Revista Eletrônica de Iniciação Científica em Computação, 2026, 24:1
ISSN: 3085-8461 • doi: 10.5753/reic.2026.6969

Artigo de Pesquisa/Research Paper

Ambientes de Desenvolvimento Integrado de Programação: Uma análise comparativa baseada nas diretrizes WCAG 2.2, WAI-ARIA 1.3, UAAG 2.2 e ATAG 2.0

Integrated Programming Development Environments: A comparative analysis based on WCAG 2.2, WAI-ARIA 1.3, UAAG 2.2, ATAG 2.0 guidelines

Luanna Bahia da Silva [Universidade do Estado do Mato Grosso | *luanna.bahia@unemat.br*]
Alexandre Berndt [Universidade do Estado do Mato Grosso | *alexandreberndt@unemat.br*]

Bacharelado em Ciência da Computação, Universidade do Estado de Mato Grosso Campus Universitário de Barra do Bugres Deputado Renê Barbour, Rua A, 130, Bairro São Raimundo, Barra do Bugres, MT, 78390-000, Brasil.

Resumo. Este artigo propôs uma análise comparativa de Ambientes de Desenvolvimento Integrado (IDEs) com recursos de acessibilidade adaptativa, visando a inclusão de pessoas com deficiência visual em ambientes de programação. O estudo buscou investigar o nível de acessibilidade oferecido pelas IDEs atuais e as razões pelas quais ainda não existe uma IDE Web com recursos de acessibilidade integrados. A hipótese central foi que, apesar dos avanços tecnológicos entre 2020 e 2025, as ferramentas modernas ainda apresentam barreiras significativas, principalmente devido à falta de integração eficiente com tecnologias assistivas. A metodologia adotada combinou pesquisas exploratória e comparativa, avaliando as IDEs selecionadas de acordo com as diretrizes da W3C (WCAG, WAI-ARIA, UAAG e ATAG). A avaliação foi realizada com base nos princípios de usabilidade, acessibilidade e comunicabilidade, utilizando testes práticos com leitor de tela (NVDA), um protótipo chamado Editor Visual Inclusivo (EVICODE), Visual Studio Code for the Web e Online-IDE. Os resultados contribuem para o desenvolvimento de ferramentas mais inclusivas e fortalecem o compromisso com a democratização do acesso à tecnologia.

Abstract. This article presented a comparative analysis of Integrated Development Environments (IDEs) with adaptive accessibility features, involving the inclusion of visually impaired people in programming environments. The study sought to investigate the level of accessibility offered by current IDEs and the reasons why a Web IDE with integrated accessibility features does not yet exist. The central hypothesis is that, despite technological advances between 2020 and 2025, modern tools still present significant barriers, mainly due to the lack of efficient technology with assistive technologies. The methodology adopted combined exploratory and comparative research, evaluating the selected IDEs according to the W3C guidelines (WCAG, WAI-ARIA, UAAG and ATAG). The evaluation was carried out based on the principles of usability, accessibility and communicability, using practical tests with a screen reader (NVDA), a prototype called Inclusive Visual Editor (EVICODE), Visual Studio for the Web and Online-IDE. The positive results for the development of more inclusive tools strengthen the commitment to democratizing access to technology.

Palavras-chave: Ambientes de Desenvolvimento Integrado (IDEs), Acessibilidade, Deficiência Visual, Tecnologias Assistivas, Análise Comparativa, W3C, WCAG, WAI-ARIA, UAAG, ATAG, Leitor de Tela

Keywords: Integrated Development Environments (IDEs), Accessibility, Visual Impairment, Assistive Technologies, Comparative Analysis, W3C, WCAG, WAI-ARIA, UAAG, ATAG, Screen Reader

Recebido/Received: 23 November 2025 • **Aceito/Accepted:** 20 March 2026 • **Publicado/Published:** 02 April 2026

1 Introdução

A inserção de pessoas com deficiência visual no mercado de trabalho e em espaços educacionais enfrentam obstáculos, sobretudo nas áreas voltadas à tecnologia e ao desenvolvimento de software Petrausch and Loitsch [2017]; Lopes [2023]. Apesar dos marcos legais, como a Lei Brasileira de Inclusão (Lei nº 13.146/2015) Brasil [2015], ainda existe uma escassez de ferramentas de programação eficazes e acessíveis.

A lacuna entre a necessidade de inclusão e as ferramentas disponíveis foi o ponto de partida deste trabalho. Um levantamento exploratório inicial (detalhado na Seção 4.3) confirmou a dificuldade em encontrar Ambientes de Desenvolvimento Integrado (IDEs) baseados na web que se declarassem acessíveis.

Portanto, este artigo teve como objetivo principal realizar uma análise comparativa da acessibilidade de IDEs Web populares no período de 2020 a 2025. O estudo buscou responder quais barreiras de acessibilidade existem nessas ferramentas e por que a integração com tecnologias assistivas ainda é deficiente.

Para atingir esse objetivo, foi adotada uma metodologia de estudo de casos múltiplos Yin [2001], analisando três perfis de IDEs: `vscode.dev` (representando alta complexidade e popularidade de mercado), `Online-IDE.com` (representando o minimalismo) e um protótipo chamado `EVICODE`, desenvolvido no estágio supervisionado[1] com foco em acessibilidade.

[1] O estágio supervisionado refere-se às atividades desenvolvidas na Unemat de Barra do Bugres - MT, entre os semestres letivos de 2024/1 e 2025/1, onde a autora atuou com um grupo de estudantes do Curso de Ciência da Computação no desenvolvimento de uma IDE Web com recursos de acessibilidade integrados.

A avaliação foi fundamentada nas diretrizes W3C (WCAG, WAI-ARIA, UAAG e ATAG) e nos princípios de Acessibilidade, Usabilidade e Comunicabilidade [Sonza *et al*., 2013, p. 316], aplicados através de um conjunto de tarefas práticas (Seção 4.3).

A análise revelou um padrão de falha crítica em todas as ferramentas, incluindo o EVICODE. Barreiras severas, como a ausência de feedback sonoro para erros de sintaxe (uma violação da diretriz ATAG 2.0) e a inacessibilidade dos terminais de execução de código, foram identificadas, demonstrando que o foco das ferramentas atuais tem sido dado à acessibilidade da interface, mas não à acessibilidade do fluxo de trabalho de programação.

Este artigo está estruturado da seguinte forma: A Seção 2 apresenta a fundamentação teórica sobre acessibilidade digital, legislação e as diretrizes W3C. A Seção 3 revisa os trabalhos relacionados. A Seção 4 detalha a metodologia de pesquisa, incluindo os critérios de avaliação e as tarefas de teste. A Seção 5 apresenta os resultados e a análise comparativa de cada IDE. Por fim, a Seção 6 apresenta a conclusão, as limitações do estudo e sugestões para trabalhos futuros.

2 Fundamentação teórica

Esta pesquisa se fundamenta em um levantamento bibliográfico que abrange as áreas de Acessibilidade Digital, Tecnologia Assistiva, Engenharia de Software e Interação Humano-Computador (IHC). A revisão da literatura explorou os conceitos de acessibilidade digital e sua evolução legislativa no Brasil, com foco em marcos como a Lei nº 13.146/2015 Brasil [2015] e o Decreto nº 5.296/2004 Brasil [2004]. A análise se aprofunda nas diretrizes e recomendações de acessibilidade estabelecidas pelo W3C [2025a], incluindo as especificações do WCAG [2025], WAI-ARIA [2017], UAAG [2015] e ATAG [2015], que servem como referencial técnico para a avaliação das ferramentas. Além disso, o trabalho examina o conceito de Desenho Universal de Carletto and Cambiaghi [2007], a importância de tecnologias assistivas como os leitores de tela (ex: NVDA) e a estrutura dos Ambientes de Desenvolvimento Integrado (IDEs), proporcionando a base teórica para a análise comparativa de acessibilidade.

2.1 Acessibilidade digital

De acordo com o World Wide Web Consortium – W3C [2025a],

> A web é fundamentalmente projetada para funcionar para todas as pessoas, independentemente de seu hardware, software, idioma, localização ou capacidade. Quando a web atinge esse objetivo, ela é acessível a pessoas com uma gama variada de audição, movimento, visão e capacidade cognitiva.

Essa citação conceitua a acessibilidade digital, que vai além de aspectos técnicos e se liga à inclusão de Pessoas com Deficiência (PcD), dando oportunidades igualitárias a todos, permitindo utilizar ou acessar recursos e funcionalidades em um ambiente de desenvolvimento, na web ou em softwares. A acessibilidade digital consiste, portanto, no desenvolvimento de sites, ferramentas e aplicações que possam ser utilizadas de forma plena por todas as pessoas, inclusive aquelas com deficiência. Isso envolve garantir que os conteúdos sejam perceptíveis, operáveis, compreensíveis e robustos, princípios fundamentais das Diretrizes de Acessibilidade para Conteúdo Web (WCAG), elaboradas pelo próprio W3C.

Ao desenvolver soluções digitais acessíveis, é fundamental reconhecer que a web deve atender usuários com variados tipos de deficiência, incluindo aquelas de ordem auditiva, visual, física, cognitiva, da fala e neurológica. W3C [2025b] Ou seja, os designs de websites e IDEs antes de serem desenvolvidos precisam considerar uma variedade de situações, permitindo o acesso a todas as pessoas, com navegação fluída e autônoma.

