Revista Brasileira de Informática na Educação – RBIE
Brazilian Journal of Computers in Education
(ISSN online: 2317-6121; print: 1414-5685)
https://sol.sbc.org.br/journals/index.php/rbie

Submission: 21/06/2024; 1st round notif.: 13/06/2025; New version: 20/08/2025; 2nd round notif.: 02/10/2025;
Camera ready: 07/10/2025; Edition review: 15/10/2025; Available online: 15/10/2025; Published: 15/10/2025;

# Development and Accessibility Automatic Assessment of ADA Blocks: A Virtual Assistant that Supports the Choice of Block Programming Tools

*Ana Paula Juliana Perin*
*Federal University of Paraná*
*ORCID: 0000-0001-6470-681X*
*apjperin@inf.ufpr.br*

*Deivid Eive Santos Silva*
*Federal University of Western Pará*
*ORCID: 0000-0003-1066-0750*
*deivid.silva@ufopa.edu.br*

*Natasha M. C. Valentim*
*Federal University of Paraná*
*ORCID: 0000-0002-6027-3452*
*natasha@inf.ufpr.br*

## Abstract

*Currently, personalized learning is sought to encourage 21st Century skills and competencies in students, such as logical reasoning and computational thinking. Block programming can support this process. However, it was noticed that teachers, such as those in high school, have difficulties in choosing programming tools depending on their context of use. Therefore, the ADA Blocks virtual assistant was developed to assist teachers in choosing block programming tools. The ADA Blocks development methodology includes systematic mapping study, preliminary and feasibility studies, and accessibility automatic evaluation. As a result of the feasibility study, the usefulness, ease of use, and limitations of ADA Blocks were identified. The automated accessibility assessment, performed with tools such as AccessMonitor and ASES, identified technical and content issues. Based on these findings, the assistant was improved, with adjustments such as hierarchical text organization and the insertion of textual references in images for assistive technologies ('alt' attribute). After the improvements, a new automatic evaluation showed a significant increase in accessibility scores, rising from 61.57% to 88.41% on ASES and from 5.2 to 6.9 on AccessMonitor. Finally, ADA Blocks demonstrated positive accessibility results in its new version. This work fills a gap in the literature, being the first to evaluate the accessibility of virtual assistants for recommending block programming tools. Future prospects include evaluation with users with disabilities and experts, eventually identifying usability issues and refining the assistant based on user feedback.*

***Keywords:*** *ADA Blocks; Virtual Assistant; Accessibility; Automatic Evaluators; Block Programming.*

*Cite as: Perin, A. P. J., Silva, D. E. S., & Valentim, N. M. C. (2025). Development and Accessibility Automatic Assessment of ADA Blocks: A Virtual Assistant that Supports the Choice of Block Programming Tools. Brazilian Journal of Computers in Education, vol. 33, 1448–1483. https://doi.org/10.5753//RBIE.2025.4630.*

## 1 Introduction

Constant transformations in society, such as the introduction of computer systems, bring educational challenges, especially in the use of Digital Information and Communication Technologies (DICTs). DICTs, such as block programming tools, digital games, Virtual Assistants, among others, enable teaching and learning processes aimed at innovation through individual and collective experimentation. Furthermore, they allow individuals to work across diverse areas (Cabral et al., 2019), as proposed in STEM (Science, Technology, Engineering, and Mathematics) (Lee & Malyn-Smith, 2020).

Students must develop skills and competencies to face the challenges of the 21st Century, including the ability to deal with technological resources and processes (Himmetoğlu et al., 2020; D. E. Silva et al., 2020). Competencies are understood as a set of skills and knowledge related to each other and that can be developed through practice or experience. Concerning skills, these are understood as qualities that the student needs to develop to carry out some activity (BNCC, 2018).

According to the World Economic Forum's 2020 The Future of Jobs report, the primary skills that will be most valued in the job market by 2025 include Analytical and innovative thinking; Active learning and learning strategies; Solving complex problems; Critical and analytical thinking; Leadership and social influence; Use of technology for monitoring and control; Technology design and programming; Resilience, stress tolerance, and flexibility; Reasoning, problem-solving, and ideation. In addition to these skills, (Führ & Haubenthal, 2019) also emphasize the importance of developing the following skills and competencies in students: Mastery of technology; Leadership; Collaboration; Good communication; Emotional intelligence; Autonomy; and Teamwork.

One way to prepare students for the challenges of the 21st Century is through block programming, which makes programming more attractive and intuitive. Furthermore, block programming can be used in an interdisciplinary way, working on concepts related to logic and computational thinking. Block programming tools are software that allow users to create programs by dragging and dropping pre-made blocks of code.

In this sense, the Brazilian Computing Society (Sociedade Brasileira de Computação – SBC) contributed to including block programming tools and other DICTs in Basic Education in conjunction with the guidelines of the Brazilian National Common Curricular Base (Base Nacional Comum Curricular – BNCC) (BNCC, 2018). The SBC emphasizes the need to teach computing in schools as a way to prepare citizens with essential knowledge and skills for life in the 21st Century (SBC, 2018).

However, the inclusion of block programming in high school can face significant challenges, as many teachers do not know how to choose which programming tool to adopt in their subject. Therefore, for this purpose, the ADA Blocks virtual assistant was developed (Perin, Silva, & Valentim, 2022). The construction of ADA Blocks involved nine distinct stages. Some of the steps are: a Systematic Mapping Study (SMS) was carried out to investigate which DICTs support the teaching of programming and/or computational thinking through block programming; opinion survey was carried out with high school teachers, focused on the use of block programming tools as support material for teaching and learning processes; a benchmark of the available block programming tools and feasibility study of the ADA Blocks virtual assistant with high school teachers. Based on the results obtained in this study, the ADA Blocks assistant has been improved. Posteriorly, an ADA Blocks Accessibility Automatic Assessment was conducted. Finally, improvements were made to ADA accessibility based on the results of the accessibility automatic evaluators.

The ADA Blocks assistant has a set of questions that cover various aspects such as support material, language, discipline, among other relevant characteristics. These questions facilitate the process of suggesting the most appropriate block programming tools for the context of each teacher's discipline. To this end, an analysis of block programming tools intended for use by students was conducted (Perin, Silva, & Valentim, 2021). As a result, a knowledge base of block programming tools was identified and categorized according to their characteristics, such as platforms, operating systems, and support material. To obtain a tool suggested by ADA Blocks, the teacher must answer a questionnaire that addresses the characteristics of the tools and the context of use. Therefore, the tools suggested by ADA Blocks and the knowledge base presented (Perin, Silva, & Valentim, 2022) can support the teacher in selecting the most appropriate tool for the context of their discipline. This type of tool can assisting in adopting personalized approaches in the use of block programming tools (Cárdenas-Cobo et al., 2024). Thus contributing to the development of essential skills and competencies for life in the 21st Century of its students. Although ADA Blocks was developed for high school teachers, in the ADA Blocks feasibility study, it was realized that it can also serve other levels of education, such as Elementary Education and Higher Education, mainly for new students in programming.

Still in the ADA Blocks feasibility study, one of the participants highlighted that ADA Blocks does not provide accessibility for blind teachers or those with low vision compared to reviews of Assistive Technology resources, such as described audio, despite the assistant already having accessibility resources for the deaf (Perin, Silva, & Valentim, 2022). Based on this feedback from participants, the idea was to evaluate ADA Blocks through accessibility automatic evaluators to identify other accessibility problems to make them accessible to all users, regardless of their specific needs, adhering to the Web Content Accessibility Guidelines (WCAG) (W3C, 2008) and the guidelines from the Electronic Government Accessibility Model (eMAG) (eMAG, 2014). Conducting this type of study with ADA Blocks is an important step in identifying which aspects of accessibility need the most attention, and which help ensure that the assistant provides an inclusive and accessible experience for all teachers. This study's initial go was to automatically assess the compliance of the ADA Blocks virtual assistant with WCAG/eMAG guidelines using automated evaluators. The results allowed for the implementation of accessibility improvements based on identified nonconformities and the verification of the effectiveness of these corrections through automated reassessment—the process and results of which are detailed in this paper. Furthermore, this work fills a gap in the literature, being the first study focused on the accessibility assessment of virtual assistants for recommending block programming tools.

In general, ADA Blocks main contribution to the Computing in Education community is that through it, teachers can explore an interdisciplinary approach, integrating Computing into various subjects in the school curriculum. Additionally, ADA Blocks recommendations enable the use of exercise recommendation systems, such as CARAMBA (Cárdenas-Cobo et al., 2024), or systems integrated with block programming environments, such as Beecrowd (included in the ADA Blocks database and recommended by it), which has a problem repository and an automatic assessment system (Zaffalon et al., 2022). Thus, tool recommendations by ADA Blocks aligned with adapted exercise recommendation tools can intensify the personalization of the teaching and learning process.

This allows students to develop essential 21st Century skills and competencies such as computational thinking, problem solving, creativity and collaboration. Furthermore, ADA Blocks promotes a broader view of computing, not just limited to the technical aspect, but also encouraging the application of technology in different contexts and disciplines. This reflects the importance of approaching information technology as a transversal tool, capable of enriching learning in different areas of knowledge. This contribution aligns with BNCC and SBC proposal to promote interdisciplinarity through IT in Education. Therefore, making this assistant accessible is crucial for a better user experience and for a diversity of teachers.

This article is organized as follows: Section 2 presents the theoretical foundation. Section 3 presents related work. Section 4 presents the methodology. Section 5 presents planning, execution and analysis of the results of automatic accessibility assessments carried out in ADA Blocks. In Section 6, the accessibility improvements implemented in ADA Blocks after the accessibility automatic assessments are described. Finally, Section 7 presents final considerations and future perspectives.

## 2 Theoretical Foundation

### 2.1 Block programming tools

Textual programming can be seen as a problem for students who are having their first contact with programming (Hudin & Adii, 2024; Medeiros et al., 2019; M. V. R. Souza & França, 2013; Yu et al., 2025). In this context, the use of textual programming languages can make the learning process a little more difficult due to the complexity of the syntax of these programming languages (Burnett & McIntyre, 1995; Hudin & Adii, 2024; Medeiros et al., 2019). Complexity leads researchers and professionals involved in teaching programming to think in new ways to facilitate learning. Therefore, to minimize these difficulties, Block-based visual programming (BVP) environments were designed and developed, giving rise to block programming environments (Yu et al., 2025).Block programming environments are made up of specific characteristics and colors that indicate their function (commands and values). This union of characteristics and colors makes it possible to form the structure of the programs. These commands and values include plug-in blocks, allowing the programming action to be less complex and enabling other audiences, such as teachers and basic education students, to have experience with programming (Rios et al., 2019), and there are no problems with syntax (Cárdenas-Cobo et al., 2024).

It is believed that block programming environments can make programming, naturally related to Mathematics and Logic concepts, more attractive through a more intuitive and visual experience. Furthermore, block programming environments, such as Scratch, significantly increase academic performance and socio-emotional skills (Cárdenas-Cobo et al., 2024), corroborating their interdisciplinary effectiveness. Another important concept that can be related to block programming is computational thinking, characterized by a set of skills to solve problems, design systems and understand human behavior, being based on concepts from Computer Science (Wing, 2006). Computational thinking is one of the skills of the 21st Century, highlighting its need to be worked on in basic education, as it helps students understand computational concepts and develop other skills such as autonomy and logical reasoning (Guggemos et al., 2019; Papadakis & Orfanakis, 2018; Vinayakumar et al., 2018; Yett et al., 2020). The teaching of computational thinking can be inserted in high school through the introduction of programming languages in blocks (Hudin & Adii, 2024; Vinayakumar et al., 2018). Block programming can also help improve computational thinking, critical thinking, creativity, teamwork, resilience, among others. Furthermore, block programming combined with Educational Robotics helps high school students learn computational concepts, keeping them engaged (Yett et al., 2020).

In this way, block programming encourages digital culture at school. Furthermore, it was realized that it is possible to make use of block programming with emerging technologies such as: a) Robotics through Lego[1]; b) IoT mediated by Scratch for Arduino (S4A[2]); and, c) 3D modeling and simulation and/or digital games using the ENGAGE[3] tool, among others. In this way, when making use of block programming and emerging technologies in the classroom, the teacher will also benefit from the teaching and learning processes by learning about the features of the block programming tool they want to work on, through the exchange of experiences with other teachers, and the exchange of experiences with students (Haduong & Brennan, 2019). In short, the teacher will be able to encourage the development of skills in students, through practical activities.

Therefore, block programming tools must be accessible to people with different types of disabilities, ensuring that all students have equal access and opportunity to learn programming. The accessibility of these tools not only promotes the inclusion of students with disabilities, but also strengthens diversity and equity in the educational environment.

By making the interfaces and functionalities of block programming tools accessible, it would be possible to develop 21st Century skills and competencies in a greater number of students, regardless of their physical or cognitive limitations. In the next section, we will explore some accessibility concepts, highlighting strategies that can be adopted to ensure accessibility on websites.

### 2.2 Accessibility

Accessibility, in general, refers to the ability of individuals with disabilities to use a system, application, or website effectively. Accessibility can be understood as the adaptation of usability to suit a specific group of users, and, in this context, metrics such as task completion rates and times, as well as self-reported metrics, can be employed to measure the usability of any system for users with different types of disabilities (Albert & Tullis, 2022).

According to IBGE data from 2010, around 46 million Brazilians, equivalent to approximately 24% of the population, said they faced some degree of difficulty (such as seeing, hearing, walking or climbing steps) or had some intellectual disability. Visual impairment is the most common, followed by physical impairment, then hearing impairment and, finally, intellectual disability (IBGE, 2010).

