

## Dicionário de Elementos do Evento

# Programação em Blocos

### Resumo

A Programação em Blocos, também chamada de linguagem de programação visual, é uma abordagem pedagógica voltada ao ensino da lógica computacional que substitui a digitação de código por meio de interfaces gráficas interativas que permitem o desenvolvimento de programas a partir do arraste e encaixe de blocos geométricos, beneficiando principalmente pessoas iniciantes no campo da programação, como crianças e adolescentes, por sua abordagem visual e lúdica (SOMMER et al., 2023).

Brennan e Resnick (2012) caracterizam este como um ambiente de autoria computacional no qual estudantes podem deixar de ser meros consumidores e passar a atuar como pessoas criadoras de mídias interativas, desenvolvendo suas próprias histórias, simulações e jogos digitais. Nessa dinâmica, as pessoas envolvidas projetam encaixando blocos de instrução de programação como fariam ao encaixar peças de um quebra-cabeça.

Dessa forma, o encaixe visual dos blocos apoia a aprendizagem por meio de conceitos estruturados de design (CARDOSO, 2023). Essa mediação gráfica pode minimizar as dificuldades comumente associadas à rigidez sintática e ao nível de abstração exigido pela implementação tradicional de códigos (CARDOSO, 2023).

### Criado por

A origem da Programação em Blocos está relacionada ao trabalho de Seymour Papert, matemático sul-africano que colaborou diretamente com Jean Piaget e, posteriormente, cofundou o MIT Media Lab (RAABE; COUTO; BLIKSTEIN, 2020) (CARDOSO, 2023). Em 1967, em parceria com Cynthia Solomon e Wally Feurzeig, Papert concebeu a linguagem LOGO, considerada o primeiro ambiente de programação voltado especificamente para o público infantil, cujo propósito era tornar a computação uma experiência lúdica e engajadora para as crianças (RAABE; COUTO; BLIKSTEIN, 2020).

A LOGO operou como pioneira do construcionismo ao permitir que as crianças tangibilizassem conceitos por meio do controle de uma tartaruga mecânica programável (ROCHA et al., 2022). Assim, abriu caminho para o desenvolvimento de estruturas de programação mais recentes, entre as quais se destaca o Scratch.

O Scratch foi idealizado por Mitchel Resnick, pesquisador e professor estadunidense do MIT Media Lab, lançado oficialmente em 2007 com o propósito de auxiliar iniciantes no estudo da programação computacional por meio de uma interface baseada em blocos, em substituição às linhas de comando textuais ([SOUZA, 2019](#)).

Na década de 1990, Resnick assumiu a cátedra de LEGO Papert Professor of Learning Research e fundou o grupo de pesquisa Lifelong Kindergarten, dedicado a desenvolver tecnologias voltadas a engajar crianças em experiências criativas de aprendizagem ([SOUZA, 2019](#)). A concepção do Scratch originada nesse grupo foi guiada pelos Quatro P's da Aprendizagem Criativa formulados por [Resnick \(2014\)](#), inspirados no funcionamento do jardim de infância: Projetos (Projects), Paixão (Passion), Pares (Peers) e Pensar Brincando (Play).

## PARA SABER MAIS, ACESSE:

Para conhecer mais sobre Mitchel Resnick e os fundamentos pedagógicos que orientaram a criação do Scratch, assista à palestra: "[Let's teach kids to code](#)" do TEDx Talks.

### Teve sua autoria apropriada por outrem?

No percurso da Programação em Blocos, aconteceram apropriações incentivadas. Ao incorporar as dinâmicas de reutilização e remixagem como práticas computacionais legítimas, ferramentas como o Scratch incentivam estudantes a se apropriarem de códigos desenvolvidos por terceiros na comunidade para a construção de outros sistemas ([BRENNAN; RESNICK, 2012](#)). Essa pluralidade está presente em diversas adaptações como o S4A, o PictoBlox, o Open-Roberta e o Tinkercad, um simulador virtual que combina blocos modulares com a linguagem C para o desenho de microcircuitos ([CARDOSO, 2023](#)). Na esfera dos kits comerciais, a empresa LEGO também utilizou a lógica de quebra-cabeça visual ao estruturar o software do kit LEGO Mindstorms Education EV3 ([PEREZ, 2020](#)).

Essa apropriação é considerada parte do funcionamento saudável da comunidade. [Resnick e colegas \(2009\)](#) relatam que, no início do Scratch, algumas pessoas ficaram incomodadas ao verem seus projetos remixados por outras, sentindo que suas criações estavam sendo "roubadas". Para lidar com essa percepção, os criadores da plataforma

promoveram discussões sobre os valores da cultura *open source*, com o objetivo de fazer as pessoas sentirem orgulho, e não incômodo, com esse processo de apropriação. Para garantir essa liberdade de forma ética, a plataforma foi estruturada sob licença *Creative Commons*. Segundo Resnick (2014), essa lógica sustenta-se no pilar dos Pares (Peers) da Aprendizagem Criativa, que defende que as pessoas devem ter liberdade para se inspirar e construir sobre o trabalho umas das outras, desde que os devidos créditos de autoria sejam atribuídos.

## Quando

O marco inicial do paradigma situa-se em 1967, período de desenvolvimento da linguagem LOGO por Seymour Papert e seus colegas no MIT. Já a consolidação da Programação em Blocos tem como marco o ano de 2007, quando o grupo Lifelong Kindergarten, sob a liderança de Mitchel Resnick, anunciou e iniciou o desenvolvimento do Scratch (RAABE; COUTO; BLIKSTEIN, 2020).

## Onde

Estados Unidos, mais especificamente em Cambridge, Massachusetts, cidade que concentrou os três principais centros de pesquisa responsáveis pelos fundadores da área: o laboratório de inteligência artificial do Massachusetts Institute of Technology (MIT), o MIT Media Lab e o laboratório independente Bolt, Beranek and Newman (BBN) (RAABE; COUTO; BLIKSTEIN, 2020).

## Para/Com Quem

O Scratch, principal representante da Programação em Blocos, está disponível gratuitamente por meio do endereço <https://scratch.mit.edu>. Embora tenha sido concebido especialmente para crianças e adolescentes, a plataforma foi desenhada com o objetivo de tornar a programação mais fácil para todas as pessoas, de todas as idades, origens e interesses, o que resultou na participação ativa também de pessoas adultas na comunidade (RESNICK et al., 2009).