Contudo, os benefícios da acessibilidade digital não se restringem às pessoas com deficiência permanente. A acessibilidade também favorece usuários sem deficiência, como aqueles que utilizam dispositivos com telas pequenas (como celulares), pessoas idosas com habilidades alteradas pelo envelhecimento, ou ainda usuários com limitações temporárias, como um braço imobilizado ou a ausência momentânea de óculos. Situações ambientais ou tecnológicas também se beneficiam da acessibilidade, como ao utilizar a web sob luz solar intensa, em locais ruidosos ou com conexões lentas ou instáveis. W3C [2025b]

Esses exemplos demonstram que práticas de acessibilidade digital promovem um design mais universal, resultando em sistemas mais robustos, flexíveis e inclusivos. Dessa forma, ao adotar padrões de acessibilidade, como os propostos nas Diretrizes da W3C, os desenvolvedores não apenas atendem às necessidades de pessoas com deficiência, mas também melhoram a experiência de todos os usuários, independentemente do contexto em que acessam a tecnologia.

Estudos anteriores, como os de Behar *et al*. [2008], enfatizam que a acessibilidade é fundamental para garantir que pessoas com deficiência visual possam acessar e utilizar ferramentas de programação. Nesse sentido, Sonza *et al*. [2013] ressaltam que pensar e buscar a inclusão sociodigital de Pessoas com Deficiência (PcD) significa projetar um mundo onde a igualdade de direitos, a dignidade e o respeito às diferenças sejam fatores preponderantes. Para esses autores, a acessibilidade digital é essencial para eliminar as barreiras que impedem o pleno uso de sistemas computacionais.

Além disso, ao tratar da construção de sites e ambientes digitais, Sonza *et al*. [2013] destaca que é importante considerar três princípios básicos: usabilidade, acessibilidade e comunicabilidade [Sonza *et al*., 2013, p. 316]. Os autores afirmam que "[..] desenvolvedores do site devem conhecer perfeitamente o que são esses princípios, como devem ser usados e como eles impactam na construção de sites acessíveis" [Sonza *et al*., 2013, p. 316].

Muitas ferramentas digitais ainda apresentam limitações, sobretudo na ausência de padronização ou até mesmo nas diretrizes como a da W3C. A falta de estudos voltados a essa área coloca em foco um gap importante na literatura, trazendo a necessidade de forma significativa a iniciativa para construção de ambiente de desenvolvimento integrado web ou até mesmo ideias não colocadas em práticas na construção da literatura.

De acordo com Berners-Lee [2005], "O poder da web está em sua universalidade. O acesso por todas as pessoas, independentemente da deficiência, é um aspecto essencial". Essa afirmação reforça a noção de que a acessibilidade deve ser um princípio fundamental desde a ideia de criação de

quaisquer sistemas web ou computacional digital, incluindo especialmente ambientes de desenvolvimento. No entanto, a escassez de iniciativas voltadas à inclusão de pessoas com deficiência visual nesses ambientes evidência um desnível entre os avanços legislativos e normativos e a prática desse desenvolvimento.

Portanto, a análise comparativa proposta neste projeto visa preencher essas lacunas, contribuindo para o desenvolvimento de ambientes de programação web mais inclusivos, padronizados e acessíveis a todos os usuários, especialmente aqueles que utilizam leitores de tela.

2.2 Legislação Brasileira de Acessibilidade Digital

A acessibilidade em Ambientes de Desenvolvimento Integrado (IDEs) para Pessoas com Deficiência (PcD) é um tema central na discussão sobre inclusão, em especial na programação e educação. Vários livros, tcc, artigos, sites, leis e entre outros enfatizam a necessidade da inclusão da acessibilidade. Em conformidade com o Decreto nº 5.296/2004 Brasil [2004], que define diretrizes de acessibilidade em diversas áreas, inclusive a comunicação digital. De acordo com o Capítulo VI, que aborda o acesso à informação e à comunicação, o artigo 47 estabelece que

> [..] será obrigatória a acessibilidade nos portais e sítios eletrônicos da administração pública na rede mundial de computadores (internet), para o uso das pessoas portadoras de deficiência visual, garantindo-lhes o pleno acesso às informações disponíveis.

Ela estabelece a acessibilidade como um direito básico e obriga o desenvolvimento de soluções tecnológicas que garante igualdade para todos. Outro marco relevante é a Lei nº 13.146/2015 Brasil [2015], que define que

> [...] pessoa com deficiência aquela que tem impedimento de longo prazo de natureza física, mental, intelectual ou sensorial, o qual, em interação com uma ou mais barreiras, pode obstruir sua participação plena e efetiva na sociedade em igualdade de condições com as demais pessoas Brasil [2015].

Apesar dos avanços legais, a aplicação prática da acessibilidade em ferramentas de desenvolvimento como as IDEs ainda enfrenta obstáculos. Estudos recentes demonstram a existência de barreiras significativas, como a incompatibilidade com leitores de tela, a dificuldade de adaptação de interfaces e a falta de suporte à navegação não visual Petrausch and Loitsch [2017]. Essas dificuldades são verificadas tanto na perspectiva de desenvolvedores com baixa visão Lopes [2023] quanto no contexto de estudantes com deficiência visual no aprendizado de programação Zen and Tavares [2023]. Compreender a legislação vigente é essencial para avaliar o grau de conformidade das plataformas com os direitos das pessoas com deficiência visual.

2.3 Diretrizes de Acessibilidade: W3C, WAI-ARIA, UAAG, ATAG e Desenho Universal

Segundo o W3C [2025a] o conceito de World Wide Web é

> [...] um consórcio internacional no qual organizações filiadas, uma equipe em tempo integral e o público trabalham juntos para desenvolver padrões para a Web. Liderado pelo inventor da web Tim Berners-Lee e o CEO Jeffrey Jaffe, o W3C tem como missão Conduzir a World Wide Web para que atinja todo seu potencial, desenvolvendo protocolos e diretrizes que garantam seu crescimento de longo prazo.

Dentro do W3C, existem várias diretrizes e aplicações voltados à acessibilidade na web, dentre eles estão as principais:

- Web Accessibility Initiative (WAI) WAI [2025]
- Web Content Accessibility Guidelines (WCAG) WCAG [2025]
- Accessible Rich Internet Applications (WAI-ARIA) WAI-ARIA [2017]
- User Agent Accessibility Guidelines (UAAG) UAAG [2015]
- Authoring Tool Accessibility Guidelines (ATAG) ATAG [2015]

O W3C afirma que "A WAI (Web Accessibility Initiative) do W3C desenvolve padrões e materiais de suporte para ajudá-lo a entender e implementar a acessibilidade" WAI [2025]. Ou seja, a WAI atua como uma iniciativa que organiza e desenvolve diversas diretrizes essenciais para a acessibilidade digital. Dentro da WAI, foram criadas as principais diretrizes que compõem seu conjunto de padrões: as Web Content Accessibility Guidelines (WCAG), que orientam a criação de conteúdos acessíveis; as User Agent Accessibility Guidelines (UAAG), voltadas para agentes do usuário como navegadores e leitores de tela; e as Authoring Tool Accessibility Guidelines (ATAG), que definem requisitos para ferramentas de autoria acessíveis. Juntas, essas diretrizes formam o núcleo das recomendações da WAI para promover a inclusão digital na Web.

A WCAG atualmente (2025) na versão 2.2 abrange uma variedade de recomendações para se obter um conteúdo da web mais acessível, segundo a diretriz da WCAG ela faz com que o conteúdo da web fique

> [...] mais acessível a uma gama mais ampla de pessoas com deficiência, incluindo acomodações para cegueira e baixa visão, surdez e perda auditiva, movimento limitado, deficiências de fala, fotossensibilidade e combinações destes, e algumas acomodações para dificuldades de aprendizagem e limitações cognitivas [...]

onde de fato essa abordagem pode ser aplicada a qualquer tipo de dispositivo WCAG [2025].

A WAI-ARIA na versão 1.1 é "[...] uma especificação técnica que fornece uma estrutura para melhorar a acessibilidade e a interoperabilidade de conteúdo e aplicativos da web" W3C [2006]. Ela foi desenvolvida para suprir as limitações de acessibilidade presentes em conteúdos e aplicações web que usam tecnologias como JavaScript, AJAX e HTML5, fornecendo atributos e funções que permitem descrever este comportamento, o estado e a estrutura desses elementos. Essas tecnologias muitas vezes criam interfaces complexas e interativas que não são interpretadas corretamente por tecnologias assistivas, como leitores de tela, o que dificulta o acesso para pessoas com deficiência.

A UAAG na versão 2.0, diz

> [...] os desenvolvedores na concepção de usuários agentes que tornam a web mais acessível para pessoas com Deficiência. Os agentes de usuário incluem navegadores, extensões

> de navegador, reprodutores de mídia, leitores e outros aplicativos que renderizar conteúdo da web UAAG [2015].

O objetivo principal da UAAG é garantir que os agentes de usuário fiquem acessíveis para pessoas com deficiência (PcD) permitindo uma navegação adequada com qualquer ambiente de desenvolvimento (IDE), atendendo a tecnologia assistiva na construção de qualquer sistema.