[1]https://makecode.mindstorms.com/#editor
[2]http://s4a.cat/index_pt.html
[3]https://intellimedia.ncsu.edu/engage/

In 2022 no Brazil, approximately 18.6 million individuals aged two years or older had some form of disability. Additionally, it was found that 25.6% of people with disabilities had completed at least high school, while this rate was 57.3% for people without disabilities. In the third quarter of 2022, the illiteracy rate among people with disabilities reached 19.5%, in contrast to the 4.1% rate observed among people without disabilities (IBGE, 2022). These data highlight the accentuation of inequalities faced by people with disabilities. This demonstrates the need for educational initiatives that include these people, in order to ensure that they acquire the skills and competencies necessary for the 21st Century. Furthermore, they demonstrate the need for Web pages to be accessible and suitable to serve all users.

In this sense, on December 2, 2004, in Brazil, standards and criteria were established that aim to promote accessibility for people with disabilities or reduced mobility through Decree nº. 5,296. The goal is to ensure that public administration electronic portals and websites are accessible, seeking to ensure that people with disabilities have access to information (Brasil, 2004). Following the decree, the Brazilian government created eMAG to guide the development of adaptation of content on federal government websites, thus facilitating access to available information and services (Lemos et al., 2019). eMAG is an Accessibility Model in electronic Government, prepared by the Department of Electronic Government in partnership with the *Acessibilidade Brasil* Non-Governmental Organizations. The eMAG consists of an adaptation of the WCAG to the reality of the Brazilian government. The eMAG recommendations aim to standardize the implementation of digital accessibility to Brazil's needs, also meeting international recommendations, becoming mandatory on Brazilian government websites and portals in 2007 (eMAG, 2014).

Therefore, to ensure real inclusion on the Internet, there is a clear need for web pages to be developed with an emphasis on accessibility. When taking into account the general aspects of usability, it becomes clear that accessibility allows everyone to use an interface effectively, efficiently and satisfactorily (ISO, 2003). There are several approaches to evaluating the accessibility of web pages, one of which is the use of automatic tools, such as the Accessibility Evaluator and Simulator in Sites (Avaliador e Simulador de Acessibilidade em Sítios – ASES). This type of tool scans digital content for violations of web accessibility guidelines, generally based on WCAG.

WCAG is a set of guidelines developed by the World Wide Web Consortium (W3C) to ensure that web content is accessible to people with different types of disabilities. These guidelines provide technical recommendations for making web content more accessible, addressing areas such as perception, operability, and comprehension for different groups of users, including those with visual, hearing, motor, cognitive, or other disabilities (W3C, 2008).

In this sense, for a web page to comply with WCAG, all compliance criteria must be met. For each criterion, specific techniques are provided with examples of how the objective of the criterion can be achieved and tested. The criteria are indicated by three levels of conformity (W3C, 2008), with Level A: most significant accessibility barriers. Complying with level A criteria alone does not guarantee a highly accessible site; Level AA: meeting all level AA success criteria guarantees a very accessible website, that is, the website will be accessible to most users, under most circumstances, through the use of most technologies; and Level AAA: the triple A compliance level is quite meticulous, that is, it aims to guarantee an optimized level of accessibility. At this level, most success criteria refer to very specific situations, usually aiming to refine the AA level criteria. Maintaining compliance with certain AAA criteria can be a costly and sometimes difficult process to implement.

Complying with WCAG enables the universalization of online resources, and brings social, economic and psychological benefits. Social, as it promotes the inclusion of users in activities such as work, study, social interaction and entertainment. Economical, as it integrates these people as potential new customers in different sectors. And Psychological, as it tends to raise self-esteem and reduce the feeling of powerlessness in front of others (E. R. Souza & Mont'Alvão, 2012). Therefore, an accessible web page is one that minimizes the impediments that result in limitations of access to information, making it essential to carry out accessibility assessments to ensure that websites are in fact accessible to their users (Nielsen, 2007).

## 3 Related Work

### 3.1 Block programming

Jocius et al. (2020) worked on training 116 teachers with the Snap![4] tool. The teachers were from the Humanities, Science and Mathematics disciplines. The training began through PRADA (Pattern Recognition, Abstraction, Decomposition, and Algorithms), presenting computational thinking, followed by code infusion sessions using the Use-Modify-Create learning framework, allowing the use, modification and creation of new codes during the learning process. Teachers created a lesson plan for their subject and suggested activities that could be implemented in the classroom. In addition, they carried out a collaborative activity to map and describe the patterns of PRADA elements and created teaching materials, such as slides and handouts, to present what they learned to participants.

Ferreira and Sant'Ana (2025) describes the experience of an online mini-course offered by the Mathematics Education Study Group (Grupo de Estudos em Educação Matemática – GEEM) at the Southwest Bahia State University (Universidade Estadual do Sudoeste da Bahia – UESB) in the second half of 2022. Aiming to train teachers and future Mathematics teachers in the creation of educational digital games, the initiative used the Scratch programming language as its primary tool, targeting pedagogical applications in the classroom. Six students—four undergraduates and two practicing teachers—participated in a 30-hour program spread across 13 synchronous meetings (via Google Meet) and asynchronous activities (in Google Classroom). The students rated the experience as enriching for their training, highlighting plans to apply the games to their teaching practices. The study concludes that game production in Scratch is a viable strategy for pedagogical innovation in Mathematics, but highlights the need for ongoing technical support and adaptation to school realities.

Finally, in the study by Hyury and Jailton (2025) describes a study conducted with 10 Technical High School teachers at a Brazilian public institution, focused on training them to apply computational thinking (CT) in the classroom using Scratch. The study included 10 teachers, who taught courses such as Portuguese, Mathematics, Physics, Chemistry, Biology, Nursing, Business Administration, and Marketing, with between 3 and 20 years of teaching experience (average, 10 years). None had a computer science background, and only 20% were familiar with Scratch before the training. The training consisted of ten in-person meetings (totaling 20 hours), structured progressively: it began with theoretical concepts of CT (decomposition, pattern recognition, ab-

[4]https://snap.berkeley.edu

straction, and algorithms), progressed to practical exercises in Scratch, and the development of interdisciplinary projects aligned with BNCC competencies. The training demonstrated that Scratch is an effective tool for introducing CT in an accessible and interdisciplinary manner, transforming pedagogical practices. The results reinforce the need for ongoing training, investment in technological infrastructure in schools, and institutional policies that support the integration of CT into the curriculum.

The above studies using block programming tools in teacher training show positive results, but they have significant limitations that reinforce the relevance of an assistant like ADA Blocks. All studies highlight the viability of block programming tools (Snap!, Scratch) as accessible environments for teachers without a computer science background. The "Use-Modify-Create" structure and the progressive focus (theory, practice, projects) proved effective. However, a limitation lies in the need for ongoing support from researchers during the classroom implementation process. This indicates that initial usability is good, but sustainability and autonomy in daily use are challenging. Teachers demonstrated high motivation when creating lesson plans and materials specific to their subjects, reporting professional enrichment and an intention to apply their knowledge (as evidenced by Jocius et al. (2020), Ferreira and Sant'Ana (2025) and Hyury and Jailton (2025)). However, motivation appears to be intrinsically linked to the training context and support provided, raising questions about its maintenance when this external support is withdrawn. Given these limitations, ADA Blocks emerges as a potentially transformative solution, offering autonomy in choosing block programming tools and ongoing contextualized support (via links and support materials). ADA Blocks not only promotes tool adoption and maintains teacher motivation (overcoming dependence on external support), but it can also optimize the learning process. This is because the selected tool will be more aligned with specific pedagogical objectives, ensuring more effective application in the classroom.

### 3.2 Accessibility Automatic Assessment

In the study carried out by Pivetta et al. (2014), accessibility in Virtual Teaching and Learning Environments (VTLE) for deaf users was evaluated in a hybrid way (automatic and human evaluators). The VTLE used in the study was Moodle, and automatic assessment was conducted using AChecker (Accessibility Checker), TAW and WAAT (Web Accessibility Assessment Tool). The results of the accessibility analysis performed by the automatic evaluators highlight the importance of appropriate labels and titles, as indicated by your metrics, to ensure that page elements are functionally appropriate and understandable. However, even though WCAG guidelines address access to information and attempt to validate aspects such as subtitles, there is a lack of support for videos in sign language by validators, leaving crucial elements of communication open for deaf people who use this language.

In the study by E. H. Silva (2021), they sought to evaluate the VLE-DE (Virtual Learning Environment for Distance Education) of the Federal Institute of Education, Science and Technology of Pernambuco (IFPE). The research method adopted was a qualitative and quantitative approach, with an exploratory-descriptive character, using eMAG as a reference, together with the ASES and AccessMonitor tools. The authors identified accessibility issues in the Marking (HTML) and Content/Information sections. Regarding problems in the Marking section, 56 errors and 199 warnings were found. The most frequent errors were: Respecting Web Standards and

Organizing the HTML code in a logical and semantic way. As for the warnings, they stood out mainly in the areas of “Respect Web Standards” regarding the logical and semantic organization of the HTML code, and in the issue of not opening new instances without the user’s request. Regarding accessibility problems in the Content/Information Section, 47 errors and 8 warnings were found. Key issues identified include the need to clearly and succinctly describe links and provide textual alternatives to images on the page. As for the warnings, those related to the clear and succinct description of the links stood out. Finally, the authors highlighted the need for improvements and adoption of programming practices that aim to avoid or reduce accessibility flaws in a web system.

Viana et al. (2017) evaluated accessibility in the SOLAR Virtual Learning Environment, which supports academic activities at the Open University of Brazil and the Federal University of Ceará. In the context of the study, the authors focused on the use of SOLAR by students who are blind or have low vision. Thus, the accessibility assessment took place in the hybrid modality. The first part is the use of automatic evaluators. The second part consists of the evaluation of SOLAR by two users with low vision. As a result of the automatic evaluation, approximately 90.7% of accessibility errors in SOLAR were classified as critical, while 9.3% were classified as serious. As a result of the manual evaluation, users used the NonVisual Desktop Access (NVDA) screen reader. When users were unable to use the screen reader, they used the browser’s font enlargement feature by 500%. Users had difficulties navigating the login form because the description of some figures was confusing with the description of the form. Therefore, users suggested changes to the ‘alt’ attribute, used to describe images.

Did not identify studies in the literature that address the accessibility assessment of virtual assistants, using automatic evaluators. Although ADA Blocks is not a VLE, it can contribute to the teaching and learning process by recommending block programming tools to teachers based on their context of use. Therefore, it is important to evaluate accessibility aspects in the ADA Blocks virtual assistant, in order to identify possible problems that do not comply with WCAG and eMAG guidelines, thus enabling improvements to the assistant. The analysis of studies on automatic accessibility assessment (Pivetta et al. (2014), E. H. Silva (2021), Viana et al. (2017)) reveal recurring accessibility problems on web pages. Some problems mentioned were: a) unclear, not succinct or missing description of page elements (mentioned by E. H. Silva (2021) and Viana et al. (2017)); b) labels missing or not correctly associated with fields, causing confusion, especially for screen reader users (evidenced by Viana et al. (2017); c) poor structuring of code semantics through incorrect use of HTML elements, lack of heading hierarchy (h1-h6), and logical organization, making navigation and understanding difficult for assistive technologies (strongly mentioned by E. H. Silva (2021) as "Respect Web Standards" and "Organize HTML code logically and semantically"); d) Lack of support for full keyboard navigation (not explicitly detailed, but implicit in the need for logical organization for sequential navigation); and e) Difficulties with screen readers (NVDA) and screen magnifiers (mentioned by Viana et al. (2017)), in addition to the lack of support for Libras videos in the automatic validators (mentioned by Pivetta et al. (2014)). Some of these issues were also identified in the automatic evaluation of ADA Blocks. This persistence can often be due to programmers’ lack of knowledge or negligence regarding accessibility guidelines. Furthermore, studies show that, although automatic validators (such as ASES and AccessMonitor) are essential for identifying technical issues, they do not fully capture the experience of users with disabilities. Therefore, the next step will consist of conducting manual evaluation with users and experts.

# 4 Methodology

The ADA Blocks virtual assistant was developed with the aim of providing information about the different block programming tools available, their characteristics, disciplines that can be worked on, whether support material is available, among other characteristics. The construction of ADA Blocks followed an evidence-based methodology and consisted of 09 steps (Figure 1), which are: (1) an opinion survey carried out with high school teachers on the use of DICTs in teaching processes and learning during the remote period; (2) a Systematic Mapping Study (SMS) that investigated which DICTs support the teaching of programming and/or computational thinking through block programming; (3) an opinion survey with high school teachers on the use of block programming tools as support material for teaching and learning processes; (4) a benchmark of block programming tools; (5) the implementation of the ADA Blocks virtual assistant; (6) a feasibility study of the ADA Blocks virtual assistant carried out with high school teachers; (7) evolution of the ADA Blocks assistant based on the results obtained in the feasibility study; (8) accessibility assessment with automatic evaluators; and (9) evolution of the ADA Blocks assistant based on the results obtained in the accessibility automatic assessment. The results of these steps will be presented below.

