Bruno Selau Gonçalves
Lucia Giraffa

CRIANDO UMA OFICINA DE SCRATCH

Relatório Técnico

EDITORA & LIVRARIA
VECHER EDUCOM

Bruno Selau Gonçalves
Lucia Giraffa

CRIANDO UMA OFICINA DE SCRATCH

EDITORA & LIVRARIA
VECHER EDUCOM

© Todos os direitos reservados aos autores. 2021.

Relatório técnico. Apoio do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), através da Bolsa de Produtividade em Pesquisa - PQ - Processo: 312864/2020-5.

ISBN: 978-65-84591-00-4

DOI: 10.47585/9786584591004

Editoração: Marcelo Rodríguez

Vecher
Avenida Paulista, 171, 4º andar
CEP 01.311-904
São Paulo, SP
www.editora.vecher.com.br

Dados Internacionais de Catalogação na Publicação (CIP) de acordo com ISBD

G635c Gonçalves, Bruno Selau

Criando uma oficina de Scratch [recurso eletrônico] / Bruno Selau Gonçalves, Lucia Giraffa. - São Paulo : Vecher, 2021.
58 p. : il. : PDF ; 2,8 MB.

ISBN: 978-65-84591-00-4 (Ebook)

1. Educação. 2. Oficina de Scratch. I. Giraffa, Lucia. II. Título.

CDD 370
2021-4307 CDU 37

Elaborado por Vagner Rodolfo da Silva - CRB-8/9410

Índice para catálogo sistemático:
1. Educação 370
2. Educação 37

APRESENTAÇÃO

Apresentamos neste ebook o conjunto de conhecimentos adquiridos ao decorrer da bolsa de iniciação científica através de estudo e pesquisa, aqui se encontram os materiais e conhecimentos que foram desenvolvidas neste período, estando organizados da forma didática que pretende fixar conceitos básicos de pensamento computacional para aquele que usufruir deste manual.

A Base Nacional Comum Curricular (BNCC)[1] organiza a formação dos estudantes em um paradigma alicerçado em competências, habilidades e atitudes a serem construídas nos estudantes, por ocasião do processo formal de ensino.

A BNCC menciona o tripé (mundo digital, cultura digital e educação digital) os elementos integrantes deste pensar contemporâneo e aponta para questões tais como: metodologias ativas, empreendedorismo, inovação, criatividade, pensamento computacional e outros.

O Pensamento Computacional (PC) pode ser entendido como a compreensão de um problema e de suas características fornecendo ao autor/a da ferramental para montar uma estratégia de solução para este problema.

A Base apresenta o desenvolvimento do Pensamento Computacional como uma disciplina transversal, entendendo-a como mais um meio de resolução de problemas, de formulação de questões em contextos diversificados. Destaca a importância que os estudantes "precisam ser capazes de traduzir uma situação dada em outras linguagens" (BRASIL, 2005, p. 271).

Ao planejarmos a adoção de práticas pedagógicas e as realidades de cada escola dispersas neste pais plural, é necessário considerar a questão do Pensamento Computacional, o qual deverá estar presente, de forma integrada nas diversas disciplinas que compõem o currículo escolar da Educação Básica, a partir do ano de 2020.

Assim sendo, o grupo de pesquisa **Argos**[2] vem desenvolvendo pesquisas relacionadas com Educação Digital e, uma de suas vertes de investigação, está associado a programação para iniciantes, desta forma, o PC se integra como elemento de pesquisa de pesquisa em atividades plugada e desplugadas.

Neste relatório, apresentamos o resultado de um trabalho desenvolvido ao longo de 3 anos com oferta de oficinas para programadores iniciantes, usando como mote a criação de jogos. Por se tratar de um ambiente de programação usando blocos, onde o nível de abstração e detalhamento fica encapsulado nos blocos, e acreditamos que esta estragia se mostra muito interessante para trabalhar os pilares do PC como mencionado por Brackman (2017) em sua tese.