A ATAG na versão 2.0, é uma diretriz que fornece passos de como realizar a

> [...] criação de conteúdo da Web ferramentas que são mais acessíveis aos autores com deficiência (Parte A) e projetadas para permitir, apoiar e promover a produção de conteúdo web mais acessível por todos os autores (Parte B).ATAG [2015]

Dessa forma, ela tem como objetivo tornar ferramentas de autoria própria acessíveis para todos os usuários. Um exemplo prático abordado neste trabalho é a IDE Web desenvolvida no estágio supervisionado chamada EVICODE, cuja proposta é atender a esses princípios ao oferecer um ambiente de programação acessível.

O Desenho Universal (DU) segundo Carletto and Cambiaghi [2007] é o

> processo de criar os produtos que são acessíveis para todas as pessoas, independente de suas características pessoais, idade ou habilidades. [...] A meta é que qualquer ambiente ou produto poderá ser alcançado, manipulado e usado, independentemente do tamanho do corpo do indivíduo, sua postura ou sua mobilidade [Carletto and Cambiaghi, 2007, p. 10].

A ideia do Desenho Universal é criar soluções que já sejam acessíveis para todos desde o início, sem precisar de adaptações posteriores para pessoas com deficiência. Esse conceito valoriza as diferentes formas de ser e viver no mundo, buscando desenvolver produtos e tecnologias que sejam funcionais, agradáveis e que façam sentido para o maior número de pessoas possível. Com isso, não são apenas as pessoas com deficiência que se beneficiam, mas também idosos, crianças e até mesmo quem enfrenta limitações temporárias, como uma lesão no braço, por exemplo. No contexto das tecnologias digitais, como os ambientes de desenvolvimento integrados (IDEs), o Desenho Universal é uma base importante para garantir acessibilidade de forma mais ampla e eficiente.

O Desenho Universal possui sete princípios, criado por um grupo de arquitetos e Ron Mace na década de 90, estes princípios são adotados para qualquer desenvolvimento de acessibilidade [Carletto and Cambiaghi, 2007, p. 12-17]. Os princípios são:

1. Igualitário (Uso equiparável)
2. Adaptável (Uso flexível)
3. Óbvio (Uso simples e intuitivo)
4. Conhecido (Informação de fácil percepção)
5. Seguro (Tolerante ao erro)
6. Sem esforço (Baixo esforço físico)
7. Abrangente (Dimensão e espaço para aproximação e uso)

Dessa forma, as diretrizes da W3C, como a WCAG, WAI-ARIA, UAAG e ATAG, junto aos princípios do Desenho Universal, formam a base teórica essencial para a construção de ambientes digitais acessíveis. A aplicação integrada desses referenciais promove não apenas a conformidade técnica, mas também a inclusão efetiva de pessoas com diferentes tipos de deficiência, especialmente aquelas com deficiência visual. Para o desenvolvimento de Ambientes de Desenvolvimento Integrado (IDEs) acessíveis, compreender e aplicar essas diretrizes e princípios é fundamental para garantir que as ferramentas sejam utilizáveis, intuitivas e adaptáveis a todas as necessidades dos usuários, contribuindo para a democratização do acesso à tecnologia e ao conhecimento.

2.4 Tecnologia Assistiva para deficientes visuais

Tecnologia assistiva (TA) de acordo com Bersch and Pelose [2006] é

> [...] uma área de conhecimento que engloba recursos e serviços com o objetivo de proporcionar ou ampliar habilidades funcionais de uma pessoa com deficiência ou com incapacidades advindas do envelhecimento. O objetivo da TA é o de promover qualidade de vida e inclusão social de seus usuários [Bersch and Pelose, 2006, p. 7].

A TA é, portanto, uma área ampla que abrange desde dispositivos físicos simples, como lupas e bengalas, até soluções de software complexas [Motta, 2022, p. 20]. De acordo com uma pesquisa de Sartoretto and Bersch [2017], existem 12 categorias de TA, que incluem:

1. Auxílios para a vida diária e vida prática
2. Comunicação Aumentativa e Alternativa (CAA)
3. Recursos de acessibilidade ao computador
4. Sistemas de controle de ambiente
5. Projetos arquitetônicos para acessibilidade
6. Órteses e próteses
7. Adequação Postural
8. Auxílios de mobilidade
9. Auxílios para ampliação da função visual e recursos que traduzem conteúdos visuais em áudio ou informação tátil
10. Auxílios para melhorar a função auditiva e recursos utilizados pra traduzir os conteúdos de áudio em imagens, texto e língua de sinais
11. Mobilidade em veículos
12. Esportes e lazer

Dentre esse amplo espectro, e para atender ao foco deste artigo, a análise se concentra nas TAs voltadas para o uso de interfaces e sistemas digitais. Especificamente, este trabalho se relaciona com a Categoria 3 ("Recursos de acessibilidade ao computador") e a Categoria 9 ("Auxílios para ampliação da função visual e recursos que traduzem conteúdos visuais em áudio ou informação tátil") Motta [2022]; Sartoretto and Bersch [2017].

Nesse contexto digital, destacam-se os recursos específicos amplamente utilizados em ambientes de desenvolvimento integrados (IDEs) para pessoas com deficiência visual. Entre eles, os leitores de tela, como NVDA, JAWS e VoiceOver, que convertem o conteúdo visual em áudio; as linhas Braille, que tornam o texto acessível por meio do tato; softwares de ampliação de tela, como o ZoomText; temas de alto contraste e modos escuros, que facilitam a visualização para usuários com baixa visão; além de suporte à navegação por teclado e controle por comandos de voz. Esses recursos são fundamentais para promover a inclusão de desenvolvedores com deficiência visual no universo da programação.

2.5 Ambientes de Desenvolvimento Integrados (IDEs)

Segundo a RedHat [2023], um Ambiente de Desenvolvimento Integrado (IDE)

> [...] é um software para criar aplicações que combina ferramentas comuns de desenvolvedor em uma única interface de usuário gráfica (GUI). Um IDE geralmente consiste em: Editor de código-fonte: é um editor de texto que auxilia na criação de código de software por meio de funcionalidades como destaque da sintaxe com indicadores visuais, recurso de preenchimento automático específico da linguagem e verificação de bugs durante o desenvolvimento. Automação de compilação local: são utilitários que automatizam tarefas simples e repetíveis durante a criação de uma compilação local do software usada pelo desenvolvedor. São tarefas como compilação de código-fonte em código binário, criação de pacotes de código binário e execução de testes automatizados. Debugger: é um programa usado para testar outros programas e mostrar graficamente a localização do bug no código original. RedHat [2023]

O objetivo dessas ferramentas é reunir em um único software ou sistema web, todos os recursos para realizar o desenvolvimento de um software, facilitando a vida de um programador.

Dessa forma, cada IDE possui suas particularidades e funcionalidades específicas para atender diferentes linguagens e perfis de desenvolvedores. Segundo a RedHat [2023], as IDEs mais utilizadas atualmente incluem IntelliJ IDEA, Visual Studio Code, Eclipse e Android Studio, entre outras. Por exemplo, o Visual Studio Code destaca-se por sua leveza, ampla biblioteca de extensões e grande comunidade de usuários, o que o torna uma das IDEs mais populares no desenvolvimento web e de software em geral Microsoft [2025a]. Já o IntelliJ é bastante apreciado por seu suporte avançado a linguagens como Java e Kotlin, além de recursos inteligentes para produtividade Intellij [2000].

O Android Studio, por sua vez, é a IDE oficial para desenvolvimento de aplicativos Android, fornecendo ferramentas específicas para design, codificação, teste e depuração em dispositivos móveis. Sua integração com o Android SDK facilita a criação de apps otimizados para a vasta gama de dispositivos Android for Developers [2025].

O Eclipse configura-se como uma das IDEs mais tradicionais e consolidadas no mercado. Sua principal vantagem reside na flexibilidade e na ampla capacidade de expansão, viabilizada por um sistema robusto de plugins que possibilita a adaptação da ferramenta a diversas linguagens e tipos de projeto. Em razão dessas características, é amplamente utilizado em ambientes corporativos e em projetos de grande escala, sobretudo no desenvolvimento em Java Eclipse [2025].

3 Trabalhos relacionados

Esta seção apresenta trabalhos relacionados sobre metodologias usadas para verificar a acessibilidade em ambientes.

Um estudo relevante é o de Mateus *et al*. [2023], apresentado no Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais (IHC). Embora focado em aplicativos móveis, sua metodologia é central para esta pesquisa, pois comparou múltiplas abordagens de avaliação de acessibilidade: testes com usuários com deficiência visual, inspeções manuais por especialistas (de IHC e de desenvolvimento) e ferramentas automatizadas. O estudo demonstrou que as ferramentas automatizadas tiveram desempenho limitado, enquanto os testes com usuários e as inspeções de especialistas em IHC revelaram uma maior diversidade de problemas. Esses achados reforçam a importância da avaliação prática para identificar barreiras reais, alinhando-se diretamente à escolha metodológica deste trabalho de utilizar o NVDA para simular a experiência do usuário.

