupo



4.3.2.1 Pontuação Geral (Overall PSSUQ)

O PSSUQ Score reflete a satisfação geral dos participantes com o sistema como um todo. Neste caso, os resultados indicaram que o **Grupo 1**, que compreendia desenvolvedores mais experientes, obteve a média mais baixa, 1.53, o que pode sugerir que os desenvolvedores mais experientes estavam mais satisfeitos com a ferramenta. Isso pode ser explicado pelo fato de que esses participantes, por terem mais experiência com Acessibilidade, viram maior importância na ferramenta e, portanto, sua avaliação foi mais mais positiva.

Por outro lado, o **Grupo 3**, que consistia em desenvolvedores menos experientes, obteve a média mais alta, 2.16, indicando uma percepção ainda positiva, mas menos favorável do sistema. Possivelmente, esses participantes tiveram dificuldade adicional ao usar a ferramenta devido à falta de experiência no desenvolvimento de aplicativos ou não se é preocupado com a necessidade de aplicar acessibilidade.

O **Grupo 2** teve uma média intermediária (1.97), o que reflete uma visão mais equilibrada do sistema.

4.3.2.2 Análise do *System Usefulness* (SYSUSE)

A escala da *System Usefulness* (SYSUSE) reflete quanto os participantes acharam a ferramenta útil para o desenvolvimento de aplicações. A média mais baixa entre os grupos foi alcançada pelo **Grupo 1** (1.08), sugerindo que os desenvolvedores mais experientes nesta área estavam mais satisfeitos com a ferramenta. Basicamente, a ferramenta pode ter apoiado eficazmente a aplicação de acessibilidade. Os desenvolvedores mais inexperientes, no **Grupo 3** acharam o uso da ferramenta menos valioso, mas ainda importante, para suas atividades (1.91).

4.3.2.3 Análise do *Information Quality* (INFOQUAL)

A subescala *Information Quality* (INFOQUAL) avalia a qualidade das informações fornecidas pela ferramenta, como clareza, precisão e completude. O **Grupo 2** atribuiu a maior média (2.79) para esta subescala, próximo da nota do **Grupo 3**, sugerindo que a clareza das informações não é ruim mas poderia ser melhorada para atender às necessidades de desenvolvedores menos experientes, que provavelmente precisam de mais

suporte para adaptar suas práticas às exigências de acessibilidade.

O **Grupo 1**, com uma média de 1.93, demonstrou estar relativamente satisfeito com a qualidade das informações fornecidas, refletindo que apresentação de informações é adequada para desenvolvedores mais experientes, que provavelmente requerem menos orientação detalhada.

Uma razão para o valor mais alto que o esperado se mostra ao analizar o item 7 do PSSUQ: *O sistema apresentou mensagens de erros claras que me ajudassem a resolver o problema*. Esse item obteve uma média geral de 3.625 pontos, demonstrando a alta necessidade que a ferramenta precisava para ajudar em momentos de erro de uso ou de instalação.

4.3.2.4 Análise do *Interface Quality* (INTERQUAL)

A subescala *Interface Quality* (INTERQUAL) avalia a qualidade da interface do sistema, incluindo aspectos como facilidade de uso, organização dos elementos e responsividade. As médias de todos os grupos foram bastante satisfatórias para o esperado, principalmente porque a ferramenta é integrada ao Xcode e é totalmente adaptada às pré-visualizações já existentes [9].

5 DISCUSSÃO DOS RESULTADOS

A solução atingiu, em geral, os objetivos propostos, de acordo com a análise dos resultados obtidos pela ferramenta desenvolvida para facilitar a implementação e visualização de acessibilidade em aplicativos móveis. O impacto no fluxo de trabalho, a percepção dos desenvolvedores sobre a usabilidade da ferramenta e a eficácia em atender aos critérios de acessibilidade são os principais tópicos de discussão.

De acordo com os dados do questionário PSSUQ [21], os desenvolvedores com mais experiência em construir interfaces fáceis de usar foram os mais beneficiados pela ferramenta. Os participantes apreciaram a integração direta com o textit Xcode e os recursos oferecidos. A baixa pontuação de usabilidade da ferramenta (SYSUSE) entre os desenvolvedores mais experientes indica que ela ajudou significativamente na detecção e correção de falhas de acessibilidade.

No entanto, os desenvolvedores menos experientes apresentaram maiores dificuldades em utilizar a ferramenta, conforme evidenciado pelas notas mais elevadas no quesito *Information Quality* (INFOQUAL). Esse fator pode estar relacionado à falta de mensagens de erro claras apresentadas pela ferramenta, sugerindo que há espaço para melhorias na clareza e acessibilidade das informações fornecidas. Essa lacuna reflete a necessidade de facilitar o uso da ferramenta para desenvolvedores com menos experiência em acessibilidade [10].