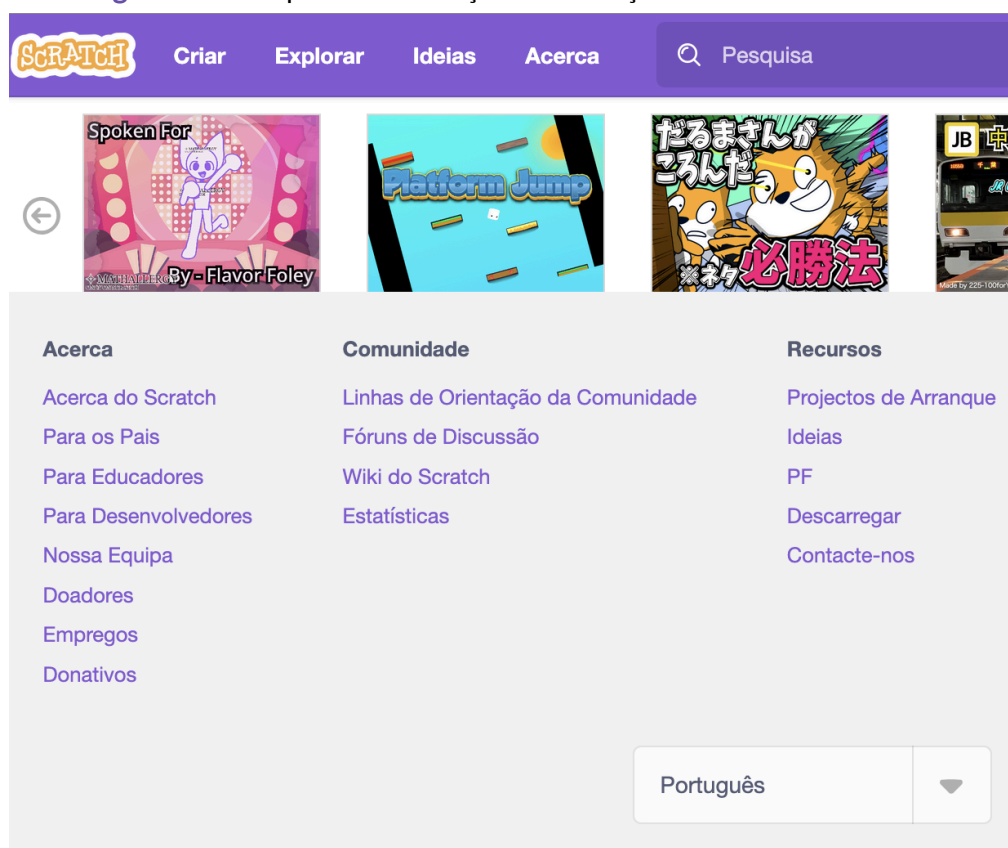
O acesso à plataforma também se mostra flexível quanto ao dispositivo e à conectividade: o uso recomendado é por meio de computador, embora também seja possível acessá-la, com algumas restrições, pelo smartphone (SOUZA, 2019). Da mesma forma, tanto o editor online quanto o offline oferecem funcionalidades equivalentes (SOUZA, 2019).



Esse alcance multiplataforma contribui para a consolidação de uma comunidade internacional de aprendizagem criativa presente em mais de 150 países (MOURA; SANTOS; BARROS, 2025). Para impulsionar esse compartilhamento internacional e a colaboração, Resnick e colegas (2009) relatam que a infraestrutura dos blocos foi projetada para ser traduzida para qualquer idioma por uma rede global de voluntários.

A escolha do idioma para acessar os conteúdos é facilitada, pois basta entrar no site oficial, rolar a tela até o final e então escolher o idioma mais aplicável ao público em questão. A Figura 1 apresenta a tela e o local para a escolha do idioma adequado.

**Figura 1** - Exemplo de localização da definição de idioma no Scratch



Fonte: Pessoas autoras (2025)

## Quem foi excluído

Assim como outras tecnologias educacionais, a Programação em Blocos não está livre de barreiras de acesso. No cotidiano escolar, é comum que estudantes não possuam dispositivo móvel próprio ou acesso à internet, o que agrava a exclusão digital associada



à desigualdade socioeconômica: populações em situação de vulnerabilidade enfrentam acesso precarizado às tecnologias digitais (SILVEIRA, 2001).

Além disso, a exclusão também adquire contornos estruturais, pois, historicamente, a retórica de que os computadores teriam um apelo 'universal' ignorou diferenças de classe, raça, gênero e contexto cultural, privilegiando perfis masculinos, brancos e de alta escolaridade do Norte Global e falhando ao ser transposta para as realidades de outros países (AMES, 2018).

### Vínculo (redes, instituições etc)

O principal vínculo institucional da Programação em Blocos é com a Scratch Foundation, mediada pelo MIT (SOUZA, 2019). No Brasil existem duas instituições que têm se destacado: a Rede Brasileira de Aprendizagem Criativa e a Scratch Brasil.

### PARA SABER MAIS, ACESSE:

SCRATCH BRASIL. *Scratch Brasil*. Disponível em:  
<https://scratchbrasil.org.br/>. Acesso em: 4 out. 2025.

APRENDIZAGEM CRIATIVA. *Aprendizagem Criativa*. Disponível em:  
<https://aprendizagemcriativa.org/>. Acesso em: 4 out. 2025.

SCRATCH FOUNDATION. *Scratch Foundation*. Disponível em:  
<https://www.scratchfoundation.org/>. Acesso em: 4 out. 2025.

### O que resolve (qual problema/necessidade atende)?

Um dos principais obstáculos enfrentados por iniciantes na programação tradicional é a complexidade sintática: a necessidade de memorizar regras rígidas e escrever linhas de código com precisão, o que frequentemente dificulta a compreensão de conceitos de lógica e abstração (CARDOSO, 2023). Para superar essa barreira, Resnick e colegas (2009) defendem a premissa de que uma linguagem educacional deve ter um "baixo piso", ou seja, deve oferecer facilidade imediata para que as pessoas possam iniciar suas

criações. Assim, a Programação em Blocos foi concebida justamente para propiciar que principiantes desenvolvam programas sem a necessidade de aprender a sintaxe de uma linguagem de programação (MOURA; SANTOS; BARROS, 2025).

Ao dispensar a memorização de sintaxes e regras textuais, a Programação em Blocos desloca o foco do aprendizado para a lógica e a resolução de problemas, propiciando que as pessoas aprendizes se concentrem no desenvolvimento do raciocínio lógico e na construção de algoritmos (CARDOSO, 2023). Esse deslocamento se materializa na própria mecânica da interface: em vez de digitar linhas de texto, crianças e iniciantes criam seus programas ao arrastar e encaixar blocos de comando, em uma dinâmica que pode tornar conceitos complexos mais compreensíveis (SOMMER et al., 2023). Essa linguagem gráfica também favorece a identificação das estruturas dos códigos construídos e a percepção da dinâmica entre os comandos por parte de quem programa (CARDOSO, 2023).

### Componente e materiais utilizados

Do ponto de vista material, o uso pedagógico de ferramentas de Programação em Blocos requer uma infraestrutura de *hardware* simples. Para o **Scratch**, recomenda-se o acesso por meio de notebooks ou computadores desktop, sendo permitida a utilização em celulares e tablets sob restrições (SOUZA, 2019). A plataforma é distribuída gratuitamente pelo portal <https://scratch.mit.edu>, que disponibiliza instaladores *offline* para os sistemas operacionais Windows, macOS, ChromeOS e Android (SOUZA, 2019).