Na aplicação direta das diretrizes como critério de avaliação, o trabalho de Baldiris *et al*. [2022] também oferece um precedente metodológico importante. O estudo avaliou quatro ferramentas de autoria (authoring tools) populares em duas frentes distintas: primeiro, a acessibilidade das próprias interfaces das ferramentas, com base nas recomendações do ATAG 2.0; e segundo, a acessibilidade do conteúdo educacional gerado por elas, utilizando as recomendações do WCAG 2.1. A metodologia empregou uma avaliação manual sistemática, auxiliada por tecnologias assistivas como o leitor de tela NVDA. Essa abordagem de verificar tanto a ferramenta (ATAG) quanto o seu resultado (WCAG) reforça a validade do método adotado neste artigo.

Especificamente sobre a análise de IDEs, o estudo de Petrausch and Loitsch [2017] investigou a acessibilidade da IDE Eclipse para usuários com deficiência visual. A metodologia foi baseada em um questionário online aplicado a 14 desenvolvedores cegos ou com baixa visão, que relataram usar leitores de tela como NVDA e JAWS. O estudo focou nas dificuldades práticas encontradas por eles, revelando problemas significativos na identificação de erros e avisos, inacessibilidade de plugins de terceiros e falta de anúncios em tempo real. Embora o objeto seja uma IDE desktop e o método um survey (levantamento por questionário), os resultados identificam as mesmas barreiras reais de uso que esta pesquisa analisou por meio de testes práticos.

No contexto nacional, trabalhos recentes também aplicam metodologias similares. O estudo de Lopes [2023] apresenta um estudo de caso prático focado na verificação do nível de acessibilidade de ferramentas de desenvolvimento sob a perspectiva de um desenvolvedor com baixa visão, utilizando uma ferramenta de ampliação. De forma complementar, Zen and Tavares [2023] investiga estratégias de acessibilidade em IDEs no contexto educacional, reforçando a importância da análise de ferramentas voltadas para o aprendizado, o que se conecta à seleção do Online-IDE.com neste estudo.

Esses trabalhos demonstram a validade da combinação de testes práticos com tecnologias assistivas e análise baseada em diretrizes para investigar a acessibilidade de software. No entanto, a presente pesquisa se distingue por aplicar essa metodologia especificamente ao domínio de IDEs Web recentes (2020-2025), um nicho ainda pouco explorado na literatura, e por incluir uma IDE Web desenvolvida chamada EVICODE, com foco em acessibilidade como ponto de comparação.

4 Metodologia

Este trabalho adotou uma abordagem metodológica que combina a pesquisa exploratória, descritiva e comparativa. O principal objetivo foi analisar e comparar a acessibilidade de Ambientes de Desenvolvimento Integrado (IDEs) Web modernas, lançadas entre 2020 e 2025. A pesquisa foi conduzida em três etapas principais: um levantamento inicial para selecionar as IDEs com base em critérios de representatividade de mercado e complexidade da interface; uma análise compa-

rativa baseada nos princípios de Acessibilidade, Usabilidade e Comunicabilidade [Sonza *et al*., 2013, p. 316], aplicados por meio de tarefas práticas de desenvolvimento; e, por fim, a comparação dos resultados do EVICODE desenvolvida no estágio supervisionado.

4.1 Levantamento e Seleção de IDEs Web

A primeira etapa deste trabalho consistiu em um levantamento exploratório para identificar IDEs Web com recursos de acessibilidade, com foco em ferramentas lançadas ou significativamente atualizadas entre 2020 e 2025. Para esta fase, optou-se pela utilização do motor de busca Google como ferramenta principal. A escolha se justifica por três motivos centrais: primeiro, reflete o comportamento de desenvolvedores, público-alvo das ferramentas, que comumente utilizam buscadores para encontrar e avaliar novas tecnologias. Segundo, a natureza da busca por softwares recentes e suas documentações online consideradas "literatura cinzenta"torna os motores de busca mais eficazes do que bases de dados acadêmicas tradicionais. Terceiro, a pesquisa visava mapear o cenário como ele se apresenta a um usuário comum, incluindo a dificuldade de encontrar soluções que se declarem acessíveis.

Foram utilizadas as seguintes palavras-chave, em português: "IDEs acessíveis", "IDEs Web acessíveis", "IDE com tecnologias assistivas para programação", "acessibilidade em ambientes de desenvolvimento web", "IDE web com leitor de tela"e "IDEs web 2020-2025".

A primeira etapa da metodologia foi o levantamento de ferramentas. O resultado foi notável: não foi encontrado nenhum Ambiente de Desenvolvimento Integrado (IDE) baseado na web que, em sua documentação oficial ou material de divulgação, se apresentasse como uma solução com recursos de acessibilidade integrados para pessoas com deficiência visual.

As buscas realizadas com as palavras-chave ["IDEs acessíveis", "IDEs Web acessíveis", etc.] não retornaram ferramentas práticas que se enquadrassem nos critérios do estudo (lançadas ou atualizadas entre 2020 e 2025). Em vez disso, os resultados foram predominantemente compostos por:

1. **Listagens de Ferramentas Genéricas:** Foram encontrados diversos artigos e rankings de "melhores IDEs"para o período de 2020-2025. No entanto, os critérios de avaliação dessas ferramentas se limitavam a desempenho, integração com outras tecnologias e popularidade, sem qualquer menção a recursos de acessibilidade nativos.
2. **Documentação Conceitual:** As buscas retornaram predominantemente conteúdo teórico, como guias sobre as diretrizes W3C, artigos definindo "Tecnologia Assistiva"e manuais de boas práticas para a criação de aplicações web acessíveis. Este material educa sobre como tornar um produto acessível, mas não aponta para um ambiente de desenvolvimento que seja, ele mesmo, acessível.
3. **Foco em Ferramentas Assistivas Externas:** Em vez de IDEs com leitores de tela integrados, os resultados indicaram softwares leitores de tela de terceiros (como NVDA e JAWS), tratando a acessibilidade como uma responsabilidade do usuário e de uma adaptação externa, e não como uma funcionalidade nativa da ferramenta de desenvolvimento.

O resultado mais importante desta etapa exploratória foi perceber essa aparente realidade. Os dados levantados sugerem um grande vazio entre a teoria (o que se discute sobre inclusão) e a prática (as ferramentas disponíveis). O fato de não terem sido encontradas IDEs Web que se promovam como acessíveis reforça o ponto principal deste trabalho: programadores com deficiência visual podem não ter acesso fácil às suas próprias ferramentas de trabalho. Por isso, torna-se importante e urgente estudar esse assunto.

4.2 Ferramentas Selecionadas para Análise

Diante da ausência de IDEs web que se promovessem como acessíveis (conforme descrito na seção anterior), a metodologia deste estudo se voltou para a análise de ferramentas populares. A seleção dos objetos de estudo não foi aleatória, mas sim baseada em uma abordagem de estudo de casos múltiplos [Yin, 2001, p. 67].

O método utilizado para a seleção dos casos foi a amostragem intencional (ou proposital) [Patton, 2002, p. 230]. As ferramentas não foram escolhidas por conveniência, mas sim para representar perfis distintos e relevantes para a análise comparativa. De acordo com Patton [Patton, 2002, p. 230], a lógica da amostragem proposital é selecionar casos ricos em informação, dos quais se pode aprender mais sobre o problema em questão. Dessa forma, cada caso foi selecionado para cobrir um aspecto diferente do problema de pesquisa:

1. **Caso de Maior Representatividade de Mercado:** Visual Studio Code for the Web (vscode.dev) (Figura 1) Microsoft [2025b]. Conforme exibido na imagem, a interface da ferramenta organiza-se em áreas distintas: a barra lateral de atividades e gerenciamento de arquivos à esquerda, o amplo editor de código ao centro e a barra de status na parte inferior. A seleção desta ferramenta se justifica por sua massiva popularidade. De acordo com a "Stack Overflow Developer Survey", o Visual Studio Code é a IDE mais utilizada pela comunidade global de desenvolvedores Overflow [2023]. Analisá-lo garante que os resultados reflitam a realidade da ferramenta utilizada pela vasta maioria dos profissionais.
2. **Caso Contrastante (Minimalismo):** Online-IDE.com (OneCompiler) (Figura 2) OneCompiler [2025]. A figura ilustra o layout da ferramenta, que é dividido visualmente em dois painéis principais: o editor de código, posicionado à esquerda, e o terminal de entrada e saída de dados, à direita, com o botão de execução em destaque no topo. Esta ferramenta foi escolhida por representar o extremo oposto do VS Code: a simplicidade e o minimalismo, um princípio de design frequentemente associado à boa usabilidade Nielsen [2024]. Sua seleção permite investigar a hipótese de que a complexidade da interface é um fator determinante nas barreiras de acessibilidade.
3. **Caso de Comparação (EVICODE):** A IDE desenvolvida no estágio supervisionado (EVICODE) (Figura 3). A imagem apresenta o design do protótipo, composto por uma área central de edição e painéis laterais específicos para as funcionalidades assistivas implementadas, como os menus de configuração do leitor e comandos de voz. Além das ferramentas de mercado, a análise in-

clui este protótipo funcional. Ele foi incluído não como um produto, mas como um ponto de comparação para demonstrar como a implementação de diretrizes de acessibilidade desde a concepção do projeto pode resultar em uma experiência de uso mais inclusiva.