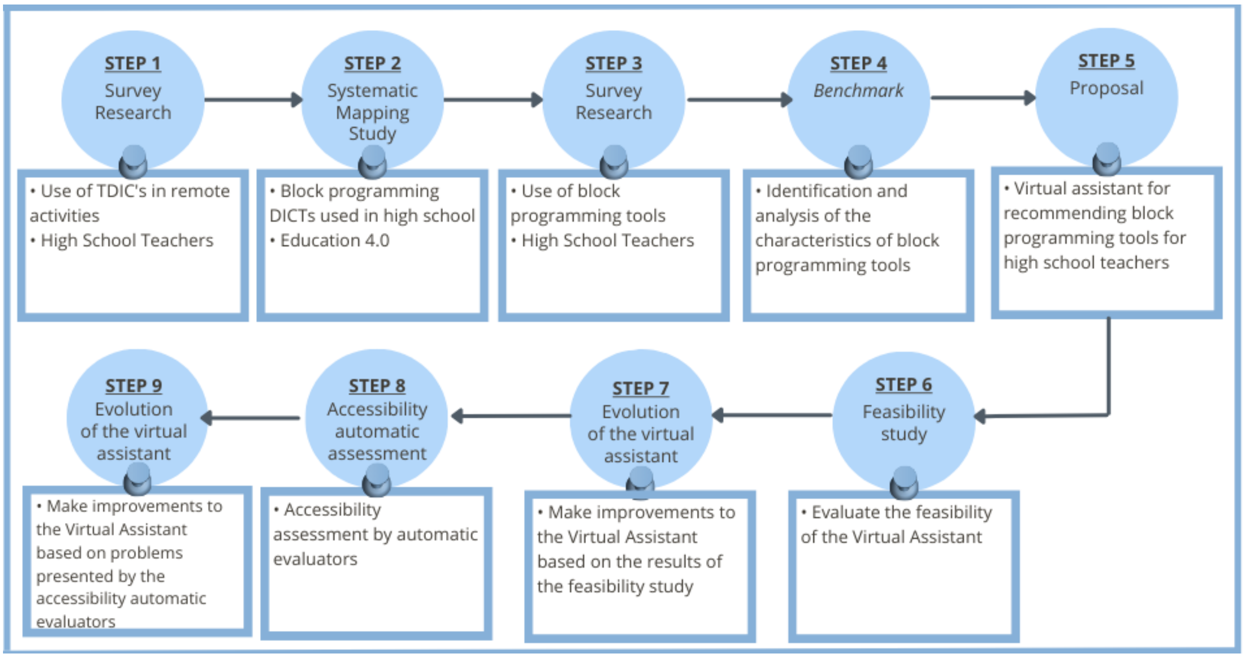


Figure 1: ADA Blocks development methodology..

## 4.1 Survey on the use of DICTs in remote teaching by teachers

This survey was carried out using a questionnaire that received 255 responses from high school teachers. Teachers at this level were chosen for this study due to the ease of participation during the pandemic, as students would need parental consent, and social distancing would make this practice more difficult. In this context, schools were forced to migrate to remote learning, and teachers were challenged to use DICTs in teaching and learning processes. Therefore, the objective of this survey was to identify the difficulties teachers faced in using DICTs. The survey was shared using social media and messaging apps, and monitored for fourteen days.

The questionnaire was organized and structured into three parts: 1) Teacher characterization data, 2) Teacher experience with remote activities, and 3) Teacher perception of students in remote activities. The responses were individually analyzed and peer-reviewed. The full results and detailed analysis of this study were published in the 12th Computer on the Beach 2021 (Perin, Silva, & Valentim, 2021).

### 4.2 Systematic Mapping Study

The main question of this SMS was: "What are the DICTs that support teaching and learning processes and/or computational thinking through block programming in high school?". The SMS was carried out as described by Kitchenham and Charters (2007). The goal of the SMS was defined according to the GQM paradigm (Goal-Question-Metric), proposed by Basili and Rombach (1988), being: **Analyze** Scientific Publications; **With the purpose of** Characterizing; **In relation to** DICTs applied in High School to teach programming and/or computational thinking through block programming; **From the point of view of** Informatics in Education and Computing Education Researchers; **In the context of** Primary sources available in the SCOPUS, ACM, IEEEXplore and SPRINGERLINK search engines.

To help answer the main question, eight research sub-questions (SQs) were defined. The subquestions are: **SQ1.** "Was block programming used at school?"; **SQ1.1.** "In which learning space is block programming being used to support Education?"; **SQ2.** "Who is responsible for teaching block programming?"; **SQ3.** "Was there training in using the block programming tool? If yes, who received training?"; **SQ4.** "To which target audience and high school grade are block programming tools being used?"; **SQ5.** "In which class were block programming activities carried out?"; **SQ6.** "What tools support Education in High School?"; **SQ6.1.** "What are the emerging technologies can be used to support Education?".

The string search used was: *"high school" OR "senior high" OR "K-10" OR "K-11" OR "K-12"* AND *"block programming" OR "block-based programming" OR "block-based coding" OR "block interface" OR "block-based tool" OR "block-based platform" OR "block-based language" OR "block-based approach" OR "block -based methodology" OR "block-based process" OR "visual block programming"* AND *"e-learning" OR "active learning" OR "Education 4.0" OR "blended learning" OR "computational thinking".* When running *string* in search engines, 507 studies were returned. In the first filter, 239 studies were selected by reading the title and *abstract*. In the second filter, 46 studies were selected by reading the entire article.

Each article selected for this SMS was independently reviewed and evaluated by two researchers (the author of this research and advisor) and one researcher (co-advisor), all with experience in research on Computer Science in Education. The researchers decided whether or not the publication should be included based on a set of criteria divided into (1) Inclusion Criteria (IC) and Exclusion Criteria (EC). Criteria for inclusion of articles: **IC1.** Publications on ICTs that support block programming and/or computational thinking through block programming in the context of Education 4.0 in high school; **IC2.** Publications describing experimental studies on the use of DTICs that support block programming and/or computational thinking through block programming in the context of Education 4.0 in high school. Criteria for exclusion of articles:

**EC1.** Publications that did not meet the inclusion criteria were not selected; **CE2.** Publications not in the defined languages (English and Portuguese) were not selected; **CE3.** Publications not available for consultation or download in open format, meaning that the information was free of charge, were not selected; **CE4.** Duplicate publications were not selected; **CE5.** Publications not peer-reviewed (gray literature) were not selected.

To conduct the mapping, the selection of primary studies was carried out in two stages (first filter and second filter). In the first filter, two researchers and one researcher read the title and abstract. A justification was provided for each article excluded in this stage. In the second filter, the articles that passed the first filter were read in full. For article selection, both stages underwent the same processes: 1) The researchers analyzed the inclusion and exclusion criteria and recorded the results obtained; 2) The researchers reached consensus when there was no unanimity on the inclusion or exclusion of a publication; 3) In the consensus stage, in case of disagreement on the inclusion of a publication, the study was included for the next stage. The full results and detailed analysis of this study were published in the Informatics in Education journal in 2023 (Perin, Silva, & Valentim, 2023).

### 4.3 Survey on the use of block programming tools

In this survey, 36 Brazilian high school teachers participated who have some type of experience with block programming tools. In this survey, we sought to understand how and which block programming tools are being used by Brazilian high school teachers. The survey was shared via social media such as LinkedIn, messaging apps such as WhatsApp and Telegram, and institutional email. The data collection period totaled twenty-six days. The questionnaire contained ten multiple-choice questions about teachers' experience in carrying out activities with block programming tools and two questions to be answered in an open format. To analyze the data obtained in these open questions, open coding and axial coding were used, a subset of the procedures of the Grounded Theory (GT) method (Corbin & Strauss, 2014). The first step was open coding, where the data were coded according to each participant's response to each question. Subsequently, in axial coding, the codes were grouped according to their properties and related to each other, thus forming categories that represent their characteristics. The third step (selective coding) was not performed, as the goal was not to create a theory. The open and axial coding steps were sufficient for data analysis. The responses were individually analyzed and peer-reviewed. The full results and detailed analysis of this study were published in the XVIII Brazilian Symposium on Information Systems (SBSI) in 2022 (Perin, Silva, & Valentim, 2022).

### 4.4 Benchmark

The motivation arose from the need to search for other block programming tools, and to identify and characterize these tools, because in SMS a diversity of block programming environments and their functionalities were observed. Furthermore, from the second survey carried out with high school teachers (Subsection 4.3) the need arose to identify which subjects at this level of education the tools can be worked on, which is one of the characteristics investigated in benchmark. Investigating the different tools available and their categorization can help teachers choose the most appropriate tool for their subject and context of use.

In benchmark, five characteristics of block programming tools were analyzed: (1) Platforms on which block programming tools work; (2) Operating System; (3) Support material for the teacher and/or student and what type of material is available; (4) Emerging Technologies that can be used in conjunction with block programming tools; and (5) Disciplines in which the tools can be worked. In total, 58 tools were identified and had link access, which allowed the analysis of the characteristics mentioned above.

The strategies for Executing the benchmark and searching for tools were: a) Identifying the tools returned in the SMS; and b) Manually searching the web using keywords in English such as "block programming", "block-based programming", "block-based coding", "block interface", "block-based tool, "block-based platform", "block-based language", "block-based approach", "block-based methodology", "block-based process", and "visual block programming", and the same terms in Portuguese. Finally, in the Analysis step, the identified tools were individually analyzed regarding their characteristics and peer-reviewed. The full results and detailed analysis of this study were published in the XXXII Brazilian Symposium on Informatics in Education (SBIE) in 2021 (Perin, Silva, & Valentim, 2021).

### 4.5 Initial version of the ADA Blocks Virtual Assistant

The ADA Blocks[5] virtual assistant was built based on surveys (Subsections 4.1 and 4.3), SMS (Subsection 4.2) and benchmark (Subsection 4.4). It consists of a recommendation technology for a set of 58 block programming tools for high school teachers.

The recommendation questionnaire has ten questions: (1) In which discipline do you want to use the block programming tool?; (2) Do you want to use any support material?; (3) Who will use the support material?; (4) What type of support material do you want to use?; (5) Which platform do you want to use the block programming tool?; (6) Which operating system for mobile do you want to install the tool?; (7) Which desktop operating system do you want to install the tool?; (8) Do you want to use a block programming tool in conjunction with another technology? For example, Robotics, IoT, among others; (9) What technology do you want to work with in combination with block programming?; and (10) Which block programming tool language do you want to use? Among the answers, disciplines, platforms, operating systems and emerging technologies can be found.

The first version of ADA Blocks contained the following navigation paths: (1) **“Home page”**, where there is an interaction between ADA and the user that occurs through a static image, which includes a greeting, an explanation of the purpose of the virtual assistant and navigation guidance; (2) **“Frequently asked questions: ADA Answers!”**, presents six topics that the teacher may not know or have doubts about: “What is block programming and what is its importance?”, “What is block programming? What are block programming environments?”, “Why block programming?”, “Why use block programming in the classroom?”, “What is Education 4.0?” and “What is the relationship between block programming and Education 4.0?”; (3) **“Choosing Block Programming Tools”**, presents the block programming tool recommendation questionnaire, in addition to a static image of ADA’s interaction with the user; (4) **“Programming tools in blocks by category”**, presents a repository with the tools and their characteristics, available for free consultation, according to each category. When selecting a category, subcategories are displayed that

[5]https://adablocks.com.br/

can be chosen to consult with a summary of related tools. There is also an option to consult all tools, regardless of category; (5) **"Meet the authors"**, presents the authors of ADA Blocks and ways to contact them to suggest improvements or clarify possible doubts. After developing ADA Blocks, a feasibility study was conducted, which is detailed in the following subsection.

### 4.6 Feasibility Study

The feasibility study was carried out with 13 high school teachers (Regular and Professional/Technological) who teach common core subjects in public and private schools. The study was approved by the Research Ethics Committee (CEP)[6] and The full results and detailed analysis of this study were published in the XXXIII Brazilian Symposium on Informatics in Education – SBIE 2022 (Perin, Silva, & Valentim, 2022).

### 4.7 Version 2 of ADA Blocks

As a result of the feasibility study, a change was made to the home page, replacing the static image with a GIF. This change made it possible to simulate a conversation, and provide a better presentation and interaction of ADA. Furthermore, changes were made to the recommendation questionnaire: (1) replacement of the initial image with a video (Figure 2), in order to improve interaction and show teachers how to navigate the questionnaire; (2) multiple choice question replacement for checkboxes. With this flexibility in the questionnaire, the number of suggested tools and their choice possibilities increased; (3) addition of the response option "I do not wish to inform" to improve the usefulness of the assistant for teachers from other areas and teachers of subjects or courses related to training itineraries (new BNCC guidelines) (Figure 3); and, (4) replacing the static image with GIF at the end of the questionnaire and editing the interaction text (Figure 4).

To minimize the difficulties faced by some participants, a tutorial with instructions for using ADA Blocks was created and made available. ADA Ajuda ("ADA Help") presents a welcome video, explains the objective of the ADA Blocks assistant in helping teachers choose block programming tools and guides them to watch the videos on how to use the assistant, which are separated by topics.

Another improvement made to ADA Blocks is related to technical terms, as highlighted by a participant in the feasibility study. In the recommendation questionnaire, the acronym "IoT" was replaced by its meaning in Portuguese, Internet das Coisas ("Internet of Things" – IoT). This version of ADA Blocks was published in the Extended Annals of the XII Brazilian Congress of Informatics In Education, Apps.Edu Competition (Perin, Silva, & Valentim, 2023).

### 4.8 Accessibility Automatic Assessments of ADA Blocks

Some advantages of using automatic evaluators consist of the speed in obtaining results and the reliability of the process, as it produces results that can be reproduced (Vigo & Brajnik, 2011). For this reason, we sought to evaluate ADA Blocks using accessibility automatic evaluators. Next, the planning, execution, and analysis of the automatic evaluations carried out will be presented.

[6]Federal University of Paraná (UFPR) - CAAE: 52371621.7.0000.0102 - Opinion Number: 5.140.422.


Figure 2: Recommendation questionnaire initial interaction screen.