Para utilizar todo o potencial da plataforma, incluindo as opções de compartilhar e comentar projetos, é necessário realizar o cadastro e confirmar o e-mail vinculado à conta. Sem essa confirmação, o acesso se restringe à exploração das criações de outras pessoas e à produção de projetos próprios, sem a possibilidade de publicá-los (SOUZA, 2019).

A Figura 2 apresenta a divisão da tela do Scratch em: 1) Blocos disponíveis para programação; 2) Área destinada à programação com os blocos selecionados; 3) Área destinada à simulação que será apresentada quando o conjunto de blocos da área 2 for executado.

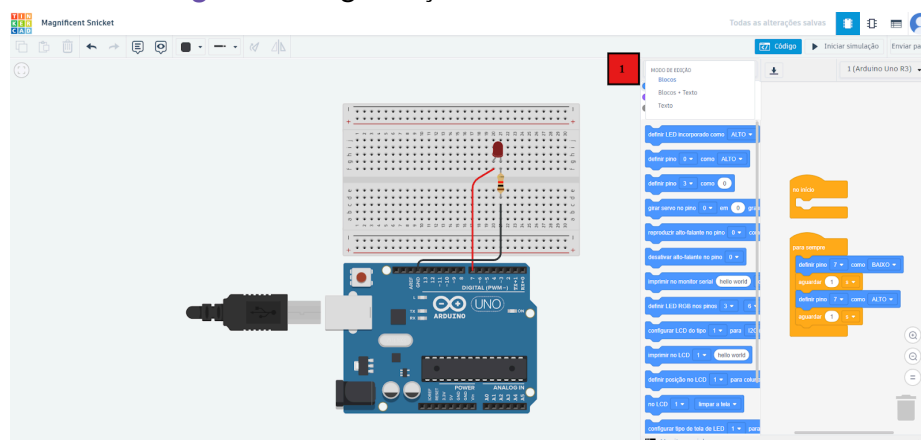
**Figura 2** - Exemplo de Tela e estrutura do Scratch



Fonte: Recurso Educacional Aberto desenvolvido pelo Grupo PET-CoCE (Disponível em: <https://arcaz.ct.utfpr.edu.br/items/show/1468>)

Já o **Tinkercad** (<https://www.tinkercad.com/>) é um ambiente de simulação e prototipação on-line que possibilita a criação de circuitos eletrônicos e a simulação de ambientes virtuais de programação, interligando diferentes dispositivos eletrônicos, protoboards e placas de desenvolvimento como o Arduino e o BBC Micro:bit (Figura 3) (CARDOSO, 2023). Nesse ambiente, os materiais utilizados são virtualizados: componentes como LEDs, resistores e jumpers são emulados diretamente na tela do computador, o que viabiliza o acesso a esses recursos sem exigir investimento na aquisição de kits físicos (CARDOSO, 2023).

**Figura 3** - Programação em blocos dentro do Tinkercad



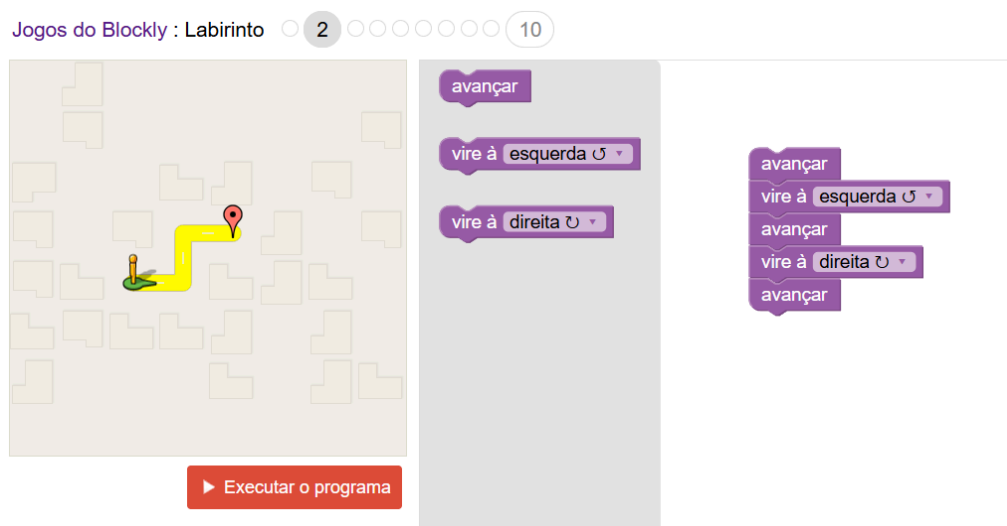
Fonte: Pessoas autoras (2025)



Entre as variações voltadas à integração direta com hardware, o **PictoBlox** (<https://pictoblox.ai/>) se destaca por incorporar a estrutura da plataforma Scratch, permitindo tanto a programação em blocos quanto a interação com diversas placas de desenvolvimento, como o Arduino. Sua interface, baseada no Scratch 3.0, preserva as funcionalidades de arrastar e soltar proporcionadas pela programação em blocos (CARDOSO, 2023).

O **Blockly Games**, por outro lado, é uma variação voltada à resolução de desafios lógicos. Um conjunto de jogos organizados em sequências de etapas que devem ser completadas com o auxílio da programação em blocos. Está disponível gratuitamente em <https://blockly.games> (CARDOSO, 2023).

**Figura 4** - Programação em blocos do Blockly.games



Fonte: Pessoas autoras (2025)

### Aspectos inovadores

Considerando que inovar implica em mudar a forma de pensar/fazer anterior, por meio de uma proposição entendida como melhor, a origem da Programação em Blocos está atrelada a aspectos como a minimização de barreiras de aprendizagem características da programação tradicional, como a necessidade de reconhecer e memorizar a escrita de todos os comandos de uma linguagem (CARDOSO, 2023). Ao substituir a digitação por uma interface visual, essas plataformas reduzem a exigência de memorização de sintaxes e regras características das demais linguagens de programação (CARDOSO, 2023).

Para Resnick (2013), a transposição dessa barreira técnica representa uma inovação paradigmática: permite deslocar o foco do "aprender a programar" para a concepção do



"programar para aprender". Assim como na escrita tradicional, a programação deixa de ser uma técnica isolada para se tornar uma nova forma de aprendizado e de expressão criativa, através da qual a pessoa organiza, refina e reflete sobre suas ideias.

Essa inovação se manifesta concretamente na interação de arrastar e inserir comandos em áreas determinadas da interface, o que aproxima o ato de programar de uma experiência tangível e manipulável ([SOMMER et al., 2023](#)). Apoiando-se na filosofia dos Quatro P's da Aprendizagem Criativa idealizada por [Resnick \(2014\)](#) (Projetos, Paixão, Pares e Pensar Brincando), essa inovação visual estimula a coautoria, a criatividade e a resolução de problemas complexos de forma colaborativa.