Para realizar os testes práticos de acessibilidade, foi selecionado o seguinte software de tecnologia assistiva:

- **Leitor de Tela NVDA (NonVisual Desktop Access):** A escolha pelo NVDA é justificada por sua alta taxa de adoção na comunidade de usuários com deficiência visual. De acordo com pesquisas de mercado, como a "WebAIM Screen Reader User Survey", o NVDA é um dos softwares gratuitos mais populares do mundo. Sua utilização garante que a análise reflita a experiência de um segmento de usuários real e significativo WebAIM [2024].

A análise comparativa entre essas três IDEs, utilizando o NVDA como ferramenta de teste, permitirá gerar evidências concretas sobre as barreiras de acessibilidade existentes e o potencial de melhoria na área.

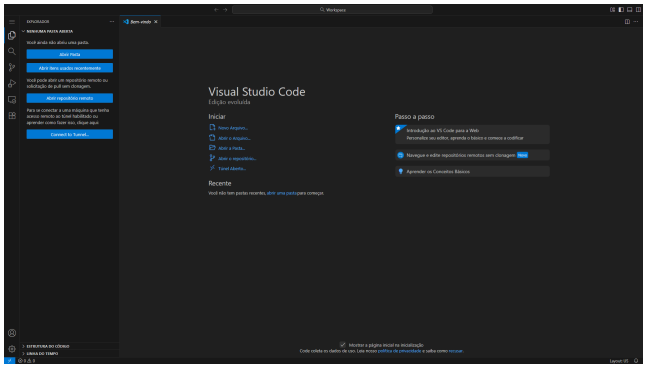

Figura 1. Interface do Visual Studio Code for the Web

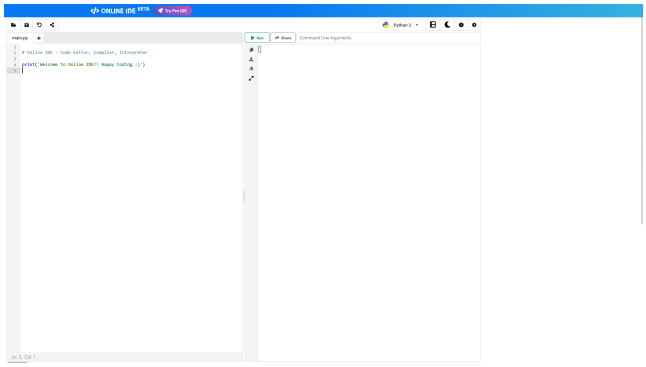

Figura 2. Interface do Online-IDE.com

4.3 Critérios de Avaliação e Tarefas de Teste

Para garantir uma análise alinhada aos objetivos do estudo, a metodologia de avaliação foi estruturada em duas partes: um conjunto de Critérios de Avaliação (o que avaliar) e um conjunto de Tarefas de Teste (como avaliar).

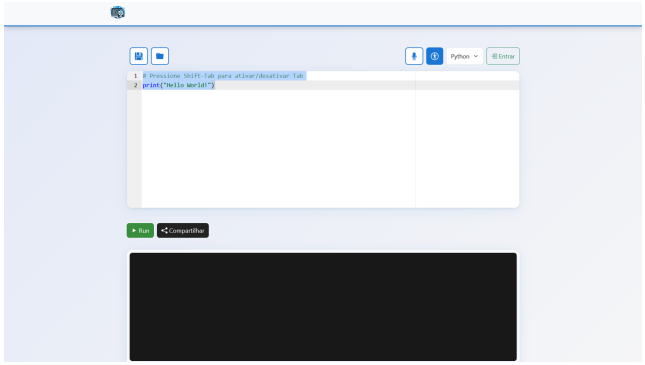

Figura 3. Interface do Editor Visual Inclusivo - EVICODE

4.3.1 *Critérios de Avaliação*

Baseado na metodologia descrita no artigo e fundamentada em [Sonza *et al.*, 2013, p. 316], cada IDE foi analisada sob três princípios centrais, que formaram a base da matriz de avaliação:

- **Acessibilidade:** Avaliou o nível de conformidade técnica com as diretrizes W3C. Verificou-se se a ferramenta era perceptível, operável e compatível com tecnologias assistivas (NVDA), tendo como base as diretrizes WCAG 2.2, WAI-ARIA 1.3, ATAG 2.0 e UAAG 2.0.
- **Usabilidade:** Avaliou a facilidade de uso e a eficiência para um usuário com deficiência visual. Buscou-se responder se era possível completar uma tarefa de forma fluida e se a curva de aprendizado dos atalhos era razoável.
- **Comunicabilidade:** Avaliou a qualidade do feedback fornecido pela interface ao usuário através do leitor de tela. Buscou-se entender se o NVDA anunciava as informações corretas, no momento certo, e se o usuário compreendia o que estava acontecendo na IDE (ex: erros de sintaxe) apenas pelo feedback sonoro.

4.3.2 *Tarefas de Teste*

Para aplicar os três critérios acima de forma prática e consistente, um conjunto de 4 tarefas simulando o fluxo de trabalho de um desenvolvedor foi executado em cada IDE, utilizando exclusivamente o teclado, mouse e o leitor de tela NVDA:

- **Tarefa 1: Navegação e Gerenciamento de Arquivos:** Abrir a IDE, navegar pela interface, acessar os arquivos, navegar entre pastas e criar um novo arquivo (ex: "index.html ou main.py").
- **Tarefa 2: Edição de Código e Feedback em Tempo Real:** Abrir o arquivo no editor, escrever um bloco de código simples, verificar o feedback do NVDA durante a digitação, o anúncio de autocompletar e, crucialmente, o anúncio de erros de sintaxe.
- **Tarefa 3: Execução e Verificação de Saída:** Encontrar e acionar a função de "Executar"o código e, em seguida, acessar e ler a saída no console (seja uma mensagem de sucesso ou um erro de compilação).
- **Tarefa 4: Acesso a Recursos de Suporte:** Acessar os menus principais (ex: "Arquivo", "Configurações") via teclado e tentar ativar um recurso de acessibilidade, como um tema de alto contraste.

Os resultados apresentados na Seção Resultados e Análise são o produto da avaliação dos três critérios Acessibilidade, Usabilidade e Comunicabilidade durante a execução de cada uma dessas quatro tarefas em cada IDE.

5 Resultados e Análise

Nesta seção, apresentamos os resultados da análise prática de cada uma das três ferramentas selecionadas, utilizando o leitor de tela NVDA. A avaliação buscou identificar tanto os recursos de acessibilidade presentes quanto as barreiras encontradas durante a execução de tarefas comuns de programação, como escrever, navegar e depurar código.

5.1 Visual Studio Code for the Web

A análise do `vscode.dev` (Figura 1) Microsoft [2025b] revelou uma ferramenta que, apesar de funcional para edição de texto, apresenta barreiras de acessibilidade, usabilidade e comunicabilidade que impedem o fluxo de trabalho completo de um desenvolvedor com deficiência visual. Os resultados da avaliação, baseados nos três critérios metodológicos, estão detalhados na Tabela 1. O texto a seguir mostra as barreiras e sucessos encontrados em cada tarefa de teste.

5.1.1 Análise detalhada por Tarefa (VS Code)

Tarefa 1: Navegação e Gerenciamento de Arquivos **Status: Concluída com barreiras significativas.** A navegação com mouse apresentou uma falha de Comunicabilidade, pois o NVDA demonstrou latência, terminando de ler o elemento anterior antes de anunciar o novo foco do mouse. A navegação com teclado ("Tab") teve uma falha crítica de Usabilidade, mostrando-se inviável, pois a IDE não anuncia a localização dos elementos (barra lateral, menu, etc.), impedindo o usuário de construir um "mapa mental"[2] da interface. Em Acessibilidade, a criação de uma nova pasta não foi possível apenas com o teclado (falha no WCAG 2.2 - Teclado), forçando o uso do mouse.

Tarefa 2: Edição de Código e Feedback em Tempo Real **Status: Concluída com barreiras severas.** A Comunicabilidade foi bastante comprometida: o leitor de tela não descreve as sugestões do autocomplete (apenas as lê), não informa o número da linha atual e não anuncia o fechamento automático de parênteses. A falha mais crítica de Acessibilidade foi que o NVDA não identificou nem anunciou os erros de sintaxe no código, uma violação direta da diretriz ATAG 2.0 (Alternativas Acessíveis), que exige que a ferramenta auxilie na correção de erros.

Tarefa 3: Execução e Verificação de Saída **Status: Falha Crítica.** A tarefa não pôde ser concluída. O `vscode.dev` não suporta a execução de Python ou a extensão "Live Server"(HTML), exibindo um aviso de incompatibilidade. Isso representa uma falha crítica de Acessibilidade, pois a funcionalidade principal não está disponível. Adicionalmente, foi encontrada uma falha de Usabilidade no marketplace de extensões, onde a seleção de um item exige o uso das setas do teclado, e não a navegação padrão "Tab"+ "Enter", quebrando a consistência da interação.