Então, devemos ressaltar que o que lhes apresentamos aqui não surgiu apenas de uma pesquisa de maneira isolada, mas também do convívio em grupo desta pesquisa com outras de mesma linha de pensamento. Portanto, gostariamos de agradecer aos estudantes do curso de bacharelado em Ciência da Computação, Marco Cabral e Oswaldo Junior, por generosamente compartilharem seus conhecimentos e experiências em outros projetos vinculados ao grupo **Argos**. Nos orgulhamos de ter criado um

1 Disponível em: <http://portal.mec.gov.br/index.php?option=com_docman&view=download&alias=79601-anexo-texto-bncc-reexportado-pdf-2&category_slug=dezembro-2017-pdf&Itemid=30192>.

2 Disponível em: <http://dgp.cnpq.br/dgp/espelhogrupo/1961885168367047>.

ambiente agradável de busca de conhecimento, aberto a discussões construtivas, conselhos, amizades e pontos de vista diferentes, que fizeram com que este fosse um projeto aberto a construção coletiva. E, para sintetizar os resultados, resolvemos compartilhar o conhecimento construído de forma gratuita, devolvendo à sociedade, na forma deste e-book, os investimentos recebidos.

Para que chegássemos aqui também tivemos outros parceiros que nos ajudaram a montar oficinas para testar nosso material. Parceiros como John Barcos, Niccolas Maganeli, Henrique Kops, entre outros, porém para que este projeto tivesse início usamos um material que me apresentou o ambiente que aqui usamos, que foi escrito por Vinicius Cerutti em um projeto anterior a este, que nos permitiu iniciar a construção do manual e dicas que aqui disponibilizamos.

Porém, não poderíamos deixar de citar a PUC/RS pelo apoio e uso dos espaços para ofertarmos as oficinas, a bolsa de IC do Bruno e ao CNPq pelas bolsa de produtividade e pesquisa da autora, também orientadora dos projetos, a qual permitiu a publicação desta pesquisa.

Obrigada por fazer download deste material e esperamos que seja bastante útil.

"Quem caminha sozinho chega mais rápido,
quem caminha acompanhado vai mais longe"
(Provérbio Africano)

SUMÁRIO INTERATIVO

1. INTRODUÇÃO AO SCRATCH

Scratch é projetado, desenvolvido e gerenciado pela Scratch Foundation, uma organização sem fins lucrativos. É fornecido gratuitamente e está disponível em <https://scratch.mit.edu/>. O lema do projeto é ofertar ambiente para programação criativa para todXs.

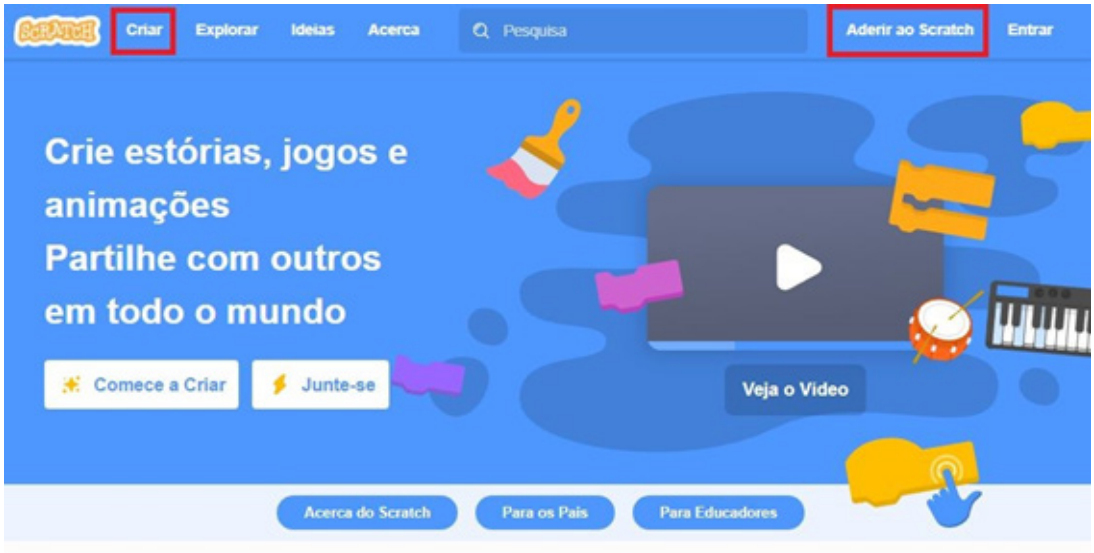
O objetivo deste ambiente é proporcionar uma experiência, em alto nível, relacionada às competências e habilidades inerentes à programação. Não existem restrições de idade ou pré-requisitos específicos, milhões de pessoas criam projetos Scratch numa grande variedade de contextos, incluindo lares, escolas, museus, bibliotecas e centros comunitários.