### Princípio de funcionamento

O princípio de funcionamento da Programação em Blocos baseia-se em uma interface gráfica, na qual as linhas de código são substituídas por blocos modulares que representam funções e comandos computacionais ([CARDOSO, 2023](#)). A dinâmica de operação consiste na ação de clicar, arrastar e soltar esses blocos em uma área de edição, conectando-os uns aos outros de forma similar ao encaixe de peças de um quebra-cabeça ([SOUZA, 2019](#)) ([RESNICK et al., 2009](#)).

Cada bloco contém uma instrução predefinida. Para facilitar o uso, as próprias formas geométricas dos conectores sugerem visualmente como eles devem ser organizados: [Resnick e colegas \(2009\)](#) detalham que estruturas de controle (como os laços de repetição) têm formato em "C" para indicar que outros comandos devem ser posicionados em seu interior, enquanto os blocos que retornam valores assumem formas ovais (para números) ou hexagonais (para booleanos).

Além das formas geométricas, ferramentas como Scratch, PictoBlox e LEGO EV3 costumam categorizar esses blocos por tipo e natureza, como: movimento, aparência, controle de fluxo, eventos, operadores lógicos, sensores, frequentemente agrupando-os por cores ([PEREZ, 2020](#)) ([SOUZA, 2019](#)). O funcionamento exige que quem usa encadeie os blocos em uma sequência lógica, geralmente iniciada por um "bloco de evento", que atua como ponto de partida, determinando a condição para que a ação comece a ser executada pelo sistema ([SOUZA, 2019](#)).

Segundo [Resnick e colegas \(2009\)](#), o formato de encaixe convida à experimentação investigativa, permitindo que iniciantes combinem e testem diferentes sequências de forma lúdica. Uma vez estruturado, o encadeamento de blocos forma um algoritmo capaz de operar em duas vertentes. A primeira ocorre inteiramente no ambiente de software, simulando comportamentos de atores virtuais e cenários interativos

diretamente na tela do dispositivo ([SOUZA, 2019](#)). A segunda se dá a partir da integração com hardware físico, acionamento de motores, luzes e leitura de sensores, a partir da transferência da lógica em blocos para placas e microcontroladores, como o Arduino ou outra central robótica ([CARDOSO, 2023](#))([PEREZ, 2020](#)).

### Teve sentido/função inicial alterada?

A função original da Programação em Blocos esteve voltada a facilitar o ensino de lógica computacional para crianças e iniciantes a partir de uma interface visual que dispensasse a exigência da sintaxe textual ([CARDOSO, 2023](#)) ([SOMMER et al., 2023](#)).

Ao longo do tempo, essa função foi ampliada: a Programação em Blocos passou a ser empregada também na prototipagem rápida de circuitos, na Robótica Educacional e no desenvolvimento de jogos simples. O próprio Scratch, originalmente voltado à criação de histórias animadas, projetos de ciência, jogos e simulações, hoje dialoga com ambientes de prototipagem como o Tinkercad e com softwares de controle de placas e microcontroladores como o PictoBlox, ampliando seu alcance da tela do computador para a robótica física ([CARDOSO, 2023](#)). Do ponto de vista educacional, também passou a ser utilizado no ensino superior, especialmente em disciplinas iniciais ou para testar protótipos antes de sua implementação em *hardware*.

### Aplicabilidade contemporânea

A Programação em Blocos está cada vez mais presente no cenário educacional brasileiro, entre estudantes de diferentes idades ([CARDOSO, 2023](#)). Isso dá pelo que [Resnick e colegas \(2009\)](#) definem como o "teto alto" e "paredes largas": a capacidade do paradigma de suportar desde programas simples até a criação de projetos cada vez mais complexos ao longo do tempo, abarcando uma ampla diversidade de interesses para manter diferentes perfis de pessoas estudantes engajadas.

Na robótica e na automação, essa linguagem visual se une a componentes eletrônicos de baixo custo, tornando essa área mais acessível. Ferramentas como o S4A (Scratch for Arduino), o Tinkercad Circuits e o PictoBlox utilizam blocos de programação para facilitar o controle de placas eletrônicas abertas ([CARDOSO, 2023](#)). Essa mesma lógica também é usada na criação de aplicativos para celular por meio do MIT App Inventor, que permite desenvolver apps para Android e iOS usando programação em blocos ([CARDOSO, 2023](#)).

## Desdobramentos socioambientais

Os desdobramentos socioambientais da Programação em Blocos podem ser observados nas oficinas pedagógicas que unem seu uso a materiais recicláveis, funcionando como uma tentativa de reduzir barreiras cognitivas e econômicas ao acesso ao conhecimento digital (MILL; CÉSAR, 2009) (CARDOSO, 2023). Isto dialoga com o propósito original de Resnick e colegas (2009) ao criarem o Scratch: democratizar a autoria de mídias interativas, buscando que qualquer pessoa possa atuar ativamente na criação digital, independentemente de sua condição econômica.

Essa mesma lógica de acesso se estende ao enfrentamento dos impactos ambientais decorrentes do descarte acelerado de equipamentos tecnológicos. Ao integrar componentes eletrônicos reaproveitados a placas controladoras programáveis por blocos, como o S4A e o Picto Blox, projetos de robótica sustentável reutilizam motores e sensores descartados, promovendo debates sobre consumo consciente e conscientizando estudantes sobre os riscos do descarte inadequado de lixo eletrônico (ROCHA et al., 2022).

Essa combinação entre programação visual, cultura do tinkering (aprendizagem por experimentação e bricolagem) (RESNICK et al., 2009) e reaproveitamento de materiais mostra como a Programação em Blocos pode ir além do ensino de lógica computacional, servindo também como ferramenta de reflexão e de engajamento com problemas socioambientais reais.

## Desdobramentos científico-tecnológicos

Sob a ótica científico-tecnológica, a Programação em Blocos pode ser vista como uma abordagem para preparar estudantes para o entendimento de outras linguagens de programação por meio do ensino lúdico (CARDOSO, 2023). Esse processo de ganho de complexidade materializa o princípio arquitetônico do "teto alto" discutido por Resnick e colegas (2009), que defende que uma linguagem educacional, além de ser fácil para iniciantes, deve obrigatoriamente oferecer suporte estrutural para que as pessoas criem projetos cada vez mais avançados com o passar do tempo.

Esse desdobramento se conecta diretamente à automação e à robótica. Os projetos estudantis que operam com microcontroladores de baixo custo utilizam blocos visuais para estruturar o controle lógico de dispositivos físicos, introduzindo conceitos fundamentais de mecânica e engenharia computacional (CARDOSO, 2023) (SILVA FILHO, 2019).