Tarefa 4: Acesso a Recursos de Suporte **Status: Concluída com Sucesso.** Esta foi a única tarefa totalmente funcional. Conforme observado no vídeo Silva [2025c], foi possível navegar (via mouse) até o menu "Configurações"e alterar o "Tema de Cores"da IDE, demonstrando conformidade nos três critérios (Acessibilidade, Usabilidade e Comunicabilidade) para esta tarefa específica.

5.2 Online-IDE.com (OneCompiler)

A análise do `Online-IDE.com` (Figura 2) OneCompiler [2025] revelou uma ferramenta que, apesar de sua interface visualmente minimalista com poucas funcionalidades presentes, não possui recursos de acessibilidade nativos (além de tema e fonte). A ferramenta repetiu falhas encontradas no VS Code e introduziu barreiras críticas próprias, principalmente na gestão de arquivos e na execução de código, tornando seu uso difícil para um desenvolvedor com deficiência visual. Os resultados da avaliação estão detalhados na Tabela 2.

5.2.1 Análise detalhada por Tarefa (Online-IDE.com)

Tarefa 1: Navegação e Gerenciamento de Arquivos **Status: Falha Crítica.** A Comunicabilidade da ferramenta é quase nula. O NVDA, seja com mouse ou "Tab", falha em ler a maioria dos elementos, lendo apenas o botão "Executar"e o seletor de linguagem de forma inconsistente. A navegação via "Tab"apresentou a mesma falha do VS Code: o leitor de tela "corta"o anúncio anterior para ler o próximo, e não informa a localização dos elementos, impedindo a criação de um "mapa mental". Além disso, o NVDA lê elementos HTML crus[3](como "select"e "input") em vez de elementos amigáveis (como "Botão para executar o código"). A falha mais crítica de Acessibilidade é que a IDE não possui funcionalidade para criar pastas, ela só pode salvar arquivos em pastas já existentes através do sistema operacional utilizado.

Tarefa 2: Edição de Código e Feedback em Tempo Real **Status: Concluída com barreiras.** Assim como no VS Code, esta IDE falha em Comunicabilidade e Acessibilidade ao não prover nenhum *feedback* sonoro sobre erros de sintaxe, sucesso na compilação, número da linha atual ou falta de ponto e vírgula (violando a ATAG 2.0). O recurso de *autocomplete* é invisível para o leitor de tela; uma caixa de opções aparece visualmente, mas o NVDA não anuncia que uma lista de seleções está disponível. Um ponto positivo, diferente do VS Code, é que a navegação com as setas (cima/baixo) lê a linha inteira do código, facilitando a revisão.

Tarefa 3: Execução e Verificação de Saída **Status: Falha Crítica.** Esta tarefa falhou em todos os critérios. A Usabilidade foi quebrada pois o foco do teclado ficou "preso"no editor de código, impossibilitando a navegação até o botão "Executar"via "Tab"e forçando o uso do mouse. Após a execução, a falha de Acessibilidade foi total: o terminal de saída é inacessível ao leitor de tela. Embora o código tenha executado e solicitado um *input* do usuário, não foi possível ler

[2]Quando falo em Mapa Mental, refiro-me à capacidade do usuário de entender a estrutura da tela sem vê-la. Enquanto nós, videntes, batemos o olho e entendemos a tela inteira de uma vez, o usuário com deficiência visual recebe a informação de forma sequencial, item por item. Ele precisa memorizar essa ordem para se localizar. Se a IDE não seguir uma lógica, ele não consegue formar esse mapa e se perde completamente.

[3]Refere-se à falha onde o leitor de tela anuncia as tags estruturais de marcação (como div, span, input) ou atributos técnicos genéricos, em vez de ler o rótulo acessível ou a função do elemento para o usuário.

Tabela 1. Tabela de avaliação do Visual Studio Code for the Web.

Tarefa de Teste	Acessibilidade	Usabilidade	Comunicabilidade
T1: Navegação/Arquivos	**Falha** (Ação de criar pasta sem teclado - WCAG 2.2)	**Falha Crítica** (Navegação via "Tab" é inviável, sem mapa mental)	**Falha** (Latência no mouse; Diálogo técnico)
T2: Edição de Código	**Falha Crítica** (Não anuncia erros de sintaxe - ATAG 2.0)	**Falha** (Difícil navegar na loja de extensões)	**Falha Crítica** (Não anuncia linha, fecham. de parênteses)
T3: Execução de Código	**Falha Crítica** (Extensões de Python e Live Server são incompatíveis)	– (Tarefa não pôde ser concluída)	– (Tarefa não pôde ser concluída)
T4: Acesso a Recursos	**Sucesso** (Menu de temas acessível)	**Sucesso** (Menu fácil de achar)	**Sucesso** (Mudança de tema foi clara)

Tabela 2. Tabela de avaliação do Online-IDE.com (OneCompiler).

Tarefa de Teste	Acessibilidade	Usabilidade	Comunicabilidade
T1: Navegação/Arquivos	**Falha Crítica** (IDE não consegue criar pastas, apenas salvar)	**Falha Crítica** (Navegação "Tab"corta a fala; Não cria "mapa mental")	**Falha Crítica** (NVDA não lê a maioria dos elementos; Lê HTML cru)
T2: Edição de Código	**Falha Crítica** (Não anuncia erros de sintaxe - ATAG 2.0)	**Falha** (Autocomplete é invisível para o leitor de tela)	**Falha Crítica** (Não anuncia nº da linha nem erros/sucesso)
T3: Execução de Código	**Falha Crítica** (Terminal de saída é inacessível)	**Falha Crítica** (Foco do teclado fica "preso" no editor; Exige mouse)	**Falha Crítica** (Não anuncia a saída e não permite input)
T4: Acesso a Recursos	**Falha** (Lê elementos HTML - WCAG 2.2)	**Falha** (Informações confusas para o usuário)	**Falha Crítica** (Anúncios com "termos técnicos"desnecessários)

essa solicitação nem fornecer a informação, tornando a IDE difícil para se usar.

Tarefa 4: Acesso a Recursos de Suporte **Status: Falha.** Embora o acesso à página de Configurações tenha sido possível, a Comunicabilidade falhou. O NVDA leu a estrutura da página usando termos técnicos e nomes de elementos HTML (como "lista de não programadores"), em vez de rótulos claros e objetivos. Isso sobrecarrega o usuário com deficiência visual com informações de implementação desnecessárias, quando ele precisa apenas de rótulos funcionais (ex: "Botão: Mudar Tema"). Isso representa uma falha em seguir a diretriz WCAG 2.2.

5.3 Editor Visual Inclusivo - EVICODE

A terceira ferramenta analisada foi o protótipo da IDE desenvolvida no estágio supervisionado chamado EVICODE (Figura 3). Esta IDE se diferencia fundamentalmente das outras por possuir um leitor de tela acoplado, não dependendo de softwares externos como o NVDA. O EVICODE foi desenvolvido com um *layout* simples (semelhante ao `Online-IDE.com`) e, por ser um protótipo local, atualmente suporta apenas a linguagem Python. Os resultados da avaliação, detalhados na Tabela 3, revelam uma experiência de navegação superior, mas com falhas de *feedback* de programação, semelhantes às das outras ferramentas.

5.3.1 Análise detalhada por Tarefa (EVICODE)

Tarefa 1: Navegação e Gerenciamento de Arquivos **Status: Concluída com barreiras significativas.** A navegação apresentou resultados mistos. A Comunicabilidade via teclado ("Tab") foi o ponto forte: o leitor de tela embutido é fluido, interrompe a fala anterior e anuncia o novo elemento, fornecendo contexto de localização (ajudando a criar um "mapa mental"). No entanto, a navegação via mouse apresentou uma falha de Acessibilidade, pois o leitor de tela embutido não lê os elementos sob o foco do mouse, uma funcionalidade que precisaria ser implementada. Em Acessibilidade de arquivos, o EVICODE também é limitado, pois não possui uma função interna para criar novas pastas, apenas para salvar arquivos em pastas já existentes.

Tarefa 2: Edição de Código e Feedback em Tempo Real **Status: Concluída com barreiras.** Embora a Usabilidade para navegar até a área de edição seja boa, o EVICODE apresentou falhas de Comunicabilidade e Acessibilidade, similares às das outras ferramentas. O leitor de tela embutido não lê o código caractere por caractere durante a digitação, não anuncia o número da linha atual e não verbaliza o conteúdo das listas de *autocomplete*. A falha mais grave é a não identificação de erros de sintaxe (ATAG 2.0), deixando o desenvolvedor sem *feedback* sobre a correção do código.

Tarefa 3: Execução e Verificação de Saída **Status: Falha Crítica.** Esta tarefa expôs a maior falha do protótipo. A Usabilidade foi parcialmente atendida com um comando "Tab"que permite sair da área do editor e focar no botão "Executar". O código Python é executado com sucesso e a saída é impressa no terminal. No entanto, a Acessibilidade do terminal é nula: o foco do leitor de tela não consegue entrar na área do ter-

Tabela 3. Tabela de avaliação do EVICODE.