A redução do tempo necessário para o desenvolvimento também se mostra importante, principalmente para os desenvolvedores com conhecimento intermediário e avançado. A ferramenta eliminou a necessidade de alterar manualmente as configurações exibindo várias combinações de acessibilidade em tempo de desenvolvimento. Essa característica está de acordo com as descobertas de Oliveira e Filgueiras [11], que indicam a falta de suporte em ferramentas tradicionais para simular vários cenários de acessibilidade simultaneamente. No entanto, os desenvolvedores iniciantes disseram que a curva de aprendizado foi um problema no início, o que prejudicou a percepção de utilidade.

No que diz respeito à conformidade com as diretrizes de acessibilidade, a ferramenta mostrou-se eficaz em simular alguns cenários de daltonismo, deficiências motoras e labirintopatias. Por exemplo, a funcionalidade *.buttonTapArea*, que destaca os alvos de toque que não atendem ao tamanho mínimo recomendado, foi necessário para atender ao

critério 2.5.5 (Tamanho do Alvo) [5].

Os resultados indicam que a ferramenta alcançou sucesso ao reduzir o atrito no desenvolvimento de interfaces acessíveis, promovendo maior aderência às diretrizes de acessibilidade e facilitando o trabalho dos desenvolvedores. Contudo, ajustes em sua usabilidade e uma maior atenção às necessidades de desenvolvedores menos experientes podem aprimorar ainda mais a eficácia da solução, conforme sugerido por estudos recentes sobre acessibilidade e desenvolvimento de software [10] [13].

6 CONCLUSÃO

A criação de aplicativos acessíveis é um desafio pois é necessário prestar atenção às diversas necessidades dos usuários. Este trabalho buscou reduzir o atrito encontrado no processo de implementar acessibilidade em aplicativos, propondo uma ferramenta que facilita a visualização e a aplicação de componentes acessíveis durante o desenvolvimento, conformando com as diretrizes WCAG 2.2 [5].

Os resultados obtidos pela aplicação da ferramenta indicam que o seu uso permitiu maior eficiência no desenvolvimento de interfaces acessíveis no *Xcode*. Simular cenários específicos de acessibilidade rapidamente foi eficaz na melhoria da experiência dos desenvolvedores., reduzindo a margem de erro e promovendo uma cultura de acessibilidade contínua [24].

Entretanto, a análise apresentou algumas limitações que merecem atenção futura. A principal dificuldade relatada por desenvolvedores iniciantes foi a curva de aprendizado da ferramenta, principalmente na compreensão das mensagens de erro e feedbacks fornecidos. Essa questão destaca a necessidade de alguns refinamentos na interface da ferramenta e na clareza das informações apresentadas [10].

Por fim, os resultados obtidos mostram que a ferramenta proposta tem potencial para facilitar a implementação de acessibilidade em aplicativos *iOS*. A ferramenta pode se tornar uma peça importante no fluxo de trabalho de desenvolvedores e um estímulo para a adoção de práticas acessíveis desde as primeiras fases de desenvolvimento.

Este estudo, dessa forma, contribui para o campo da acessibilidade digital, oferecendo uma solução prática que responde às lacunas identificadas em pesquisas anteriores [11] [16]. Ao facilitar o desenvolvimento de aplicações acessíveis e melhorar a experiência dos desenvolvedores, a ferramenta aproxima o mercado de software de uma prática mais inclusiva e responsável, refletindo o compromisso com a acessibilidade digital.

Trabalhos Futuros

A partir dos resultados e das limitações identificadas, sugerem-se os seguintes trabalhos futuros:

- Analizar novos critérios de sucesso que podem ser aplicados e validados na ferramenta;

- Ampliar o estudo para avaliar o impacto da ferramenta em outros contextos além do desenvolvimento iOS, como o desenvolvimento de aplicativos Android;
- Realizar estudos longitudinais para avaliar o impacto da ferramenta na adoção de práticas acessíveis ao longo do tempo.

Bibliografia

[1] ORGANIZAÇÃO MUNDIAL DA SAÚDE. *Relatório mundial sobre deficiência*. 2011.

[2] STATISTA. *Taxa de adoção de smartphones em todo o mundo*. 2021. Disponível em: `https://www.statista.com/statistics/1258906/worldwide-smartphone-adoption-rate-telecommunication-by-region/`. Acesso em: 22/07/2024.

[3] ARANGO, J. *Vivendo na Informação - Design Responsável para Lugares Digitais*. 2018.