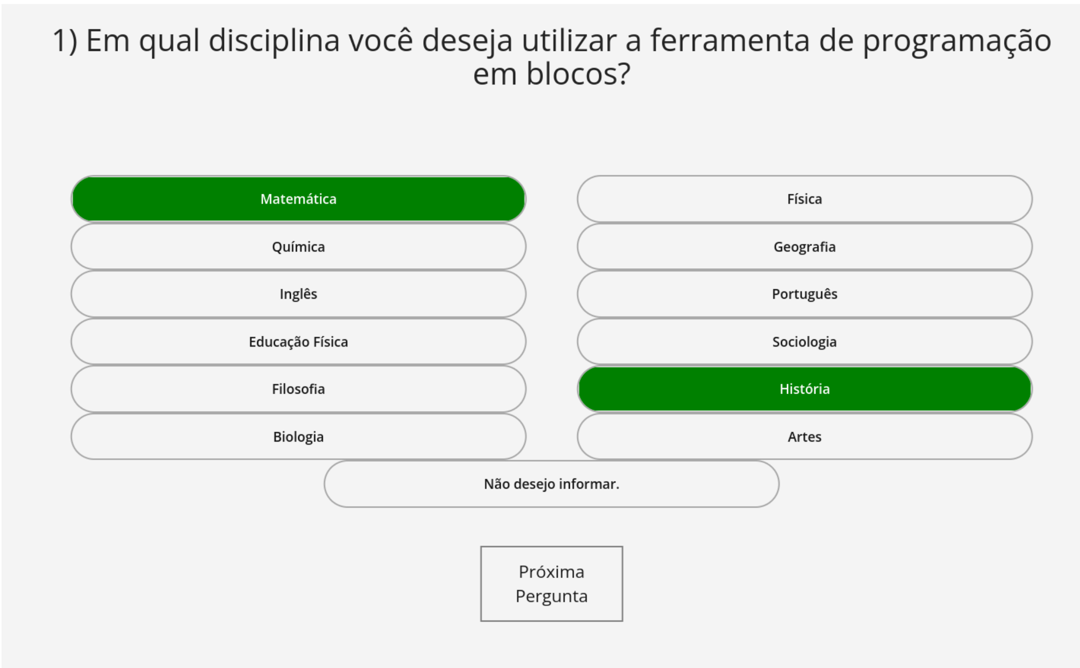


Figure 3: Home screen for the recommendation questionnaire questions.

**• Planning automatic assessments**

To carry out the accessibility automatic assessment of ADA Blocks, the evaluators that would be used were defined: AccessMonitor, ASES and e-Scanner. AccessMonitor was chosen


Figure 4: Final interaction screen with recommendation questionnaire.

because it follows WCAG guidelines, while ASES and e-Scanner were chosen because they are Brazilian platforms and follow eMAG 3.1 guidelines (based on WCAG).

AccessMonitor[7] was developed and is maintained by the Portuguese Foundation for Science and Technology (Fundação para a Ciência e a Tecnologia – FCT) unit. AccessMonitor is an automatic validator that evaluates web pages regarding the implementation of accessibility guidelines in their HTML content, based on WCAG 2.1. To use AccessMonitor, it is not necessary to carry out any installation and it is independent of browser and operating system.

ASES[8] is a Website Accessibility Evaluator and Simulator, developed by the Brazilian Department of Electronic Government in partnership with the Federal Institute of Rio Grande do Sul (Instituto Federal do Rio Grande do Sul - IFRS). It is available in Web and Desktop versions, allowing you to evaluate, simulate and correct the accessibility of pages, websites and portals (ASES, 2018), based on the recommendations indicated in the note and summary of the Accessibility Assessment that the tool generates. ASES is recommended for evaluating accessibility on web pages by eMAG 3.1 (eMAG, 2014).

e-Scanner[9] is a plug-in, built to quickly scan the source code of the document opened in the browser and display errors and alerts found, and is based on eMAG. Furthermore, the e-Scanner also displays information about eMAG and other validation tools.

It was decided that the collection of results returned by the automatic evaluators would be based on the detailed reports issued by each evaluator. The analysis of the results obtained was

[7]https://accessmonitor.acessabilidade.gov.pt/
[8]https://asesweb.governoeletronico.gov.br
[9]https://chromewebstore.google.com/detail/escanner/mpiipiobgejghkocofogeonfkapjgfmk

carried out using the Grounded Theory (GT) method, as defined by Corbin and Strauss, 2014. The execution of the evaluation and data analysis will be described below.

**• Execution of automatic evaluations**

The accessibility assessment process in the AccessMonitor and ASES automatic evaluators occurred in the same way. The evaluator's website was accessed via a web browser. In each tool, there is a field designed to insert the link to the page you want to evaluate, where the ADA Blocks[10] access link was inserted. After inserting the link, the "validar/executar" ("validate/execute") button was selected to start the evaluation. Once the evaluation was completed, the results were displayed on the evaluator interface. In AccessMonitor, the results were presented through a list of identified accessibility problems, classified by severity (A, AA or AAA), along with recommendations to correct them. In ASES, they were presented through a list of identified accessibility problems, classified by Section.

Regarding the e-Scanner, as it is a plugin and as the Google Chrome browser was used, it was necessary to access the Chrome Web Store to install it in the browser. Later, when browsing the ADA Blocks website, click on the e-Scanner icon in the browser toolbar to start the analysis. Finally, e-Scanner displayed a report with the accessibility problems found on the page.

In general, the three automatic evaluators used issued detailed reports of their evaluations, providing a comprehensive analysis of the accessibility problems found on the web pages of ADA Blocks. Data collection for this study was carried out through these reports, allowing a consistent and accurate analysis of the accessibility gaps identified by each evaluator.

**• Analysis of automatic evaluations**

In Section 5.6, the initial screens of the automatic evaluators generated in the ADA Blocks evaluation will be presented and discussed. The automatic evaluator screens present quantitative evaluation data following WCAG and eMAG standards. The AccessMonitor evaluator uses a scoring system, ASES uses a percentage evaluation metric, and e-Scanner presents the number of errors found without assigning a specific metric.

The qualitative data obtained from the automatic evaluators were analyzed using the GT method, defined by Corbin and Strauss (2014). The 1st stage carried out was open coding, where the results of the evaluations were coded as presented by the evaluator. Subsequently (2nd stage), in axial coding, the codes were grouped according to their properties and related to each other, thus forming categories that represent them. The 3rd stage (selective coding) was not carried out, as the aim is not to create a theory. The open and axial coding steps were sufficient for data analysis. The categories and codes obtained in the qualitative analysis will be described below.

## 5 Results and discussions

### 5.1 Survey on the use of DICTs in remote teaching by teachers

In general, the majority of participants demonstrated that the experience of teaching remote classes was interesting, they considered themselves creative in the teaching process and demonstrated that

[10]https://adablocks.com.br/

the experience of teaching remote classes was comfortable. With the results, it was possible to see that the teacher was challenged to use technological resources more effectively and to search for new teaching and learning strategies, making him rethink the traditional teaching model and methods. It was noticed that even though the teacher has access to technological resources, the way of teaching is still aligned with the traditional education model. Teachers did not describe having used emerging technologies, such as block programming, which can contribute to better student engagement and associate theory and practice.

This opinion survey contributed to the development of ADA Blocks, allowing us to learn how some teachers used DICTs in remote teaching. It was noticed that teachers of subjects that are not in the area of Informatics demonstrated greater difficulty in using it. In this context, these teachers may be better prepared to use these resources and be open to tool suggestions, for example, using ADA Blocks to suggest block programming tools. Additionally, this opinion survey played an important role in defining the target audience and research context, which specifically involves high school teachers who can use block programming in the classroom. This research was published in the 12th Computer on the Beach in 2021 (Perin, Silva, & Valentim, 2021).

### 5.2 Systematic Mapping Study

The answers achieved in each subquestion provided an overview of the use of block programming in the context of Education 4.0. The data shows that: **(SQ1)** 60.87% (N = 28) were carried out in the school environment. In **(SQ1.1)**, a study carried out in the Maker laboratory was identified; two studies related to university extension activities in partnership with high schools; four studies were carried out in a computer laboratory, while nine studies took place in the classroom. In **(SQ2)**, 78.26% (N = 36) of the studies were conducted by the researchers themselves. Some worked actively in the teaching process, others assisted the teacher or course instructor, others simply accompanied these teachers in conducting classes. In **(SQ3)**, 41.30% (N = 19) of the studies mention that students received training in some block programming tool and few studies mention the training of high school teachers in block programming. In **(SQ4)**, the majority of studies were carried out with students, 31 of which did not mention the High School grade. Few studies have been conducted with teachers. In **(SQ5)**, the subjects that had content related to block programming were: Mathematics (8 studies), science (8 studies), IT (6 studies), Physics (3 studies), Social Studies and/ or Humanities (2 studies), Programming Logic (1 study), English (1 study), Biology (1 study) and 1 study that was Multidisciplinary. In **(SQ6)**, the following block programming tools were identified: Scratch, MIT App Inventor and Snap!. Some can be used with emerging technologies, such as Robotics and Digital Games. In **(SQ6.1)**, Educational Robotics was identified in eleven studies, being the technology most used. Most studies do not mention the use of emerging technologies. Emerging technologies can enable students to work with confidence, solve problems, think critically and create innovative solutions and processes.

In general, SMS contributed to the construction of ADA Blocks in terms of determining the target audience. Through SMS, it was identified that few studies report the use of block programming by teachers, suggesting a need for them to use this tool, considering their specific context of use. Furthermore, it contributed to the construction of the body of knowledge that served as a database on the block programming tools that were incorporated into ADA Blocks.

An example is the Scratch tool, which can be applied in the Mathematics discipline, one of the features investigated and included in ADA Blocks to assist in the appropriate selection of tools.

### 5.3 Survey on the use of block programming tools

In this study it was possible to identify the block programming tools that teachers use or have already used. Therefore, it can be seen that 21 teachers use or have already used the Scratch tool; 16 Code.org; 16 MIT App Inventor; 12 Lego; 7 S4A; 3 Roblock; 3 MBlock; 3 Blockuino; 2 Micro:bit; 2 DroneBlocks; 2 Blocky; 1 Tinkecard; 1 Sketchware; 1 MusicBloks; 1 MakeCode; 1 Lightbot; 1 Kodular; 1 Engage; and 1 Choico. Furthermore, 77.80% (N = 28) responded that they had not received training to use the tools, while 22.20% (N = 8) responded that they had received it. This result occurred because the majority of participants are teachers in the area of Computing and are more familiar with the tools, while the others are regular high school and EJA teachers.

It was observed that some teachers were interested in using block programming tools, even when faced with difficulties. Teachers, especially those in Professional and Technological Education, have sought to use emerging technologies, such as Robotics and Digital Games, combined with block programming. Additionally, teachers listed the following skills that students can develop through block programming: computational thinking, logical reasoning and problem solving. Difficulties were also found in teaching and learning block programming, such as introducing technologies into everyday school life and teaching logic and mathematical concepts.

Furthermore, in this study some tools were identified that were not mentioned in the SMS, such as DroneBlocks and Tinkecard, but which have been used by high school teachers, and which can be used in an interdisciplinary way. These tools, even though they are not widely covered in the literature, demonstrate their relevance and applicability in the integration of different disciplines, providing teachers with creative and innovative opportunities to approach curricular content in a more comprehensive and contextualized way. In view of this, the need to carry out a benchmark was realized, in order to identify and characterize as many block programming tools as possible that can be used in High School subjects.

In general, this opinion poll was carried out in parallel to the SMS. The data from this study, combined with data from other studies presented here in subsections 4.1, 4.2 and 4.4 served as the basis for the construction of ADA Blocks.

### 5.4 Benchmark

A total of 77.59% (N = 45) work on web platforms; 29.31% (N = 17) on Windows operating system; 27.59% (N = 16) of the materials available to teachers are tutorials; 20.69% (N = 12) of student materials are tutorials; 89.66% (N = 52) are in the English language; 74.14% (N = 43) do not require registration to use; 22.41% (N = 13) can be used in the Mathematics subject; and 32.76% (N = 19) of the tools are associated with the Internet of Things (IoT).

Overall, benchmark served as a basis for identifying and categorizing these tools and was used to build ADA Blocks. After analyzing these tools, it was realized that teachers may become confused when choosing which tool best suits their context and needs.

Therefore, the need to develop ADA Blocks for teachers when choosing block programming tools in the context of high school was noted. It is believed that supporting the teacher in this purpose can help him in his teaching process, taking into account some characteristics of these tools.

## 5.5 Feasibility Study

In this study, it was observed that the majority of teachers agree that the tool is easy to use and useful. Teachers are also interested in using ADA Blocks in the future. Difficulties in use were perceived due to the adoption of technical terms. Of the limitations, teachers listed: limitations related to suggesting only one tool to the teacher and lack of accessibility resources, which motivated the conduct of an accessibility automatic assessment with ADA Blocks, where results will be presented in Section 5. Of the ideas to improve the assistant, teachers listed the implementation of tags that are related to the content, insertion of links on materials to support teacher training and insertion of accessibility resources.

## 5.6 Accessibility Automatic Assessments

The AccessMonitor evaluator (Figure 5) identified that ADA Blocks has a score of 5.2 in compliance with Accessibility standards, with 14 acceptable programming practices, where 9 are related to criterion A and 6 related to criterion AA of the WCAG. Regarding practices that need to be checked manually, 5 practices were identified, 3 related to criterion A and 2 related to criterion AAA. Regarding unacceptable practices, 12 practices were identified, of which 8 do not comply with criteria A, 2 do not comply with criteria AA and 2 do not comply with criteria AAA.