Tarefa de Teste	Acessibilidade	Usabilidade	Comunicabilidade
T1: Navegação/Arquivos	**Falha** (Não cria pastas; Foco do mouse não é lido)	**Falha (Mouse) / Sucesso (Tab)** (Navegação "Tab"é intuitiva)	**Falha (Mouse) / Sucesso (Tab)** (Leitor é fluido no "Tab"e anuncia localização; Mouse não é lido)
T2: Edição de Código	**Falha Crítica** (Não anuncia erros de sintaxe - ATAG 2.0)	**Sucesso** (Navegação "Tab"para editor é clara)	**Falha Crítica** (Não lê ao digitar, sem nº de linha)
T3: Execução de Código	**Falha Crítica** (Terminal de saída é inacessível)	**Falha** (Código executa, mas saída não pode ser lida)	**Falha** (Não há notificação de sucesso/erro)
T4: Acesso a Recursos	**Sucesso** (Comandos de voz, config. de leitor, login facial)	**Sucesso** (Menus de login e voz são claros)	**Sucesso** (Leitor embutido dá *feedback* das ações)

minal, impedindo o usuário de ler a saída do programa ou de inserir dados (como a idade no teste). Além disso, falta Comunicabilidade, pois nenhuma notificação sonora de "sucesso"ou "erro"é emitida, diferente do que acontece em outras áreas do EVICODE, como nas configurações.

Tarefa 4: Acesso a Recursos de Suporte **Status: Concluída com Sucesso.** Esta é a área onde o EVICODE mais inova, demonstrando alta Acessibilidade e Comunicabilidade com recursos que não existem nas outras ferramentas analisadas:

- **Comandos de Voz:** Permite ao usuário executar ações como "Copiar"e "Colar". O EVICODE fornece *feedback* sonoro após a conclusão da ação.
- **Configurações do Leitor:** Um menu de configurações permite ao usuário alterar a voz, o idioma e a velocidade do leitor de tela embutido. Ao salvar uma configuração, o sistema fornece um *feedback* visual e sonoro de sucesso.
- **Múltiplos Métodos de Login:** A ferramenta oferece login via e-mail/senha e também por reconhecimento facial ("Face ID"). Em todos os métodos, tanto o sucesso quanto o erro no login são reportados ao usuário com *feedback* sonoro e visual.

5.4 Análise Comparativa Baseada nas Diretrizes

Após a análise individual, esta seção apresenta uma tabela comparativa que agrupa as falhas de acessibilidade encontradas com base nas diretrizes WCAG, ATAG, WAI-ARIA e UAAG, propostas no título deste artigo. A Tabela 4 demonstra como as falhas identificadas nos testes práticos (Seções 5.1, 5.2 e 5.3) se conectam diretamente às violações das diretrizes técnicas.

6 Conclusão

Este trabalho realizou uma análise comparativa da acessibilidade em Ambientes de Desenvolvimento Integrado (IDEs) Web, com o objetivo de investigar o nível de acessibilidade das ferramentas atuais e as barreiras para a inclusão de pessoas com deficiência visual. A hipótese central, de que as ferramentas modernas ainda apresentam barreiras significativas, foi confirmada.

Respondendo à primeira pergunta de pesquisa — "qual é o nível de acessibilidade oferecido pelas IDEs atuais?"— este estudo conclui que o nível é crítico. A análise das IDEs de mercado, `vscode.dev` (Seção 4.2) e `Online-IDE.com` (Seção 4.2), demonstrou que, embora elementos de interface básicos possam ser funcionais (como a troca de temas, vista na Tarefa 4), ambas falham em prover um fluxo de trabalho de programação completo e acessível.

A descoberta mais significativa foi a identificação de um padrão de falha nas tarefas centrais de desenvolvimento, presente em todas as ferramentas analisadas. As barreiras mais críticas foram:

- A falha universal em prover *feedback* sonoro sobre erros de sintaxe em tempo real, uma violação direta do critério ATAG 2.0 (conforme Seção 2.3).
- A inacessibilidade total ou parcial dos terminais de execução de código, impedindo o desenvolvedor de ler a saída do programa ou inserir dados.
- A incompatibilidade de IDEs Web populares, como o `vscode.dev`, com extensões essenciais para a execução de código (como Python), tornando a ferramenta inútil para testes.

O protótipo EVICODE demonstrou que é possível solucionar grandes problemas de navegação (como a criação de um "mapa mental"via "Tab") e comunicabilidade (com *feedback* em menus e comandos de voz), onde as outras IDEs falharam. Contudo, o protótipo ainda falhou nas mesmas tarefas centrais (anúncio de erros e terminal inacessível).

Isto responde à segunda pergunta de pesquisa — "por que ainda não existe uma IDE Web com leitor de tela integrado?". A razão não é a impossibilidade de integrar um leitor (o EVICODE provou que isso é viável para a navegação), mas sim a complexidade do *fluxo de trabalho de programação*. A lacuna identificada neste estudo é que o foco da acessibilidade tem sido dado à interface, mas não às tarefas de programação (como depuração e acesso ao terminal). A dificuldade em resolver essas tarefas centrais é a barreira para uma IDE Web verdadeiramente acessível e integrada.

Como limitação deste estudo, reconhece-se que a avaliação foi conduzida pela própria autora utilizando um leitor de tela (NVDA) para simular a experiência (conforme Seção 4.2), e não por um grupo de desenvolvedores com deficiência visual. Para trabalhos futuros, sugere-se a validação destes achados através de testes com usuários finais (desenvolvedores

Tabela 4. Análise comparativa das falhas de acessibilidade por diretriz.

Diretriz Violada	Requisito Principal (da Fundamentação Teórica)	Barreiras Encontradas nos Testes
ATAG 2.0	A ferramenta de autoria deve ajudar o autor a identificar e corrigir erros de sintaxe (conforme Seção 2.3).	**Falha Crítica (Todas as 3 IDEs):** Nenhuma das ferramentas analisadas (`vscode.dev`, `Online-IDE.com` e o `EVICODE` anunciou erros de sintaxe ao leitor de tela.
WAI-ARIA 1.3	Fornecer *feedback* sobre mudanças de conteúdo dinâmico (como um terminal ou um *autocomplete*) (conforme Seção 2.3).	**Falha Crítica (Online-IDE, EVICODE):** O terminal de saída de código é inacessível. O usuário não pode ler a saída do programa nem inserir dados (ver Seção 5.2 e 5.3). **Falha (Online-IDE, EVICODE):** O *autocomplete* aparece visualmente, mas não é anunciado ao leitor de tela (ver Seção 5.2 e 5.3).
WCAG 2.2	Todo conteúdo deve ser acessível via teclado e o foco não pode ficar "preso"(conforme Seção 2.3).	**Falha (Online-IDE):** O foco do teclado fica "preso"no editor, exigindo o mouse para executar o código (ver Seção 5.2). **Falha (VS Code, EVICODE):** Funções essenciais (como "criar pasta") não foram acessíveis via teclado (ver Seção 5.1 e 5.3).
UAAG 2.2	O agente (IDE) deve prover "Nome, Função e Valor"corretos e não deve confundir o usuário (conforme Seção 2.3).	**Falha (Online-IDE):** O leitor de tela lê elementos HTML crus (ex: "lista de não programadores") em vez de rótulos funcionais (ver Seção 5.2). **Falha (VS Code, Online-IDE):** A navegação "Tab"não anuncia a localização (contexto), impedindo o usuário de criar um "mapa mental"(ver Seção 5.1 e 5.2).

com deficiência visual), bem como a expansão dos critérios de avaliação para abranger uma verificação dos níveis de conformidade das diretrizes WCAG. Além disso, a continuidade do desenvolvimento do EVICODE é fundamental, focando especificamente na implementação das funcionalidades que faltam.

Declarações complementares

Agradecimentos

Agradeço primeiramente a Deus e a Nossa Senhora, pelo dom da vida e por me sustentarem em cada desafio, guiando meus passos ao longo dos meus estudos. Ao meu orientador, Prof. Alexandre Berndt, minha gratidão por toda a paciência e parceria fundamental na realização deste artigo. Aos demais professores que fizeram parte da minha formação, meu muito obrigada pelos ensinamentos compartilhados e por contribuírem para o meu crescimento profissional. Aos meus pais, meu eterno agradecimento pelo amor incondicional e por, incansavelmente, ficarem acordados me esperando chegar em casa todos os dias após a faculdade o cuidado de vocês tornou este sonho possível. À minha irmã, pela cumplicidade e por estar sempre ao meu lado, e a todos os meus familiares que acreditaram no meu potencial. Sou imensamente grata também aos amigos que caminham comigo: aos colegas de faculdade, pelo companheirismo diário nesta jornada; aos amigos de longa data, que estão ao meu lado desde o ensino fundamental; e àqueles que, mesmo distantes fisicamente, se fizeram sempre presentes e essenciais. Obrigada pela motivação constante e por permanecerem comigo até hoje. Não citarei nomes, mas vocês sabem quem são. Por fim, agradeço à UNEMAT pela formação de qualidade e pelas oportunidades oferecidas.

Contribuições dos autores

Luanna Bahia da Silva é a principal contribuidora deste trabalho, responsável pela concepção das ideias, formulação dos objetivos, e redação majoritária do manuscrito. Ela também foi a responsável pela metodologia, pelo levantamento dos dados (através dos testes práticos de software) e pelas informações discutidas. Alexandre Berndt supervisionou a pesquisa, forneceu orientação geral para o trabalho e contribuiu com a revisão e edição do manuscrito. Todos os autores leram e aprovaram a versão final do manuscrito.