Essa infraestrutura de código aberto e materiais de baixo custo insere estudantes nas áreas STEM (*Science, Technology, Engineering, and Mathematics*), apresenta possibilidades para gerações de futuras pessoas cientistas, matemáticas e engenheiras a partir de metodologias de aprendizagem criativa (CAMPOS, 2017)(SOMMER et al., 2023). Contudo, aqui vale reafirmar a reflexão de Resnick e colegas (2009) sobre essa formação: embora a Programação em Blocos possa fomentar o ingresso nas áreas STEM, seu objetivo primordial deve transcender esse objetivo. Para os autores, o letramento tecnológico não visa formar profissionais de programação para o mercado de trabalho, mas nutrir uma geração de pessoas pensadoras criativas e sistemáticas, aptas a utilizar o computador para expressar suas ideias e resolver problemas em qualquer futura área de atuação.

### **Desdobramentos sócio-político-culturais**

Para discutir seus desdobramentos sociopolítico-culturais, é importante destacar que a Programação em Blocos não nasceu com o propósito de formar programadores profissionais para o mercado de trabalho. Resnick e colegas (2009) defendem desde o início que o principal objetivo do Scratch não é preparar para carreiras técnicas, mas cultivar uma geração de pessoas pensadoras criativas, capazes de usar a programação como meio de expressar suas próprias ideias.

Assim, este paradigma evidencia o confronto entre a chamada cultura *hacker* ou *maker* e a tecnológica tradicional, baseada em *softwares* e equipamentos fechados. Barbosa e Silva (2012) apontam que muitas escolas utilizam tecnologias como caixas-pretas, sem perceber o quanto isso limita a criatividade dos estudantes, defendendo que um ensino de robótica consciente deve abri-las para que o ser humano use a tecnologia a seu favor, em vez de se tornar dependente dela.

Essa mesma lógica se manifesta no movimento da Robótica Livre, que utiliza a Programação em Blocos e o acesso ao código aberto como forma de resistência à exclusão digital. Mill e César (2009) argumentam que dar acesso ao código-fonte forma criadores capazes de intervir na ordem tecnológica estabelecida, ampliando a autonomia das comunidades sobre as ferramentas que utilizam.

Além disso, a Programação em Blocos contribui para o desdobramento da própria noção de propriedade intelectual, substituindo a lógica da apropriação indevida pela cultura do compartilhamento. Resnick e colegas (2009) relatam que, no início da comunidade Scratch, algumas pessoas da comunidade se sentiram incomodadas ao verem seus projetos remixados por outras, o que gerou debates nos fóruns da

plataforma sobre os valores das comunidades de código aberto, o que levou à adoção da licença *Creative Commons* para todos os projetos compartilhados.

Brennan e Resnick (2012) complementam que o próprio ato de reutilizar e remixar códigos estimula uma leitura crítica da programação, levantando questões relevantes sobre o que é razoável tomar emprestado de outra pessoa e como atribuir os créditos de forma adequada. Souza (2019) reforça esse entendimento ao explicar que remixar, no Scratch, significa copiar, modificar e adaptar a criação de outra pessoa conforme os próprios interesses, prática não apenas permitida, mas incentivada pela plataforma, já que tudo o que é publicado fica disponível para visualização, uso e remixagem por qualquer pessoa da comunidade.

### **Implicações futuras (relação com outros eventos/ inovações)**

Resnick e colegas (2009) defendem que a Programação em Blocos deve ter "paredes largas": suportar uma multiplicidade de projetos para engajar estudantes com diferentes interesses e estilos de aprendizagem. Assim, podemos considerar que o futuro deste paradigma está relacionado à expansão para diferentes campos do saber, assim como já acontece nas áreas das ciências humanas e linguísticas, como Língua Portuguesa, História e Geografia (CARDOSO, 2023).

Além disso, a Programação em Blocos tem se consolidado como um dos principais caminhos para introduzir a Inteligência Artificial e o Aprendizado de Máquina desde a infância. Por meio de ferramentas como o PictoBlox, estudantes conseguem manipular e compreender tecnologias como visão computacional, reconhecimento de voz, análise facial e digitalização de texto (OCR), tornando o ato de programar uma atividade mais criativa (CRUZ; BENTO; LENCASTRE, 2021) (CARDOSO, 2023).

Essas implicações apontam para a possibilidade de um perfil de estudante com maior autonomia cognitiva, capacidade de se expressar criticamente e competência para lidar com problemas complexos. Essa ideia se aproxima do conceito de "fluência digital" proposto por Resnick e colegas (2009), segundo o qual jovens deixam de apenas usar as tecnologias e passam a projetar, criar e remixar mídias, tornando-se sujeitos ativos no desenvolvimento tecnológico.

Entretanto, vale ressaltar que para que as ferramentas de programação visual cumpram um papel emancipatório, seus desdobramentos futuros precisam se afastar da lógica de apenas formar mão de obra para o mercado. O foco deve estar na formação humana mais ampla e no letramento digital crítico, preparando as novas gerações para questionar os sistemas técnicos, contestar estruturas hegemônicas de

desenvolvimento e usar essas ferramentas para resolver problemas reais de suas comunidades (MILL; CÉSAR, 2009)(RESNICK et. al., 2009).

### **E no Brasil? Como é no nosso contexto?**

No cenário nacional, o uso da programação em blocos vem se expandindo. Um estudo publicado em 2017 por pesquisadores da USP na revista RENOTE (ELOY; LOPES; ANGELO, 2017) apontou que, em 2016, cerca de 276 mil brasileiros usavam a plataforma Scratch. Esse número tende a crescer com as determinações da Base Nacional Comum Curricular (BNCC), que oficializou a "Cultura Digital" como uma das competências gerais obrigatórias, prevendo o desenvolvimento do pensamento computacional e de conceitos algorítmicos desde os anos iniciais do ensino (CARDOSO, 2023) (MACHADO; AMARAL, 2021). Segundo dados mais recentes do próprio site do Scratch (SCRATCH STATISTICS, 2023), o Brasil já ultrapassou a marca de 8 milhões de usuários cadastrados, sendo um dos países com maior participação na plataforma.

No entanto, a literatura nacional aponta um descompasso entre as metas normativas e a infraestrutura das redes públicas. Ainda que abordagens baseadas em código aberto e na reutilização de sucatas busquem reduzir desigualdades econômicas e democratizar o aprendizado (MILL; CÉSAR, 2009), a prática docente esbarra com frequência em laboratórios de informática precarizados, na falta de conectividade estável e na escassez de computadores funcionais, o que transforma a inovação tecnológica em uma possibilidade teórica que enfrenta restrições para se concretizar na prática (CARDOSO, 2023).