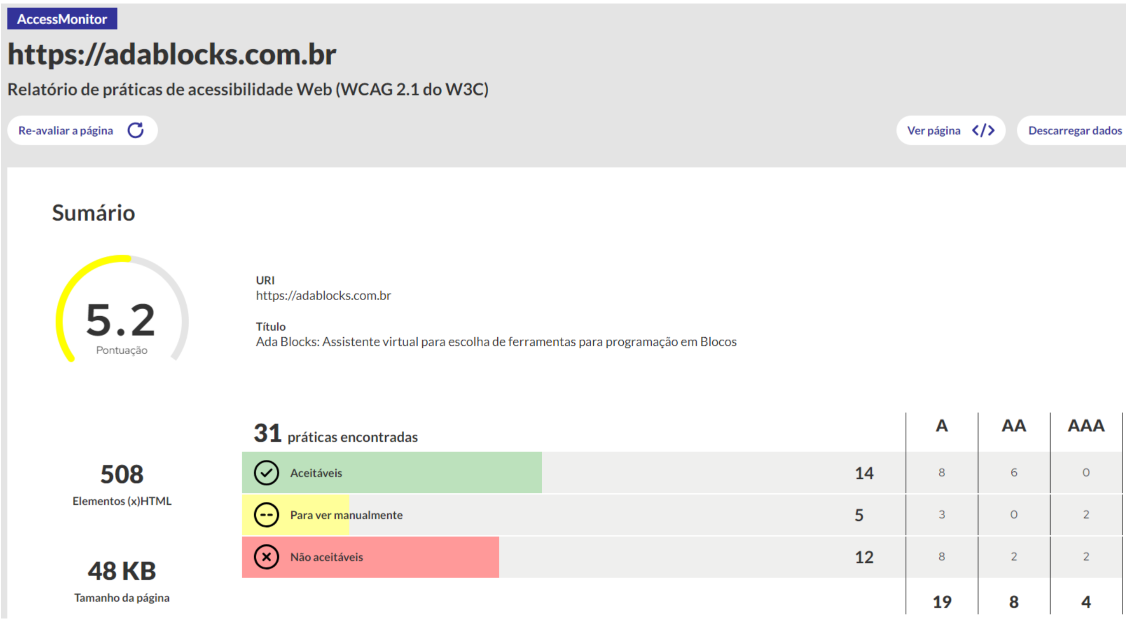


Figure 5: AccessMonitor automatic evaluator screen.

The ASES evaluator (Figure 6) identified that ADA Blocks is 61.57% compliant with Accessibility standards. A total of 40 errors were found, 16 in marking, 1 in behavior, 22 in content/information and 1 in presentation/design. In addition, 90 warnings were found, 72 for marking, 6 for behavior, 2 for content/information and 10 for multimedia. Therefore, it can be seen that most errors are related to the markup and content/information section (HTML).

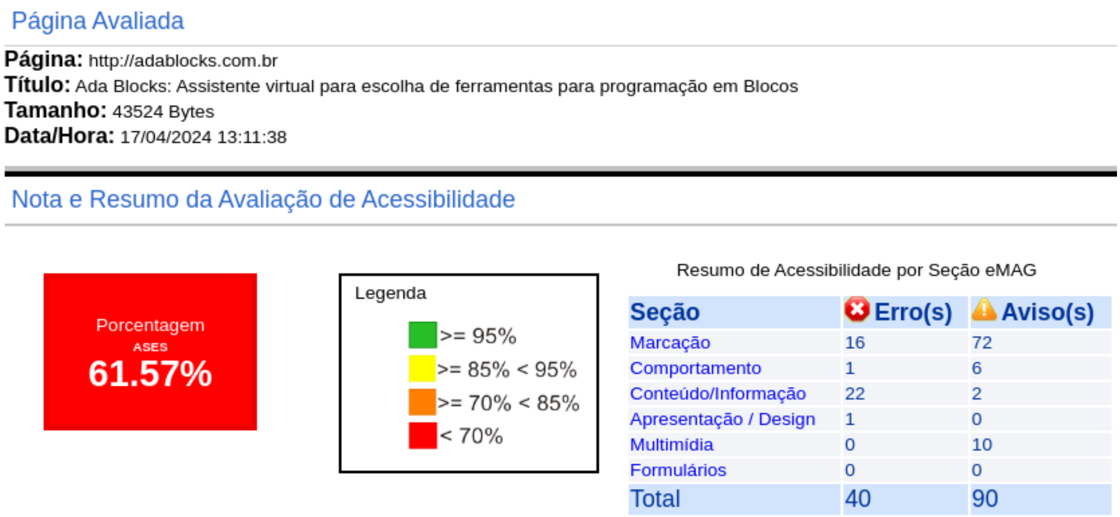


Figure 6: ASES automatic evaluator screen.

The e-Scanner evaluator (Figure 7) identified 8 errors in the development of the page related to accessibility standards and issued 24 alerts for practices that need to be checked by the programmer.

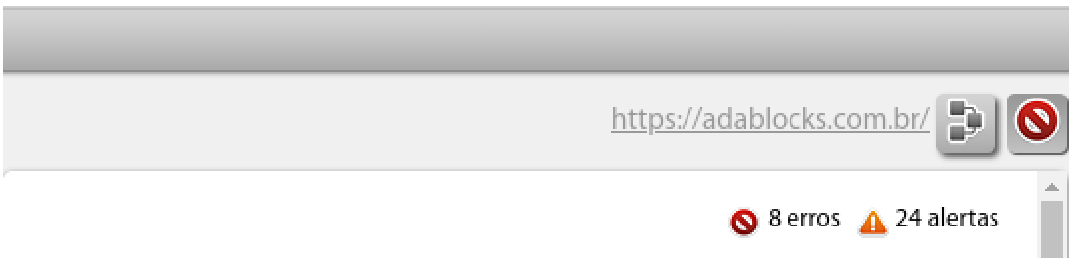


Figure 7: e-Scanner automatic evaluator screen.

Regarding **HTML errors**, errors can be cited regarding the multiple occurrences of the “H1” element of the header in the markup section (see the evaluator’s considerations C1-e-Scanner, C2-ASES and C3-ASES). Error about the absence of the ‘lang’ attribute (see evaluator considerations C4-e-Scanner, C5-ASES and C6-AccessMonitor). Error about the absence of the ‘alt’ attribute (see evaluator considerations C7-e-Scanner, C8-ASES and C9-AccessMonitor). Error regarding use of the ’script’ element without ’noscript’ (see reviewer’s consideration C10-e-Scanner).

*“Multiple occurrences of the element “H1”, on lines: 451, 462, 567, 675, 687, 689, 692, 697 and 716 (C1-e-Scanner) ”*

*“Organize HTML code logically and semantically. Check line 567 of the Source Code (C2-ASES)”*

*“Use header levels correctly. Check line 449, 451, 462, 474, 567, 567, 675, 687, 689, 692, 697, 716, 718, 729 of the Source Code (C3-ASES)”*

*"Absence of the 'lang' attribute, on line: 4 (C4-e-Scanner)"*

*"Identify the main language of the page. Check line 567 of the Source Code (C5-ASES)"*

*"I found that the 'lang' attribute is missing (C6-AccessMonitor)"*

*"Absence of the 'alt' attribute, on lines: 372 and 447. 'alt' attribute without value, on lines: 469, 480, 491, 503, 515, 526, 724, 737 and 750 (C7-e -Scanner)"*

*"Provide a text alternative for the site's images. Check lines 372, 447, 469, 480, 491, 503, 515, 526, 724, 737, 750 of the Source Code (C8-ASES)"*

*"I found 2 images on the page that do not have the necessary alternative text equivalent (C9-AccessMonitor)"*

*"Use of the 'script' element without 'noscript'" (C10-e-Scanner)*

In general, it was noticed that the three automatic evaluators identified the same HTML errors. An example of inappropriate use of heading levels is on the ADA Blocks home page (Figure 8A) where both titles are written using the same element (H1). Confirmation of inappropriate use came after manual verification of code lines 451 and 462, which correspond to these titles (Figure 8B), and which were pointed out by e-Scanner and ASES evaluators as examples of incorrect use of levels of headers (H1). WCAG recommends organizing the structure of a page in a hierarchical manner of titles and subtitles. This means, every page must contain at least one H1 level heading. Levels H2 to H6 can be used more than once on the same page. This organization meets success criteria 1.3.1 (Level A) of WCAG 2.1. In view of the problem presented, the intention is to review the lines of code related to titles, with the aim of restructuring them in a hierarchical manner. This organization aims to meet WCAG criteria and make reading ADA Blocks more accessible for screen reader systems, such as DOSVOX[11], used by blind people or those with low vision. In this way, ADA Blocks could be used by other audiences, including teachers with visual impairments who depend on assistive technology to navigate web pages.

It was also noticed that the three evaluators identified the absence of the 'lang' attribute in ADA Blocks as an HTML error. After manual verification, this failure was confirmed (Figure 9). The 'lang' attribute specifies the primary language of a web page's content. Its use is important to ensure accessibility, as it helps screen readers to correctly pronounce the page content. Furthermore, using the 'lang' attribute is related to WCAG success criterion 3.1.1 (Level A). Therefore, to guarantee the accessibility of ADA Blocks and the reading of the page by assistive technology, the 'lang' attribute must be defined as follows: "`<html lang="pt-br" >`".

Regarding the HTML error about the absence of the 'alt' attribute, pointed out by the three evaluators, the problem was also confirmed after a manual check. An example of manual evaluation related to the absence of the 'alt' attribute can be seen in Figure 10. In this example, line of code 372 (Figure 10C) concerns the ADA Blocks image that contains the blocks that represent block programming (Figure 10A). Line 447 (Figure 10D), concerns the GIF on the ADA Blocks home page, where she welcomes the teacher, and presents herself as a virtual assistant (Figure 10B). The 'alt' attribute specifies alternative text for an image if the image cannot be displayed (due to a slow connection, an error in the "src" attribute, or if the user uses a screen reader). Therefore, it is common to use the empty 'alt' attribute in HTML for decorative images. However, all

[11]https://intervox.nce.ufrj.br/dosvox/download.htm

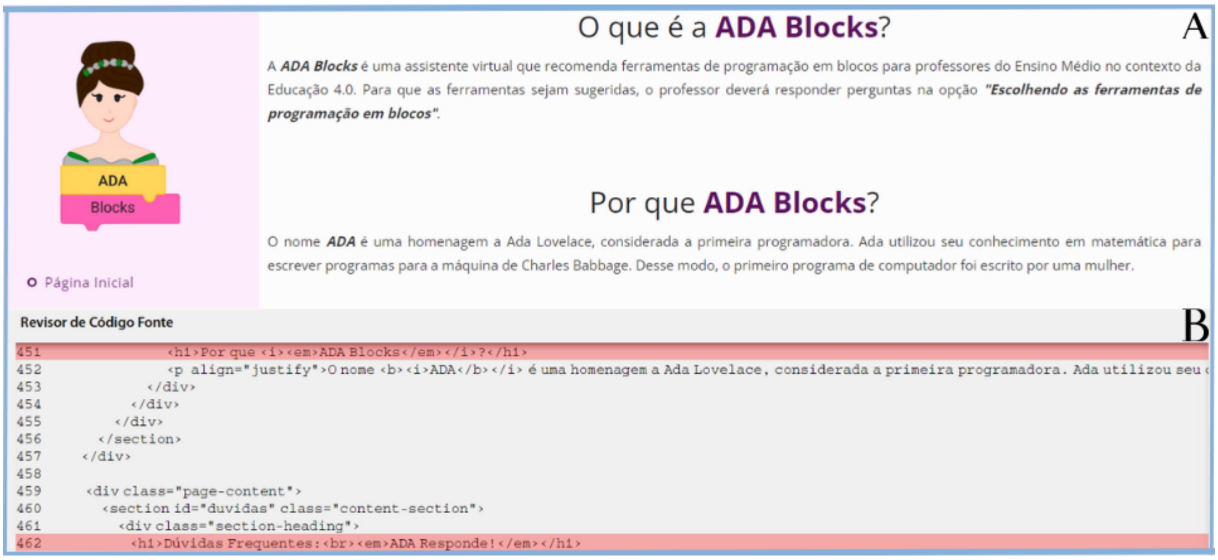


Figure 8: ADA Blocks home page and code snippets with inappropriate use of header elements.

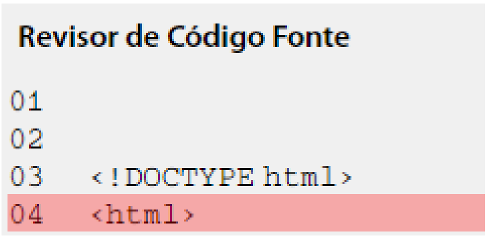


Figure 9: HTML excerpt showing absence of 'lang' attribute.

images that do not convey content, that is, decorative images, must be inserted using CSS. To solve the problem of decorative images inserted via HTML, it is necessary to create a CSS file (in this case, called style.css) where the image styles will be defined, including size, height, width and description. After defining the styles in the style.css file, on the index.html home page, the following line of code must be inserted: "`<link rel="stylesheet" href="style.css" >`". This line of code calls the style.css file and applies the styles defined in it to the HTML page. To display images according to the defined styles, you must use the following HTML structure: " <div id="elementoDoCSS" ></div>". Within the style.css file, styles can be defined for the element with the id "elementoDoCSS", including the background image and other visual properties as needed. You must also add the image description using appropriate CSS properties, such as "content" for pseudo-elements or "alt" for image elements. Therefore, using CSS to include decorative images is related to success criterion 1.1.1 (Level A) of the WCAG. Regarding the empty "alt" attribute (Figure 10D), we can solve the image description problem by inserting the "alt" attribute with its respective description, since it does not a decorative image. In this way, line of code 447 would be adjusted to: "`<img src="img/ada4.gif" alt="GIF interaction of the virtual assistant ADA Blocks welcoming users and introducing herself " width="400" >`". This adjustment must be made in all lines of code where the images are not just decorative. Thus, the decorative images will be inserted correctly, along with their descriptions, as well as the non-decorative images, making content accessible to assistive technologies.