Conflitos de interesse

Os autores declaram que não têm nenhum conflito de interesses.

Disponibilidade de artefatos de pequisa

Os dados e outros materiais criados e/ou usados nesta revisão da literatura, incluindo as gravações de tela dos testes de acessibilidade executados, estão disponíveis publicamente nos seguintes links:

- Teste de Acessibilidade - Visual Studio Code for the Web Silva [2025c]
- Teste de Acessibilidade - Online-IDE.com Silva [2025a]
- Teste de Acessibilidade - EVI CODE Silva [2025b]

Uso de Inteligência Artificial

Os autores não utilizaram ferramentas de IA generativa no desenvolvimento do estudo.

Referências

ATAG, W. W. W. C. (2015). Authoring tool accessibility guidelines (atag) 2.0. Technical report, W3C. Disponível em: `https://www.w3.org/TR/ATAG20/`. Acesso em: 17 maio 2025.

Baldiris, S., Vargas, D., Garzón, J., Avila-Garzon, C., and Burgos, D. (2022). Evaluation of authoring tools under atag and wcag recommendations. *Universal Access in the Information Society*, 22:919–930. DOI: 10.1007/s10209-022-00904-9.

Behar, P. A., Souza, E. K. d., Góes, C. G. G., and Lima, E. M. d. (2008). A importância da acessibilidade digital na construção de objetos de aprendizagem. *Revista Novas Tecnologias na Educação - Renote*, 6(2). DOI: 10.22456/1679-1916.14459.

Berners-Lee, T. (2005). Accessibility. Disponível em: https://www.w3.org/WAI/fundamentals/accessibility-intro/pt-BR. Acesso em: 12 maio 2025.

Bersch, R. d. C. R. and Pelose, M. B. (2006). Portal de ajudas técnicas para educação: equipamento e material pedagógico para educação, capacitação e recreação da pessoa com deficiência física: tecnologia assistiva: recursos de acessibilidade ao computador ii. Technical report, Secretaria de Educação Especial - ABPEE - MEC: SEESP, Brasília. Disponível em: https://pt.scribd.com/doc/169119771/tecnologia-assistiva. Acesso em: 16 maio 2025.

Brasil (2004). Decreto n. 5.296, de 2 de dezembro de 2004. regulamenta as leis n. 10.048 e 10.098. Diário Oficial da União. Disponível em: https://www.planalto.gov.br/ccivil_03/_Ato2004-2006/2004/Decreto/D5296.htm. Acesso em: 12 maio 2025.

Brasil (2015). Lei n. 13.146, de 6 de julho de 2015. institui a lei brasileira de inclusão da pessoa com deficiência (estatuto da pessoa com deficiência). Diário Oficial da União. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm. Acesso em: 12 maio 2025.

Carletto, A. C. and Cambiaghi, S. (2007). *Desenho Universal: um conceito para todos*. São Paulo, Editora Senac. Disponível em: https://www.maragabrilli.com.br/wp-content/uploads/2016/01/universal_web-1.pdf. Acesso em: 16 maio 2025.

Eclipse (2025). Developer tools ide. Disponível em: https://www.eclipse.org/topics/ide/. Acesso em: 17 maio 2025.

for Developers, G. (2025). Android studio: ambiente de desenvolvimento integrado oficial para android. Disponível em: https://developer.android.com/studio?hl=pt-br. Acesso em: 17 maio 2025.

Intellij (2000). Intellij idea: Features. Disponível em: https://www.jetbrains.com/idea/features/. Acesso em: 17 maio 2025.

Lopes, P. R. d. S. (2023). VerificaÇÃo do nÍvel de acessibilidade de ferramentas para desenvolvimento de software por um desenvolvedor baixa visÃo com a ferramenta de ampliaÇÃo de tela. Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia de Computação. Disponível em: https://repositorio.ufc.br/bitstream/riufc/73711/1/2023_tcc_prdaslopes.pdf. Acesso em: 16 maio 2025.

Mateus, D. A., Souza, M. R. d. A., and Freire, A. P. (2023). Accessibility of mobile apps for visually impaired users: Problems encountered by user evaluation, inspections and automated tools. *Anais do Simpósio Brasileiro sobre Fatores Humanos em Sistemas Computacionais (IHC)*, pages 1–11. DOI: 10.1145/3638067.36381.

Microsoft (2025a). Visual studio code documentation. Disponível em: https://code.visualstudio.com/docs. Acesso em: 17 maio 2025.

Microsoft (2025b). Visual studio code for the web. Technical report, Microsoft. Disponível em: https://vscode.dev/?vscode-lang=pt-br. Acesso em: 16 out. 2025.

Motta, T. C. (2022). *Tecnologias Assistivas*. Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte. Disponível em: https://memoria.ifrn.edu.br/handle/1044/2272. Acesso em: 17 maio 2025.

Nielsen, J. (2024). 10 usability heuristics for user interface design. Technical report, Nielsen Norman Group. Disponível em: https://www.nngroup.com/articles/ten-usability-heuristics. Acesso em: 16 out. 2025.

OneCompiler (2025). Online-ide.com. Technical report, OneCompiler. Disponível em: https://www.online-ide.com. Acesso em: 16 out. 2025.

Overflow, S. (2023). 2023 developer survey. Technical report, Stack Overflow. Disponível em: https://survey.stackoverflow.co/2023/#most-popular-technologies-new-collab-tools. Acesso em: 16 out. 2025.

Patton, M. Q. (2002). *Qualitative Research & Evaluation Methods*. Sage Publications, Thousand Oaks, CA, 3 edition.

Petrausch, V. and Loitsch, C. (2017). Accessibility analysis of the eclipse ide for users with visual impairment. *Karlsruhe Institute of Technology*. DOI: 10.3233/978-1-61499-798-6-922.

RedHat (2023). O que é ide? Disponível em: https://www.redhat.com/pt-br/topics/middleware/what-is-ide. Acesso em: 16 maio 2025.

Sartoretto, M. L. and Bersch, R. (2017). Assistiva tecnologia e educação. Disponível em: https://www.assistiva.com.br/tassistiva.html. Acesso em: 16 maio 2025.

Silva, L. B. d. (2025a). Teste de acessibilidade (tcc): Online-ide.com. YouTube. Disponível em: https://youtu.be/1VDA70McEe4.

Silva, L. B. d. (2025b). Teste de acessibilidade (tcc): Protótipo evi code. YouTube. Disponível em: https://youtu.be/b3y5TC9YGKs.

Silva, L. B. d. (2025c). Teste de acessibilidade (tcc): Visual studio code for the web. YouTube. Disponível em: https://youtu.be/X9rZ81GAcHM.

Sonza, A. P., Kade, A., Façanha, A., Rezende, A. L. A., Nascimento, G. S. d., Rosito, M. C., Bortolini, S., and Fernandes, W. L. (2013). *Acessibilidade e tecnologia assistiva: pensando a inclusãoo sociodigital de pessoas com necessidades especiais*. Séries Novos Autores da Educação Profissional e Tecnológica. Disponível em: https://scholar.google.com/scholar?cluster=9995777659293599641. Acesso em: 16 maio 2025.

UAAG, W. W. W. C. (2015). User agent accessibility guidelines (uaag) 2.0. Technical report, W3C. Disponível em: https://www.w3.org/TR/UAAG20/. Acesso em: 17 maio 2025.

W3C, N. d. I. e. C. d. P. B. (2025a). Sobre o w3c. Disponível em: https://www.w3c.br/sobre-o-w3c/. Acesso em: 13 maio 2025.

W3C, W. W. W. C. (2006). Wai-aria overview.

Disponível em: https://www.w3.org/WAI/standards-guidelines/aria/. Acesso em: 16 maio 2025.

W3C, W. W. W. C. (2025b). Accessibility. Disponível em: https://www.w3.org/mission/accessibility/. Acesso em: 13 maio 2025.

WAI, W. W. W. C. (2025). Web accessibility initiative wai. Disponível em: https://www.w3.org/WAI/. Acesso em: 16 maio 2025.

WAI-ARIA, W. W. W. C. (2017). Accessible rich internet applications (wai-aria) 1.1. Technical report, W3C. Disponível em: https://www.w3.org/TR/wai-aria-1.1/#intro_ria_accessibility. Acesso em: 16 maio 2025.

WCAG, W. W. W. C. (2025). Web content accessibility guidelines (wcag) 2.1. Technical report, W3C. Disponível em: https://www.w3.org/TR/WCAG21/#intro. Acesso em: 15 maio 2025.

WebAIM (2024). Screen reader user survey 10 results. Technical report, WebAIM. Disponível em: https://webaim.org/projects/screenreadersurvey10/. Acesso em: 16 out. 2025.

Yin, R. K. (2001). *Estudo de Caso: Planejamento e Métodos*. Bookman, Porto Alegre, 2 edition. Disponível em: http://maratavarespsictics.pbworks.com/w/file/fetch/74304716/3-YIN-planejamento_metodologia.pdf. Acesso em: 16 out. 2025.

Zen, E. and Tavares, T. A. (2023). Estratégias de acessibilidade em ides para estudantes com deficiência visual. *Congresso Brasileiro de Informática na Educação*. DOI: 10.5753/cbie_estendido.2023.234323.