### **Saberes relacionados**

De acordo com Resnick e colegas (2009), à medida que criam e compartilham projetos na plataforma, estudantes assimilam conceitos matemáticos e computacionais e desenvolvem o raciocínio sistemático. Além disso, recorrem a conceitos de biologia e história para parametrizar simulações de ecossistemas, modelar dinâmicas naturais ou construir metáforas narrativas que ajudam a explicar fenômenos abstratos da própria computação (SILVA FILHO, 2019).

Essas ferramentas ainda se relacionam com saberes das ciências humanas, das artes e da linguística. Isso porque a criação de mídias interativas e jogos digitais envolve, ao mesmo tempo, a estruturação de roteiros, a expressão artística na edição de cenários, a ordenação gramatical e o design de personagens, o que torna a programação em blocos

útil para o desenvolvimento da linguagem oral, escrita e estética na educação (SOMMER *et al.*, 2023) (CARDOSO, 2023).

É justamente por essa relação com a linguagem que Resnick (2013) defende o conceito de "programar para aprender", comparando-o à aprendizagem da escrita: assim como, na escrita tradicional, as pessoas aprendem a organizar, refinar e refletir sobre suas ideias, a codificação visual permite que os estudantes "escrevam" novas mídias e expressem sua criatividade no mundo digital.

### **Relação com o currículo e com a BNCC**

A Programação em Blocos Visuais é uma porta de entrada para o desenvolvimento do pensamento computacional no contexto escolar, ao viabilizar que estudantes manipulem estruturas lógicas (como sequências, laços de repetição, condicionais e eventos), sem a barreira da sintaxe textual própria das linguagens tradicionais (CARDOSO, 2023) (SOMMER *et al.*, 2023). Essa mediação favorece uma apropriação gradual dos conceitos computacionais, ao mesmo tempo em que sustenta práticas pedagógicas voltadas à autoria, à colaboração e à resolução situada de problemas (CARDOSO, 2023) (SOUZA, 2019).

No Brasil, essa relação se formaliza principalmente por meio da Base Nacional Comum Curricular - BNCC (BRASIL, 2017) e da BNCC-Computação (BRASIL, 2022). Neste contexto, a Programação em Blocos aparece como recurso metodológico, sobretudo nos anos iniciais e finais do Fundamental, momento em que a abstração textual ainda representa um obstáculo cognitivo e a interface de blocos propicia que crianças e adolescentes se concentrem na lógica da solução, e não na memorização de comandos (CARDOSO, 2023) (SOMMER *et al.*, 2023).

As habilidades mobilizadas dialogam com as três dimensões do pensamento computacional propostas por Brennan e Resnick (2012): os conceitos computacionais (o vocabulário lógico mobilizado pelos blocos), as práticas computacionais (o processo iterativo de testar, depurar e reaproveitar código) e as perspectivas computacionais (a forma como o estudante passa a se expressar, a se conectar e questionar o mundo por meio da tecnologia).

Assim, a proposta de seleção a seguir identifica as habilidades da BNCC-Computação (BRASIL, 2022) que podem ser diretamente articuladas com as práticas de Programação em Blocos no contexto escolar:

(EF06CO01) Classificar informações, agrupando-as em coleções (conjuntos) e associando cada coleção a um "tipo de dados".

(EF06CO02) Elaborar algoritmos que envolvam instruções sequenciais, de repetição e de seleção usando uma linguagem de programação.

(EF06CO03) Descrever com precisão a solução de um problema, construindo o programa que implementa a solução descrita.

### 7º Ano

(EF07CO01) Criar soluções de problemas para os quais seja adequado o uso de registros e matrizes unidimensionais para descrever suas informações e automatizá-las usando uma linguagem de programação.

(EF07CO02) Analisar programas para detectar e remover erros, ampliando a confiança na sua correção.

(EF07CO03) Construir soluções computacionais de problemas de diferentes áreas do conhecimento, de forma individual e colaborativa, selecionando as estruturas de dados e técnicas adequadas, aperfeiçoando e articulando saberes escolares.

(EF07CO04) Explorar propriedades básicas de grafos.

### 8º Ano

(EF08CO01) Construir soluções de problemas usando a técnica de recursão e automatizar tais soluções usando uma linguagem de programação.

(EF08CO02) Criar soluções de problemas para os quais seja adequado o uso de listas para descrever suas informações e automatizá-las usando uma linguagem de programação, empregando ou não a recursão como técnica de resolução.

(EF08CO03) Utilizar algoritmos clássicos de manipulação sobre listas.

(EF08CO04) Construir soluções computacionais de problemas de diferentes áreas do conhecimento, de forma individual e colaborativa, selecionando as estruturas de dados e técnicas adequadas, aperfeiçoando e articulando saberes escolares.

### 9º Ano



(EF09CO01) Criar soluções de problemas para os quais seja adequado o uso de árvores e grafos para descrever suas informações e automatizá-las usando uma linguagem de programação.

(EF09CO02) Construir soluções computacionais de problemas de diferentes áreas do conhecimento, de forma individual e colaborativa, selecionando as estruturas de dados e técnicas adequadas, aperfeiçoando e articulando saberes escolares.

(EF69CO01) Classificar informações, agrupando-as em coleções (conjuntos) e associando cada coleção a um "tipo de dado".

(EF69CO02) Elaborar algoritmos que envolvam instruções sequenciais, de repetição e de seleção usando uma linguagem de programação.

(EF69CO03) Descrever com precisão a solução de um problema, construindo o programa que implementa a solução descrita.

(EF69CO04) Construir soluções de problemas usando a técnica de decomposição e automatizar tais soluções usando uma linguagem de programação.

### Ensino Médio

(EM13CO04) Reconhecer o conceito de metaprogramação como uma forma de generalização na construção de programas, permitindo que algoritmos sejam entrada ou saída para outros algoritmos.

### **Quais as políticas deste artefato? Categorias CTS, tais como: não neutralidade, não determinismo tecnológico, não linearidade e participação**

Os estudos em Ciência, Tecnologia e Sociedade (CTS) oferecem bases teóricas para compreender os aspectos sociais envolvidos no desenvolvimento científico-tecnológico, contestando visões naturalizadas de uma ciência e tecnologia neutras, deterministas e lineares. Relacionada à Programação em Blocos, a primeira categoria CTS relevante é a não neutralidade, que diz respeito ao entendimento de que toda tecnologia carrega dimensões políticas e éticas, refletindo as assimetrias e as visões de mundo de seu contexto cultural e de seus grupos de desenvolvimento, o que impede tratar os sistemas computacionais como ferramentas puramente técnicas e livres de vieses (WINNER, 2017).

Essa ausência de neutralidade se manifesta nas próprias escolhas de arquitetura do Scratch: para Resnick e colegas (2009), a plataforma foi intencionalmente projetada para carregar e disseminar os valores da cultura open source e da colaboração. Ao adotar licenças Creative Commons e voltar o design da interface ao compartilhamento e à remixagem de códigos, a ferramenta expressa uma intenção política de democratização e emancipação tecnológica, em oposição a sistemas pautados pela lógica da competitividade e do fechamento do conhecimento.