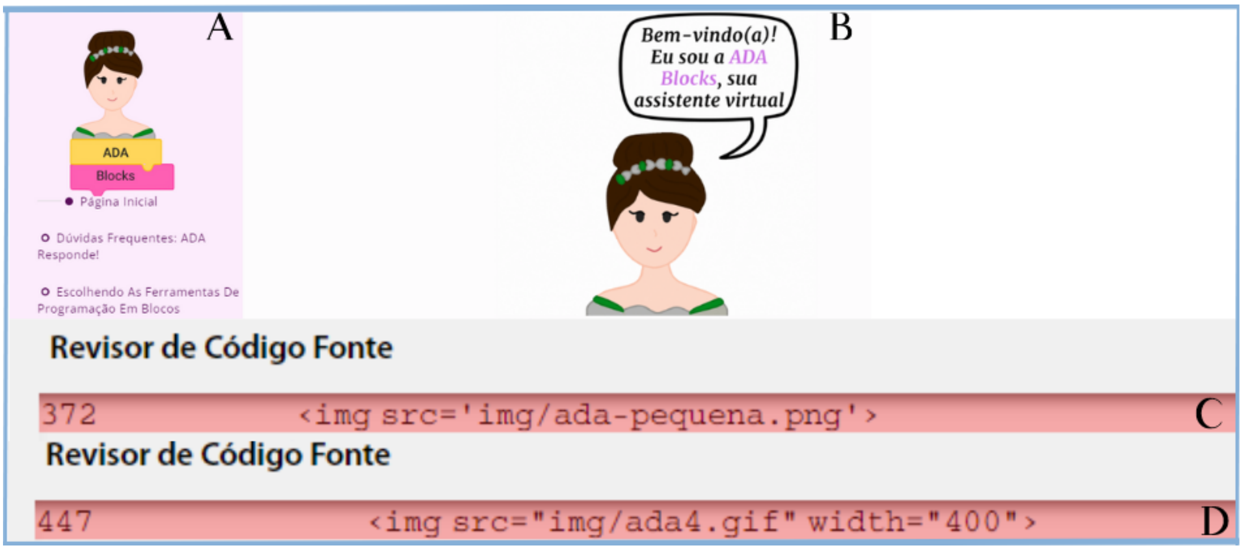


Figure 10: HTML snippet showing the absence of the 'alt' attribute.

Regarding the error caused by the use of the 'script' attribute without the 'noscript' element, it was necessary to carry out a manual search and identify that there was indeed an absence of the corresponding 'noscript' elements. According to W3C, 2024, the 'noscript' element allows authors to provide alternative content when a script is not executed. In other words, if the user's browser is so old that it cannot execute JavaScript code (a programming language that works on the client in this case), or if JavaScript support is disabled in the browser, an alternative text message must be provided. This way, the screen reader can read this message and warn the user that the script was not executed. Furthermore, using the 'noscript' attribute is related to WCAG success criterion 1.3.1 (Level A). To solve the problem of using the 'script' attribute without the 'noscript' element in ADA Blocks, it will be necessary to add the 'noscript' element right after the 'script' block. The following example represents the correct structure of how to add the 'noscript' element.

```
<script>
...has a JavaScript command that writes a message to the document....
</script>

<noscript>
<p> Provides an alternative message to be displayed if the browser
does not support JavaScript. </p>
</noscript>
```

Finally, you can see that the HTML issues mentioned above are all related to WCAG Level A. Level A establishes the minimum requirements to make content accessible. Meeting Level A success criteria means that a significant portion of accessibility barriers have been eliminated, allowing a greater number of people to use and interact with the content.

Regarding **Recommendations related to audio accessibility**, we can mention the recommendation to make descriptive and alternative transcription in Libras available in recorded audio (see the consideration of C1-e-Scanner and C2-e-Scanner).

*“Recorded audio must have a descriptive transcription.(C1-e-Scanner)”*

*“[...] The alternative in Libras is also desirable. (C2-e-Scanner)”*

ADA Blocks does not provide recorded explanatory audio tracks to allow the inclusion of descriptive or alternative transcriptions in Libras, despite having VLibras[12] configured to assist in this aspect. However, this recommendation inspired an idea for the recorded videos: adding a Libras’ interpreter in the bottom left corner of the video, translating the audio into sign language. Additionally, this recommendation highlighted the importance of audio transcription when inserting information through this format, as in a future idea of inserting podcasts of reports of experiences from teachers who used the tools suggested by ADA Blocks. It was also planned to add text right after the audio track, indicating the minutes of each speech to facilitate reading, thus increasing accessibility for deaf and blind people who use assistive technology.

Regarding **recommendations related to image accessibility**, we can mention the recommendation to provide a text alternative for form image buttons (see the consideration of C1-AccessMonitor) and the recommendation for animations that start automatically on a page provide control mechanisms for the user (pause, stop or hide animation) (see C2-e-Scanner consideration).

*“Check if the alternative textual equivalent existing in the graphic buttons serves information or function equal to that performed by the graphic button on the page (C1-AccesMonitor)”*

*“For any animation that starts automatically on the page, mechanisms must be provided so that the user can pause, stop or hide such animation (C2-e-Scanner)”*

After a more detailed manual check of WCAG, we found that, although there was no error in the code line, the recommendation of a text alternative for graphic buttons is a necessary adjustment to be made. In general, this suggestion is associated with the “aria-label” attribute, which configures text to be processed by the screen reader. The “aria-label” attribute performs the same function as the ‘alt’ attribute, with the difference that one replaces the error icon of an image not displayed when the “src” is invalid (‘alt’), while the other renders text for buttons. Therefore, we identified the need to adjust the “ir” and “voltar” (“go” and “back”) buttons in the “Dúvidas Frequentes: ADA Responde” ("Frequently Asked Questions: ADA Responds!") menu, as well as in the buttons on the ADA Blocks assistant’s recommendation form. These adjustments consist of configuring the “aria-label” attribute on these elements, making ADA Blocks more accessible and readable for screen readers.

Regarding the suggestion to allow the user to pause, stop or hide animations that start automatically, this recommendation is especially relevant for GIFs embedded in ADA Blocks. For example, we can mention the GIF on the home page, where ADA Blocks welcomes the user and introduces itself, and the GIF presented at the end of the recommendation questionnaire, after the teacher answers it. In the latter, ADA interacts, identifying the block programming tools found and suggesting them, based on the responses to the recommendation questionnaire. The inclusion of GIFs was an improvement made after the feasibility study (Subsection 4.7) with the aim of improving the assistant’s interaction with the user. However, in terms of accessibility, it still does not meet the appropriate requirements, in addition to not being in line with the user’s autonomy and control over the page. To follow this recommendation, you will need to reorganize the GIFs, separating the text from the balloons that simulate ADA’s speech from the GIF itself. This will

[12]https://www.vlibras.com.br/

allow users to completely disable GIFs from autoplaying, while the text can continue to display. This separation will make it easier to implement options to disable autoplay, either through user preferences or a switch directly in the interface.

Regarding **recommendations related to accessibility in videos**, we can mention the recommendation to use a sound or textual alternative for videos that do not include audio tracks and to use subtitles for videos that contain spoken audio and in the natural language of the page (see C1-e-Scanner and C2-e-Scanner considerations) and Recommendation to provide audio description for pre-recorded video (see C3-AccessMonitor consideration).

*"There must be a sound or textual alternative for videos that do not include audio tracks (C1-e-Scanner)"*

*"For videos that contain spoken audio and in the natural language of the page, subtitles must be provided (C2-e-Scanner)"*

*"Offer audio description for pre-recorded video. Check source code line(s): 548, 688, 690, 693, 698 (C3-AccessMonitor)"*

Video accessibility recommendations, based on WCAG, are crucial to ensuring that all people, regardless of their abilities or disabilities, can access and understand audiovisual content efficiently. After a manual analysis of the lines of code mentioned by AccessMonitor, it was confirmed that, although there are speech bubbles in ADA Blocks videos, these are not enough to guarantee complete accessibility.

Therefore, to meet the evaluators' suggestions, it is necessary to provide sound or textual alternatives for videos without audio. This involves making a narration or full transcript of video content available to users who cannot hear the audio, either through separate audio files or transcripts. When it comes to providing subtitles for videos with audio spoken in the natural language of the page, it is essential to include subtitles that are synchronized with the audio, following readability guidelines.

Additionally, for pre-recorded videos, it is recommended to provide a separate audio track that provides detailed descriptions of scenes, actions, facial expressions, and other important visual elements, ensuring complete understanding of the content for users with visual impairments. Implementing these solutions will ensure that videos on ADA Blocks are accessible to all users, promoting an inclusive and equitable experience.

Regarding **Recommendations for user control over page navigation**, we can mention the recommendation to make all page functions available via the keyboard (see the consideration of C1-e-Scanner), recommendation for focus via the keyboard be clearly marked and clickable (see C2-e-Scanner and C3-ASES considerations), and recommendation not to open new instances without user request (see C4-e-Scanner consideration).

"The page functions must be available when using only the keyboard (C1-e-Scanner)"

*"The area receiving keyboard focus must be clearly marked and clickable (C2-e-Scanner)"*

*" Enable the focused element to be visually evident (C3-ASES)"*

*"Do not open new instances without user request. Check source code line: 832 (C4-ASES)"*

In general, automatic evaluators recommended that page functions be available when only the keyboard is used because this ensures accessibility for people with disabilities or who rely exclusively on the keyboard to browse the web. After manually checking ASES and e-Scanner recommendations on the evidence of focused elements, we observed that some elements on the page are highlighted when navigated using the keyboard. However, the need to improve this evidence in existing elements and to make elements that are not yet visible is recognized, aiming for greater accessibility for all users.

To improve the visibility of in-focus elements, you can adopt one or more of the following activities: 1) Stylize the focus: Implement visual styles on the in-focus elements, such as color changes, highlights, borders or shadows, to make them stand out; 2) Improve contrast: Ensure that the focus style has adequate contrast in relation to the page background, making it easily noticeable; 3) Incorporate animations or smooth transitions: Add animations or smooth transitions to indicate the change of focus between elements, promoting a more fluid and understandable navigation experience; 4) Use ARIA attributes (Accessible Rich Internet Applications): Use ARIA attributes to describe the function and state of elements for users of assistive technology, such as screen readers; and 5) Conduct tests with real users: Conduct accessibility tests with real users, including those with visual impairments, to evaluate the effectiveness of the improvements implemented. These practices have the potential to significantly improve the accessibility of ADA Blocks, ensuring that all users can interact in an effective and understandable manner.

Regarding the recommendation not to open new instances without the user's request, this recommendation was applied because when new windows or tabs are opened automatically, users may feel confused or lost, especially if they are not expecting this action. This can also interfere with the user's browsing experience. Therefore, it is recommended that new instances are only opened in response to an explicit user action, such as clicking a specific link or button that clearly indicates that a new window or tab will be opened. This provides a more controlled, predictable and secure user experience. When manually evaluating the line of code mentioned by ASES, it was found that another instance is only opened when the user clicks on the access link, following this recommendation. However, we identified the need to provide the user with a form of confirmation, allowing them to consciously decide whether they wish to carry out the proposed action. This can contribute to an even more transparent and satisfying experience for the user.

## 6 ADA Blocks Accessibility Improvements

To minimize accessibility issues identified by automatic evaluators, ADA Blocks has been updated. Therefore, to resolve errors related to HTML heading levels, the headings of all pages were adjusted, following the accessibility best practices recommended by Mozilla (Mozilla, 2024). Thus, each repeated occurrence of <h1></h1> was adjusted to levels <h3> to <h6>, according to the appropriate hierarchy of subheadings.

Regarding the error of the 'script' element without 'noscript', the following code snippet has been added below each tag '<script>' to tag '<noscript>' containing a text message alternative for problems with JavaScript. Therefore, the correct use of the 'script' element was: " <script>...here is a JavaScript script...</script> <noscript> Your web browser does not support JavaScript because it is out of date or the support JavaScript is disabled.</noscript>". With this adjustment, screen readers will be able to warn the user when JavaScript code does not work.

To resolve the "Absence of 'lang' attribute error (line 4), the 'lang' attribute was added with the value of pt-br (Brazilian Portuguese) in tag HTML. This way, the line code that was previously "<html>" (Figure 9) was rewritten to "<html lang="pt-br" >". Thus, the main language of the page was defined, where all others were. Elements from it inherit this information, as defined by W3C accessibility standards (Penn State University, 2023; W3C, 2023).

Finally, regarding the absence of the 'alt' attribute, the 'alt' attribute was added to all code snippets that reference ADA Blocks images. Furthermore, the image description text was included in the 'alt' attribute according to the visual content of each image. For an example of text inserted in the 'alt' attribute of an image, the main image from ADA Blocks can be cited. This way, the section of code that references the image was adjusted from <img src='img/ada-pequena.png'> to <img src='img/ada-pequena.png' alt="Here's a logo from ADA Blocks. It consists of a drawing of a white woman, with brown hair, wearing a dress reminiscent of the middle ages (in reference to Ada Lovelace, there is a block diagram written Ada Blocks" .