[4] WebAIM. *The WebAIM Million: An Accessibility Analysis of the Top One Million Web Sites*. 2021. Disponível em: `https://webaim.org/projects/million/`. Acesso em: 23/07/2024.

[5] WORLD WIDE WEB CONSORTIUM (W3C). *Web Content Accessibility Guidelines (WCAG) 2.2*. 2021.

[6] APPLE INC. *Accessibility*. Disponível em: `https://developer.apple.com/documentation/accessibility`. Acesso em: 22/07/2024.

[7] YAN, S.; RAMACHANDRAN, P.G. *ACM Transactions on Accessible Computing (TACCESS)*, v.12, 2020.

[8] NA, D. *Pushing Through Friction*. Disponível em: `https://blog.danielna.com/talks/pushing-through-friction`. Acesso em: 23/07/2024.

[9] APPLE INC. *Previews in Xcode*. Disponível em: `https://developer.apple.com/documentation/swiftui/previews-in-xcode`. Acesso em: 24/07/2024.

[10] LEITE, Manoel Victor Rodrigues et al. *Acessibilidade na indústria de desenvolvimento móvel no Brasil: Conscientização, conhecimento, adoção, motivações e barreiras*. Revista de Sistemas e Software, 2021.

[11] OLIVEIRA, A. F. B. A.; FILGUEIRAS, L. V. L. *Ferramentas de assistência ao desenvolvedor para a criação de aplicativos móveis nativos acessíveis a pessoas com deficiência visual: Uma Revisão Sistemática*. 2018.

[12] APPLE INC. *SwiftUI*. Disponível em: `https://developer.apple.com/documentation/swiftui`. Acesso em: 23/07/2024.

[13] WHITAKER, R. *Developing Inclusive Mobile Apps*.

[14] GOVERNO DIGITAL. *Acessibilidade digital*. Disponível em: `https://www.gov.br/governodigital/pt-br/acessibilidade-e-usuario/acessibilidade-digital`. Acesso em: 23/07/2024.

[15] APPLE INC. *Diretrizes de interface humana - Acessibilidade*. Disponível em: `https://developer.apple.com/design/human-interface-guidelines/accessibility/overview/introduction/`. Acesso em: 23/07/2024.

[16] BOTELHO, F. H. F. *Acessibilidade à tecnologia digital: Barreiras virtuais, oportunidades reais*. Tecnologia Assistiva, v.33, 2021.

[17] Sofia Diniz and Kiev Gama. 2024. *Design Systems and Component Packages as an Interface for Accessibility*. In Anais do XXXVIII Simpósio Brasileiro de Engenharia de Software, setembro 30, 2024, Curitiba/PR, Brasil. SBC, Porto Alegre, Brasil, 592-598. DOI: https://doi.org/10.5753/sbes.2024.3566.

[18] SILVA, H. dos S. *CodeForAll: Componentes modulares para criação de aplicações acessíveis em SwiftUI*. 2023.

[19] WORLD WIDE WEB CONSORTIUM (W3C). *Accessible Rich Internet Applications (WAI-ARIA) 1.2*. 2022. Disponível em: ¡https://www.w3.org/TR/wai-aria-1.2/¿. Acesso em: 11/10/2024.

[20] PATIL, N.; BHOLE, D.; SHETE, P. *Enhanced UI Automator Viewer with improved Android accessibility evaluation features.* International Conference on Automatic Control and Dynamic Optimization Techniques ICACDOT, 2021.

[21] Lewis, James R. *Psychometric evaluation of the post-study system usability questionnaire: The PSSUQ*. Proceedings of the human factors society annual meeting. Vol. 36. No. 16. Sage CA: Los Angeles, CA: Sage Publications, 1992.

[22] APPLE INC. *Macros*. Disponível em: `https://docs.swift.org/swift-book/documentation/the-swift-programming-language/macros/`. Acesso em: 23/07/2024.

[23] BROOKE, John. SUS: a quick and dirty usability scale. In: JORDAN, P. W.; THOMAS, B.; WEERDMEESTER, B. A.; McCLELLAND, I. L. (Eds.). Usability evaluation in industry. London: Taylor and Francis, 1996. p. 189-194.

[24] LAZAR, J.; GOLDSTEIN, D. F.; TAYLOR, A. *Ensuring Digital Accessibility through Process and Policy*. Morgan Kaufmann, 2015.

[25] APPLE INC. *Accessibility Inspector*. Disponível em: `https://developer.apple.com/documentation/accessibility/accessibility-inspector`. Acesso em: 23/07/2024.

[26] APPLE INC. *VoiceOver Tests*. Disponível em: `https://developer.apple.com/videos/play/wwdc2019/252/`. Acesso em: 26/07/2024.

[27] PALANI, N. *The Web Accessibility Project*. 2022.