Nessa linha, a categoria de não determinismo tecnológico contesta a perspectiva que trata a tecnologia como uma força autônoma, capaz de guiar sozinha as mudanças da sociedade (LINSINGEN; BAZZO; PEREIRA, 2003). Para romper com esta ideia, Resnick (2014) defende que o ecossistema tecnológico deve estar conectado a uma dinâmica social humana, mediada pelos Quatro Ps da Aprendizagem Criativa: Projetos, Paixão, Pares e Pensar Brincando. Nessa mesma direção, ao afirmar que o propósito da linguagem não é atender à demanda do mercado por programadores profissionais, mas formar pessoas criativas, Resnick e colegas (2009) mostram que o rumo do desenvolvimento tecnológico pode ter a intenção de se desviar das rotas industriais hegemônicas.

Além disso, a categoria de não linearidade rejeita a ideia do modelo linear de desenvolvimento, segundo o qual o desenvolvimento científico e tecnológico se traduziria automaticamente em progresso econômico e bem-estar social, defendendo que, ao contrário, este desenvolvimento admite interpretações e caminhos diferentes, o que mostra que as escolhas da sociedade por determinadas alternativas técnicas são guiadas por dinâmicas de poder e valores, e não apenas por critérios técnicos (LINSINGEN; BAZZO; PEREIRA, 2003)(FEENBERG, 2022).

Essa categoria dialoga com o alerta de Resnick e colegas (2009) de que a simples exposição ou o acesso a artefatos digitais não garante melhorias intelectuais ou sociais. O que eles chamam de "fluência digital" autêntica, por exemplo, não se resume à interação passiva com as máquinas, como navegar, usar aplicativos ou conversar em rede. Para que a sociedade supere essa falsa promessa a respeito da tecnologia, Resnick, Berg e Eisenberg (2000) defendem que a sociedade deve romper as "caixas-pretas" impostas pelo mercado e assumir a posição de "caixa-branca", na qual as pessoas compreendem as lógicas por trás da tecnologia e atuam como "fazedores", intervindo de forma crítica na resolução dos problemas de seu próprio contexto social.

Esse último argumento também se relaciona à categoria CTS da participação (LINSINGEN; BAZZO; PEREIRA, 2003). No caso da Programação em Blocos, esse princípio se materializa na própria estrutura da comunidade Scratch, pensada não apenas como espaço técnico, mas como rede de apoio, colaboração e troca entre pares.

Resnick e colegas (2009) explicam que, para o Scratch cumprir seu propósito, a linguagem precisa estar vinculada a uma comunidade em que as pessoas possam apoiar, colaborar e criticar o trabalho umas das outras, construindo conhecimento de forma conjunta.

Ao discutir o "P" de Pares (Peers) dentro dos Quatro P's da Aprendizagem Criativa, o autor (2014) complementa que essa comunidade cumpre dois papéis centrais: funciona como público, oferecendo feedback e sugestões quando pessoas compartilham seus projetos, e como fonte de inspiração, na medida em que o contato com os projetos de demais participantes gera novas ideias. Brennan e Resnick (2012) reforçam essa perspectiva ao afirmarem que a criatividade e a aprendizagem são práticas profundamente sociais, de modo que a criação de mídias computacionais no Scratch se enriquece justamente pelas interações entre as pessoas envolvidas.

Essa dimensão participativa também aparece no uso da plataforma como espaço de expressão: Resnick e colegas (2009) relatam que muitos projetos são criados para chamar a atenção para problemas sociais, como aquecimento global e maus-tratos a animais. Esses exemplos mostram que, quando a participação é estruturalmente incentivada, a Programação em Blocos pode deixar de ser apenas uma ferramenta de ensino técnico e passar a funcionar como espaço de voz coletiva e engajamento social.

## Referências

AMES, Morgan G. Hackers, Computers, and Cooperation: A Critical History of Logo and Constructionist Learning. **Proceedings of the ACM on Human-Computer Interaction**, v. 2, n. CSCW, art. 18, novembro, 2018.

BARBOSA E SILVA, Rodrigo. **Abordagem crítica de robótica educacional**: Álvaro Vieira Pinto e Estudos de Ciência, Tecnologia e Sociedade. 2012. 145 f. Dissertação (Mestrado em Tecnologia) - Universidade Tecnológica Federal do Paraná, Curitiba, 2012.

BRASIL. Ministério da Educação. Secretaria da Educação Básica. Base Nacional Comum Curricular (BNCC). Brasília: MEC, CONSED, UNDIME, 2017.

BRASIL. Ministério da Educação. Secretaria da Educação Básica. Base Nacional Comum Curricular - Computação. Anexo ao Parecer CNE/CEB nº 2/2022, aprovado em 17 de fevereiro de 2022. Brasília. DF. 2022.

BRENNAN, Karen; RESNICK, Mitchel. Using artifact-based interviews to study the development of computational thinking in interactive media design. In: **Annual**

**American Educational Research Association meeting**, Vancouver, BC, Canada. 2012. p. 1-25.

CAMPOS, Flavio Rodrigues. Robótica educacional no Brasil: questões em aberto, desafios e perspectivas futuras. In: **Revista ibero-americana de estudos em educação**, v. 12, n. 4, p. 2108-2121, 2017.

CARDOSO, Carlos Costa. **Robótica educacional e pensamento computacional como uma metodologia para o ensino de algoritmos e programação**. 2023. 116 f. Dissertação (Mestrado em Ensino de Ciências e Matemática) – Universidade de Passo Fundo, Passo Fundo, 2023.

CRUZ, Sara; BENTO, Marco; LENCASTRE, José Alberto. Computational thinking training using Pictoblox: exploratory study with students of primary degree. In: KOMMERS, Piet et al. (Ed.). In: **Proceedings of International Conferences Internet Technologies & Society**. 2021. p. 53-60.

ELOY, Adelmo Antonio da Silva; LOPES, Roseli de Deus; ANGELO, Isabela Martins. Uso do Scratch no Brasil com objetivos educacionais: uma revisão sistemática. **RENOTE. Revista Novas Tecnologias na Educação**, Porto Alegre, v. 15, n. 1, 2017.

FEENBERG, Andrew. **Construtivismo crítico: uma filosofia da tecnologia**. São Paulo: Scientia Studia, 2022.

FRANCISCO JÚNIOR, Nacim Miguel. **Diálogos entre a robótica educacional e a sala de aula: um estudo de caso**. 2009. Dissertação (Mestrado em Educação) – Universidade do Sul de Santa Catarina, Tubarão, 2009.

LINSINGEN, Irlan von; BAZZO, Walter A.; PEREIRA, Luiz TV. **O que é ciência, tecnologia e sociedade**. In: BAZZO, Walter A., PALACIOS, Eduardo Marino García; GALBARTE, Juan Carlos González. Introdução aos estudos CTS (Ciência, Tecnología e Sociedade). Brasília, Brasil: Organización de Estados Iberoamericanos (OEI), p. 119-156, 2003.