In the "Dúvidas Frequentes: ADA Responde!" ("Frequently Asked Questions: ADA Responds!") field, the descriptions of the images using the 'alt' attribute are presented below, in the order in which the images appear. The description of the first image is:" Here is an image of ADA sitting in a chair in front of a computer on a table, her chair and her coffee cup are pink. There are questions in the ADA's head because she has doubts". This image refers to the question "O que é programação e qual a sua importância?" ("What is programming and how important is it?"). The description of the second image is: "The illustrative image shows a magnifying glass on top of the ADA monitor, and in it you can see programming in blocks on the screen". This image refers to the question "O que são os ambientes de programação em blocos?" ("What are block programming environments?").

The description of the third image is: "Image of two computer monitors. The monitor on the left shows code in textual programming language and the monitor on the right shows code programmed in blocks. In the middle of them there is a question mark". This image refers to the question "Por que programação em blocos?" ("Why program in blocks?"). The description of the fourth image is:"Here is an image showing eleven programming language blocks in blocks of various colors. In each one, they have words, being: programming, computational thinking, logical reasoning, creativity, algorithms, robotics, communication, digital games, divide and conquer, collaboration and interdisciplinarity". This image refers to the question "Por que utilizar a programação em blocos na sala de aula?" ("Why use block programming in the classroom?").

The description of the fifth image is: "Here is an image that has a computer monitor with Education 4.0 written on the screen and a question mark next to it". This image refers to the question "O que é a Educação 4.0?" ("What is Education 4.0?"). Finally, the description of the sixth image is: "Here is an image that shows the design of an ADA Blocks table. On the table, there is a monitor, a wireless keyboard, a wired mouse, an open book written 4.0, a cup of coffee, a magnifying glass, two puzzle pieces, the ADA and a dark-skinned doll showing how program in blocks relating programming to Education 4.0". This image is related to the question "Qual é a relação entre a programação em blocos e a Educação 4.0?" ("What is the relationship between block programming and Education 4.0?").

In addition to inserting the 'alt' attribute with its description, the images were inserted using CSS, following the accessibility standards defined by WCAG and eMAG. Including the 'alt' attribute and describing the context of each image solved the image accessibility problem, and also highlighted an important aspect that was not considered during the development of Ada Blocks: contextualizing images to improve the semantic accessibility of the system.

After adjustments were made, a new accessibility assessment was carried out in the automatic evaluators. The AccessMonitor evaluator (Figure 11) identified that ADA Blocks increased its Accessibility standards compliance score from 5.2 (5) to 6.9 (Figure 11). The number of acceptable programming practices changed from 14 to 17, with 11 related to criterion A (previously there were 9 practices) and 6 related to WCAG criterion AA (maintained from the previous assessment). Regarding practices that need to be checked manually, it changed from 5 to 6, with 4 related to criterion A (previously there were 3 practices) and 2 related to criterion AAA (maintained in relation to the previous assessment). Regarding unacceptable practices, it changed from 12 to 7, with 3 related to criterion A (previously there were 8 practices), 2 related to criterion AA and 2 to AAA (maintained in relation to the previous assessment). In addition to the improvement in the ADA Blocks accessibility score, it is possible to see that there has been a significant improvement in unacceptable programming practices, mainly related to WCAG Criterion A.

The ASES evaluator (Figure 12) identified that ADA Blocks increased from 61.57% 6 according to Accessibility standards to 88.41% (Figure 12). The total number of errors decreased from 40 to 14, 4 of which were marking (previously 16) and 10 were content/information (previously 22). No behavioral errors were identified (previously there was 1 error) and presentation/design errors (previously there was 1 error). In addition, the warnings decreased from 90 warnings to 71, with 56 marking (previously 72), 4 behavioral (previously 6), 1 content/information (previously 2) and 10 multimedia (maintained in relation to the previous assessment). In this way, it can be seen that the majority of errors related to the markup and content/information section (HTML) were minimized, making ADA Blocks more accessible and in accordance with WCAG and eMAG accessibility standards.

Regarding the e-Scanner evaluator (Figure 13), no errors were found related to accessibility standards in the development of the page, unlike the 8 errors previously identified (Figure 7). Of the 24 alerts previously issued, 19 alerts have now been issued for practices that need to be verified by the programmer.

## 7 Limitations

As a limitation of this study, we cite the lack of systematic recording of the internal URLs evaluated or the environment configurations (OS, browsers, screen resolution, among others). In both the test and retest stages, this limitation is present. This gap impacts the full traceability of the tests, although the general functional criteria were documented in Subsection 5.6 and Section 6. Furthermore, the scope of the accessibility evaluation was limited to automated analysis, without complementation with testing by UX/WCAG experts or validation with real users (including people with visual, auditory, motor, or cognitive disabilities). Consequently, critical metrics such as subjective perception of accessibility and the identification of other barriers were not obtained. To overcome these limitations, we propose concrete actions for future research. In the traceabil-


Figure 11: AccessMonitor automatic evaluator screen after error correction.

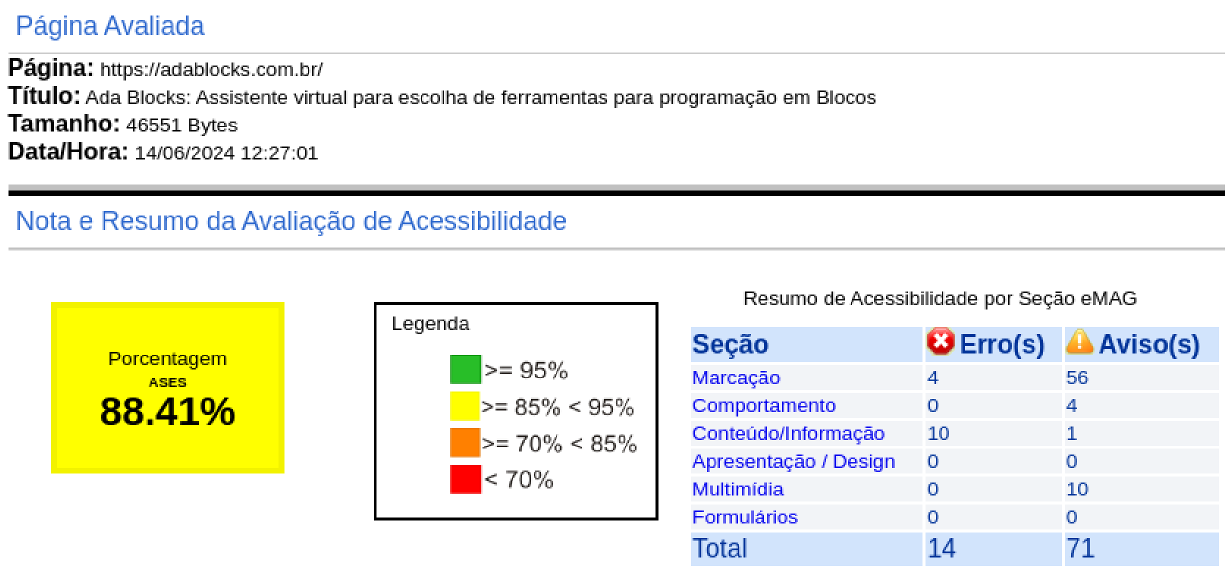


Figure 12: ASES automatic evaluator screen after error correction.

ity dimension, we will implement structured protocols for detailed recording of sampled URLs and snapshots of the technical environment (OS, browser, resolution). Regarding the multimodal evaluation, we plan to combine complementary methods: (a) testing with experts using heuristic analysis aligned with WCAG 2.2 guidelines; and (b) sessions with representative users, covering profiles such as visual and hearing impairments. In parallel, for extended validation, we intend to

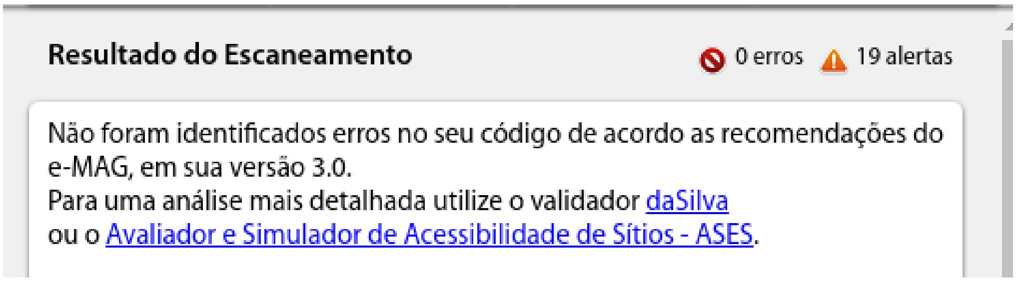


Figure 13: e-Scanner automatic evaluator screen after error correction.

include assistive devices (e.g., screen readers such as NVDA and VoiceOver, magnifiers) in the testing cycles, thus ensuring a more comprehensive analysis of accessibility barriers.

## 8 Final Considerations and Future Perspectives

In this paper, ADA Blocks and its research and development methodology were presented. ADA Blocks was developed to support and encourage high school teachers in choosing block programming tools. This virtual assistant recommends tools based on some characteristics of block programming tools. It also helps teachers choose tools to promote content personalization, develop computational thinking, among other skills and competencies.

The ADA Blocks development methodology consists of nine steps - including two surveys with high school teachers, SMS, benchmark of 58 block programming tools based on criteria such as discipline, technical support and emerging technologies (robotics, IoT), feasibility and automatic accessibility assessment. The first survey with high school teachers on the use of ICTs in teaching and learning processes during the remote period aimed to understand the difficulties teachers at this level face in using ICTs and identify which resources were used during social distancing due to the coronavirus. The SMS aimed to identify which ICTs support the teaching of programming and/or computational thinking through block programming in the context of Education 4.0. In this sense, the SMS was conducted by searching for studies published in journals, conferences, symposium, and workshops indexed in the ACM, IEEExplore, SpringerLink, and Scopus digital libraries. The second survey of high school teachers on the use of block programming tools as support materials for teaching and learning processes identified which block programming tools teachers use or have used, whether training was provided on how to use these tools, and which emerging technologies are being used in conjunction with block programming. The benchmark of block programming tools aimed to identify some specific features of block programming tools, including which platforms they work on, whether there is support material for teachers and/or students and what type of material is available, what emerging technologies are available, and which high school subjects the block programming tool can be used in. Based on these studies, ADA Blocks was innovative. The study of ADA Blocks virtual assistant with high school teachers enabled the construction of a body of knowledge for improving the assistant. Subsequently, improvements were made to the ADA Blocks, and accessibility assessments were performed using automated assessments.

Overall, ADA Blocks plays an important role in promoting the use of block programming and developing 21st Century skills and competencies by suggesting block programming tools to teachers based on their context of use. This makes it possible to bring TDICs closer to the classroom, encouraging the teaching of computing in an interdisciplinary way. By integrating block programming into different disciplines, it allows students to develop skills related to problem solving, computational thinking, collaboration and creativity. This approach contributes to preparing students with the skills and competencies necessary for the 21st Century.

Furthermore, ADA Blocks contributes to the scientific community of Informatics in Education and Computing Education and teachers of introductory programming courses at public and private universities in terms of having access to specific features of block programming tools that can support research and content of undergraduate classes in this context. For example, a researcher working on the topic of computational thinking for high school can benefit from ADA Blocks by identifying tools that can be used and that best suit the focus of their research.

This paper also presents a study evaluating the accessibility of ADA Blocks by automatic evaluators. The results of the automatic evaluators played an important role in identifying improvements in the accessibility aspect of ADA Blocks. By identifying errors in the 'alt', 'lang' attribute, among others, it was possible to appropriately adjust the descriptions of ADA Blocks images for users of assistive technologies. By identifying gaps in image descriptions, automatic evaluators provide a valuable opportunity for correction, making content more accessible to a variety of users. This not only increases the inclusion of people with visual impairments, but also improves the overall user experience, promoting a more accessible and inclusive platform for all ADA Blocks users.

The recommendations from the automatic evaluators were also essential in highlighting the importance of other accessibility alternatives in ADA Blocks. These alternatives include text transcription, audio descriptions and captions for images, videos and audios. Text transcription allows users with visual impairments or reading difficulties to access content through screen readers. Audio descriptions are crucial for people with visual impairments who rely on auditory information to understand visual content. Additionally, subtitles provide support for users who are deaf or hard of hearing, ensuring they can understand video and audio content.

Improvements were implemented in ADA Blocks, such as the inclusion of descriptions in images using the 'alt' attribute, to allow reading by assistive technologies; the insertion of decorative images via CSS; the definition of the main language of the page and the hierarchical organization of titles. These improvements resulted in a new version of ADA Blocks and an increase in accessibility scores, as assessed by AccessMonitor and ASES. Furthermore, in the e-Scanner, no more errors were identified in ADA Blocks. However, some limitations of the study were noted, such as the lack of systematic recording of the internal URLs evaluated or the environment configurations, as well as day, time, browser versions or automatic evaluators, among others. In both the test and retest stages, this limitation is present.

As future work, we intend to conduct new studies with the current version of ADA Blocks, with the aim of seeking improvements. Therefore, we intend to evaluate ADA Blocks with users with visual and hearing impairments. Furthermore, we intend to implement detailed registration protocols (technical environments + sampled URLs) in future studies. We also plan to conduct further studies with high school teachers. The aim is to investigate what teachers' most frequent questions are and, thus, include more options for frequently asked questions topics, opening paths for new academic research. Improvements are intended improvements related to accessibility in ADA Blocks, incorporating Assistive Technology resources, such as audio description, videos with Libras interpreters and subtitles. It is intended to conduct evaluation studies of ADA Blocks with experts in the field of Human-Computer Interaction (HCI), aiming to identify possible accessibility problems. This approach will make it possible to relate the problems identified by experts with the recommendations of automatic evaluators. After this stage, the intention is to update ADA Blocks based on experts' assessments, and then carry out a new assessment with accessibility automatic evaluators to check whether the identified problems have been minimized or resolved. Finally, we intend to carry out new accessibility assessment studies with teachers who have some type of disability, with the aim of ensuring that ADA Blocks is truly accessible to this target audience.