MACHADO, Aline Alvares; AMARAL, Marília Abrahão. Uma análise crítica da competência cultura digital na Base Nacional Curricular Comum. **Ciência & Educação**, Bauru, v. 27, e21034, 2021.

MILL, Daniel; CÉSAR, Danilo. Robótica pedagógica livre: sobre inclusão sócio-digital e democratização do conhecimento. **Perspectiva**, v. 27, n. 1, p. 217-248, 2009.



MORAES, Maria Candida. Informática educativa no Brasil: um pouco de história. Brasília, v. 12, p. 17-26, 1993.

MOURA, Janya Valéria da Silva; SANTOS, Claudimary Bispo dos; BARROS, Rubens Pessoa de. Integração do Scratch à proposta pedagógica de genética. **Revista Ibero-Americana de Humanidades, Ciências e Educação**, São Paulo, v. 11, n. 8, p. 1664-1681, 2025.

PEREZ, Carla Fernanda da Silva. **Robótica educacional**: uma possibilidade no ensino da Matemática. 2020. Trabalho de Conclusão de Curso (Especialização em Tecnologia, Comunicação e Técnicas de Ensino) - Universidade Tecnológica Federal do Paraná, Curitiba, 2020.

RAABE, André Luís Alice; COUTO, Natália Ellery Ribeiro; BLIKSTEIN, Paulo. Diferentes abordagens para a computação na educação básica. In: RAABE, André; ZORZO, Avelino; BLIKSTEIN, Paulo (Org.). **Computação na educação básica: fundamentos e experiências**. Porto Alegre: Penso, 2020.

RESNICK, Mitchel. **Learn to code, code to learn**. TED Talks Education, 2013. Disponível em: [https://www.ted.com/talks/mitch\\_resnick\\_let\\_s\\_teach\\_kids\\_to\\_code](https://www.ted.com/talks/mitch_resnick_let_s_teach_kids_to_code). Acesso em: 08 jul. 2026.

RESNICK, Mitchel. Give P'sa chance: Projects, peers, passion, play. In: **Constructionism and creativity**: Proceedings of the third international constructionism conference. Austrian computer society, Vienna. 2014. p. 13-20.

RESNICK, Mitchel; FLANAGAN, Mary; KELLEHER, Caitlin; MACLAURIN, Matthew; OHSHIMA, Yoshiki; PERLIN, Ken; TORRES, Robert. Growing up programming: democratizing the creation of dynamic, interactive media. In: **CHI'09 EXTENDED ABSTRACTS ON HUMAN FACTORS IN COMPUTING SYSTEMS**, 2009, Boston. Anais [...]. Nova York: ACM, 2009. p. 3293-3296.

RESNICK, Mitchel; MALONEY, John; MONROY-HERNÁNDEZ, Andrés; RUSK, Natalie; EASTMOND, Evelyn; BRENNAN, Karen; MILLNER, Amon; ROSENBAUM, Eric; SILVER, Jay; SILVERMAN, Brian; KAFAI, Yasmin. Scratch: Programming for all. **Communications of the ACM**, v. 52, n. 11, p. 60-67, nov. 2009.

RESNICK, Mitchel; BERG, Robbie; EISENBERG, Michael. Beyond black boxes: bringing transparency and aesthetics back to scientific investigation. **The journal of the Learning Sciences**, v. 9, n. 1, p. 7-30, 2000.

ROCHA, Maria do Carmo Santos; BARROS, Aparecida da Silva Xavier; ROCHA, Petterson Santos; BARROS, Bruno Xavier; SANTOS, Deric Vinicius dos. A robótica sustentável como estratégia no ensino de Ciências. *Revista Brasileira de Ensino de Ciências e Matemática, Passo Fundo*, v. 5, edição especial, p. 209-221, 2022.

SCRATCH. Statistics. Cambridge, MA: MIT Media Lab / Scratch Foundation, 2023. Disponível em: <https://scratch.mit.edu/statistics/>. Acesso em: 08 jul. 2026.

SILVA FILHO, Fernando Barros da. **Fundamentos da robótica educacional: desenvolvimento, concepções teóricas e perspectivas**. 2019. 175 f. Dissertação (Mestrado em Educação) – Universidade Federal do Ceará, Fortaleza, 2019.

SILVEIRA, Sérgio Amadeu. *Exclusão digital: a miséria na era da informação*. Editora Fundação Perseu Abramo, 2001.

SOMMER, Gustavo de Faria; CORRÊA, Fábio; DUTRA, Frederico Giffoni de Carvalho; RIBEIRO, Jurema Suely de Araújo Nery; REZENDE, Josiane da Costa Vieira. Metodologias de ensino no aprendizado de programação para crianças: uma perspectiva da literatura científica. **SCIAS. Educação, Comunicação e Tecnologia**, [S. l.], v. 5, n. 2, p. 03–20, 2023.

SOUZA, Michel Figueiredo de. **Ensino e tecnologias: aprendizagem significativa por meio da plataforma Scratch**. 2019. 78 p. Produto Educacional (Mestrado Profissional em Práticas de Educação Básica) - Colégio Pedro II/PROPGPEC, Rio de Janeiro, 2019.

WINNER, Langdon. **Do artifacts have politics?**. In: *Computer ethics*. Routledge, 2017. p. 177-192.

### **Ficha Técnica**

CC BY-SA 4.0: O trabalho está licenciado com uma Licença Creative Commons - Atribuição 4.0 Internacional. (<https://creativecommons.org/licenses/by-sa/4.0/deed.pt-br>)

### **Produção**

Ana Júlia Perez Machado

Bruna Medeiros Teixeira

Caio Gomes de Oliveira Barbosa

Damon Lopes

Enzo Westphal Tacla

Gabriel Dutra Amaral

Giovane de Oliveira Brandino

Kailaine Cristina da Silva Miguel

Kelly Dayane Aguiar

Larissa Paschoalin

Leonelo Dell Anhol Almeida

Maria Clara Diniz Almeida

Marília Abrahão Amaral

Mayla Eloize de Farias Monclaro

William Coelho De Araujo Fagundes

Universidade Tecnológica Federal do Paraná – Câmpus Curitiba

Programa de Pós-Graduação em Tecnologia e Sociedade - PPGTE

Departamento Acadêmico de Informática - DAINF

### **Grupo de Pesquisa Xuê: Participação, Interação e Computação**

Líderes: Marília Abrahão Amaral / Luiz Ernesto Merkle

### **Programa de Educação Tutorial Computando Culturas em Equidade**

Tutora: Marília Abrahão Amaral



**Projeto Inclu-e - CAAE: 74791623.1.0000.5547 - 12.set.2024**

Coordenação: Leonelo Dell Anhol Almeida / Marília Abrahão Amaral