## References

Albert, B., & Tullis, T. (2022). *Measuring the user experience: Collecting, analyzing, and presenting UX metrics*. Interactive Technologies. https://doi.org/10.1016/C2018-0-00693-3 [GS Search].

ASES. (2018). Avaliador e simulador de acessibilidade em sítios – ASES. Disponível em: https://softwarepublico.gov.br/social/ases. Acesso em:13 jun. 2024.

Basili, V. R., & Rombach, H. D. (1988). *Towards a comprehensive framework for reuse: A reuse-enabling software evolution environment* (tech. rep.). University of Maryland. College Park, MD, USA. [GS Search].

BNCC. (2018). Base nacional comum curricular. Disponível em: http://download.basenacionalcomum.mec.gov.br/. Acesso em: 13 jun. 2024.

Brasil. (2004). Decreto nº 5.296 de 02 de dezembro de 2004. http://www.planalto.gov.br/ccivil_03/_ato2004-2006/2004/decreto/d5296.htm Acesso em: 13 jun. 2024.

Burnett, M. M., & McIntyre, D. W. (1995). Visual programming [Disponível em: https://homepages.ecs.vuw.ac.nz/~elvis/db/references/burnett99visual.pdf. Acesso em: 13 jun. 2024]. *Computer-Los Alamitos-*, *28*, 14–14. [GS Search].

Cabral, A. L. T., Lima, N. V., & Albert, S. (2019). TDIC na educação básica: Perspectivas e desafios para as práticas de ensino da escrita. *Trabalhos em Linguística Aplicada*, *58*(3), 1134–1163. https://doi.org/10.1590/01031813554251420190620 [GS Search].

Cárdenas-Cobo, J., Vidal-Silva, C., Arévalo, L., & Torres, M. (2024). Applying recommendation system for developing programming competencies in children from a non-WEIRD context. *Education and Information Technologies*, *29*(8), 9355–9386. https://doi.org/10.1007/s10639-023-12156-y [GS Search].

Corbin, J., & Strauss, A. (2014). *Basics of qualitative research: Techniques and procedures for developing grounded theory*. Sage Publications. [GS Search].

eMAG. (2014). Modelo de acessibilidade em governo eletrônico. Disponível em: https://emag.governoeletronico.gov.br/. Acesso em: 13 jun. 2024.

Ferreira, R. W. C., & Sant'Ana, C. C. (2025). Jogos de matemática no Scratch: Relato de um minicurso promovido com vistas a contribuir com a formação docente. *Educação Matemática em Revista*, *30*(86), 1–17. https://doi.org/10.37001/emr.v30i86.4227 [GS Search].

Führ, R. C., & Haubenthal, W. R. (2019). Educação 4.0 e seus impactos no século XXI. *Educação no Século XXI - Volume 36 Tecnologia*, 61–78. [GS Search].

Guggemos, J., Seufert, S., & Román-González, M. (2019). Measuring computational thinking-adapting a performance test and a self-assessment instrument for German-speaking countries. *Proceedings of the 16th International Conference Cognition and Exploratory Learning in the Digital Age*, 183–191. https://doi.org/10.33965/celda2019_201911L023 [GS Search].

Haduong, P., & Brennan, K. (2019). Helping K–12 teachers get unstuck with Scratch: The design of an online professional learning experience. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 1095–1101. https://doi.org/10.1145/3287324.3287479 [GS Search].

Himmetoğlu, B., Ayduğ, D., & Bayrak, C. (2020). Education 4.0: Defining the teacher, the student, and the school manager aspects of the revolution. *Turkish Online Journal of Distance Education*, *21*(Special Issue), 1–20. https://doi.org/10.17718/tojde.770896 [GS Search].

Hudin, S. S., & Adii, A. (2024). Programming challenges experience by primary school students: A systematic literature review. *International Journal of Academic Research in Progressive Education and Development*, *13*(4), 489–502. https://ijarped.com/index.php/journal/article/view/3044

Hyury, A., & Jailton, J. (2025). Formação docente e pensamento computacional: Desafios na utilização do Scratch no ensino médio técnico. *Anais do XXXIII Workshop sobre Educação em Computação*, 746–757. https://doi.org/10.5753/wei.2025.8526 [GS Search].

IBGE. (2010). Pessoas com deficiência no Brasil em 2010. Disponível em: https://educa.ibge.gov.br/jovens/conheca-o-brasil/populacao/20551-pessoas-com-deficiencia.html. Acesso em: 13 jun. 2024.

IBGE. (2022). Pessoas com deficiência no Brasil em 2022. Disponível em: https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/37317-pessoas-com-deficiencia-tem-menor-acesso-a-educacao-ao-trabalho-e-a-renda. Acesso em: 13 jun. 2024.

ISO. (2003). ISO/TS 16071: Ergonomics of human-system interaction – guidance on accessibility for human-computer interfaces. Disponível em: https://www.iso.org/standard/30966.html. Acesso em: 13 jun. 2024.

Jocius, R., Joshi, D., Dong, Y., Robinson, R., Cateté, V., Barnes, T., Albert, J., Andrews, A., & Lytle, N. (2020). Code, connect, create: The 3C professional development model to support computational thinking infusion. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 971–977. https://doi.org/10.1145/3328778.3366797 [GS Search].

Kitchenham, B., & Charters, S. (2007). *Guidelines for performing systematic literature reviews in software engineering* (tech. rep.) (Disponível em: https://www.elsevier.com/__data/promis_misc/525444systematicreviewsguide.pdf. Acesso em: 13 jun. 2024). Keele University and University of Durham. Keele, UK. [GS Search].

Lee, I., & Malyn-Smith, J. (2020). Computational thinking integration patterns along the framework defining computational thinking from a disciplinary perspective. *Journal of Science Education and Technology*, *29*(1), 9–18. https://doi.org/10.1007/s10956-019-09802-x [GS Search].

Lemos, S. K. S., Teixeira, L. N. B., & Kafure, I. (2019). Acessível para quem? uma análise da acessibilidade em sítios institucionais. *Brazilian Journal of Development*, *5*(6), 4976–4989. https://doi.org/10.34117/bjdv5n6-0052 [GS Search].

Medeiros, R. P., Ramalho, G. L., & Falcão, T. P. (2019). A systematic literature review on teaching and learning introductory programming in higher education. *IEEE Transactions on Education*, *62*(2), 77–90. https://doi.org/10.1109/TE.2018.2864133 [GS Search].

Nielsen, J. (2007). *Usabilidade na web*. Elsevier Brasil. [GS Search].

Papadakis, S., & Orfanakis, V. (2018). Comparing novice programing environments for use in secondary education: App Inventor for Android vs. Alice. *International Journal of Technology Enhanced Learning*, *10*(1–2), 44–72. https://doi.org/10.1504/IJTEL.2018.088333 [GS Search].

Penn State University. (2023). Language tags in HTML: Declaring page language. Disponível em: https://accessibility.psu.edu/foreignlanguages/langtaghtml/. Acesso em: 13 jun. 2024.

Perin, A., Silva, D., & Valentim, N. (2023). ADA blocks: Assistente virtual para apoiar professores na escolha de ferramentas de programação em blocos. *Anais Estendidos do XII Congresso Brasileiro de Informática na Educação*, 137–140. https://doi.org/10.5753/cbie_estendido.2023.233147 [GS Search].

Perin, A., Silva, D. E., & Valentim, N. (2021). Um benchmark de ferramentas de programação em blocos que podem ser utilizadas nas salas de aula do ensino médio. *Anais do XXXII Simpósio Brasileiro de Informática na Educação*, 1162–1173. https://doi.org/10.5753/sbie.2021.217765 [GS Search].

Perin, A., Silva, D. E., & Valentim, N. (2022). Um estudo de viabilidade sobre a assistente virtual ADA blocks com professores do ensino médio. *Anais do XXXIII Simpósio Brasileiro de Informática na Educação*, 1243–1254. https://doi.org/10.5753/sbie.2022.224682 [GS Search].

Perin, A., Silva, D. E., & Valentim, N. M. C. (2023). Investigating block programming tools in high school to support education 4.0: A systematic mapping study. *Informatics in Education*, *22*(3), 463–498. https://doi.org/10.15388/infedu.2023.21 [GS Search].

Perin, A., Silva, D. E., & Valentim, N. M. (2021). Experiência de docentes do ensino médio em conduzir atividades remotas durante o distanciamento social: Uma análise baseada no contexto da educação 4.0. *Anais do Computer on the Beach*, *12*, 141–148. https://doi.org/10.14210/cotb.v12.p141-148 [GS Search].

Perin, A., Silva, D. E., & Valentim, N. M. C. (2022). Investigating the teaching of block programming in high school. *XVIII Brazilian Symposium on Information Systems*. https://doi.org/10.1145/3535511.3535543 [GS Search].

Pivetta, E. M., Saito, D. S., & Ulbricht, V. R. (2014). Surdos e acessibilidade: Análise de um ambiente virtual de ensino e aprendizagem. *Revista Brasileira de Educação Especial*, *20*(1), 147–162. https://doi.org/10.1590/S1413-65382014000100011 [GS Search].

Rios, L. K. S., Junior, A. O. C., Lima, J. P. F., Guedes, E. B., & Silva, C. A. (2019). Uma análise comparativa entre ambientes de programação em blocos para a interação com o Arduino. *Anais do Simpósio Ibero-Americano de Tecnologias Educacionais*, 345–356. [GS Search].

SBC. (2018). Diretrizes para o ensino de computação na educação básica [Disponível em: https://www.sbc.org.br/diretrizes-para-ensino-de-computacao-na-educacao-basica/. Acesso em: 13 jun. 2024]. [GS Search].

Silva, D. E., Lopes, T., Corrêa Sobrinho, M., & Valentim, N. M. C. (2020). Educação 4.0: Um estudo de caso com atividades de computação desplugada na amazônia brasileira. *Anais do Computer on the Beach*, *11*, 141–147. https://doi.org/10.14210/cotb.v11n1.p141-147 [GS Search].

Silva, E. H. (2021). *Análise de acessibilidade dos ambientes virtuais de aprendizagem (AVAs) do IFPE de acordo com as aplicações e padrões web, voltado para pessoas com deficiência visual* [Bachelor's Thesis]. Instituto Federal de Ciência e Tecnologia de Pernambuco – IFPE Campus Jaboatão dos Guararapes. [GS Search].

Souza, E. R., & Mont'Alvão, C. (2012). Web accessibility: Evaluation of a website with different semi-automatic evaluation tools. *Work*, *41*(1), 1567–1571. https://doi.org/10.3233/WOR-2012-0354-1567 [GS Search].

Souza, M. V. R., & França, A. C. C. (2013). Um estudo sobre as dificuldades no processo de aprendizagem de programação no curso de Análise e Desenvolvimento de Sistemas na FAFICA – Faculdade de Filosofia, Ciências e Letras de Caruaru-PE. *Revista da Escola Regional de Informática*, *2*(2), 19–27. [GS Search].

Viana, W., Araújo, M., Façanha, A., Pequeno, H., Fontenele, T., & Matos, B. (2017). Análise de acessibilidade no ambiente virtual de aprendizagem solar – um estudo de caso para usuários com deficiência visual. *Nuevas Ideas en Informática Educativa*, 281–289. [GS Search].

Vigo, M., & Brajnik, G. (2011). Automatic web accessibility metrics: Where we are and where we can go. *Interact. Comput.*, *23*(2), 137–155. https://doi.org/10.1016/j.intcom.2011.01.001 [GS Search].

Vinayakumar, R., Soman, K. P., & Menon, P. (2018). Fractal geometry: Enhancing computational thinking with MIT Scratch. *2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 1–6. https://doi.org/10.1109/ICCCNT.2018.8494096 [GS Search].

W3C. (2008). Web Content Accessibility Guidelines (WCAG) 2.0. Disponível em: https://www.w3.org/Translations/WCAG20-pt-br/. Acesso em: 13 jun. 2024.

W3C. (2023). Declaring language in HTML. Disponível em: https://www.w3.org/International/questions/qa-html-language-declarations.en. Acesso em: 13 jun. 2024.

W3C. (2024). Scripts: The NOSCRIPT element. Disponível em: https://www.w3.org/TR/html4/interact/scripts.html#h-18.3.1. Acesso em: 13 jun. 2024.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33–35. https://doi.org/10.1145/1118178.1118215 [GS Search].

Yett, B., Hutchins, N., Stein, G., Zare, H., Snyder, C., Biswas, G., Metelko, M., & Lédeczi, Á. (2020). A hands-on cybersecurity curriculum using a robotics platform. *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 1040–1046. https://doi.org/10.1145/3328778.3366878 [GS Search].

Yu, Q., Yu, K., & Li, B. (2025). Effects of block-based visual programming on K-12 students' learning outcomes. *Journal of Educational Computing Research*, *63*(1), 64–98. https://doi.org/10.1177/07356331241293163 [GS Search].

Zaffalon, F., Prisco, A., Souza, R., Teixeira, D., Paes, W., Evald, P., Tonin, N., Devincenzi, S., & Botelho, S. (2022). A recommender system of computer programming exercises based on student's multiple abilities and skills model. *2022 IEEE Frontiers in Education Conference (FIE)*, 1–8. https://doi.org/10.1109/FIE56618.2022.9962646 [GS Search].