No Scratch você pode programar jogos, animações e histórias com poucos conhecimentos computacionais e o desenvolvimento dos seus projetos. Porém, se você deseja ser um programador ou programadora, o início da caminhada, usando o Scratch, lhe ajudará a fixar conceitos importantes de computação de uma forma muito facilitada.

Para saber mais da concepção teórica por trás do Scratch acesse o site da fundação em <https://www.scratchfoundation.org/>. Outra dica que fornecemos é ler o livro *Jardim de Infância para a Vida Toda: Por uma Aprendizagem Criativa, Mão na Massa e Relevante para Todos*, de autoria de Mitchel Resnick e editado pela Penso Editora em maio de 2020.

O primeiro passo para programar no Scratch é se familiarizar com o ambiente, e para iniciar o desenvolvimento do programa é necessário estar no que chamamos “modo de edição”, onde aparecem os blocos para criarmos nossas aplicações. Para chegar nesta tela, entre no site do Scratch.

Figura 1: Iniciando Scratch.



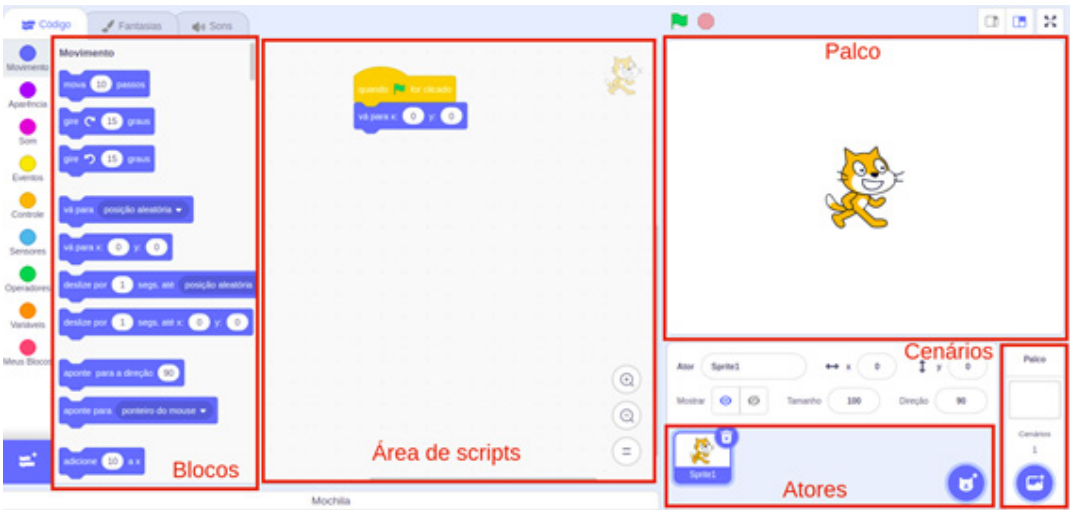
Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Dentro do site, você pode clicar em “Criar” para entrar no modo de edição e programar o seu projeto, mas antes disso recomendamos a criação de uma conta. Você pode criar uma conta clicando em “Aderir ao Scratch”. Com uma conta, você pode salvar os seus projetos diretamente na conta, não é necessário começar do zero ou baixar e carregar o projeto a cada edição.

2. MODO DE EDIÇÃO

Na Figura 2, podemos ver o modo de edição, que é a tela onde o projeto será desenvolvido. A área de blocos é fundamental para desenvolvermos nosso código, pois como o Scratch é um ambiente de programação por blocos, usaremos estes blocos para programar. A área de script é o local onde o código, que também chamamos de script, será desenvolvido, para criar programas arrastaremos os blocos, da área blocos, até a área de script, onde encaixaremos um bloco em outros blocos.

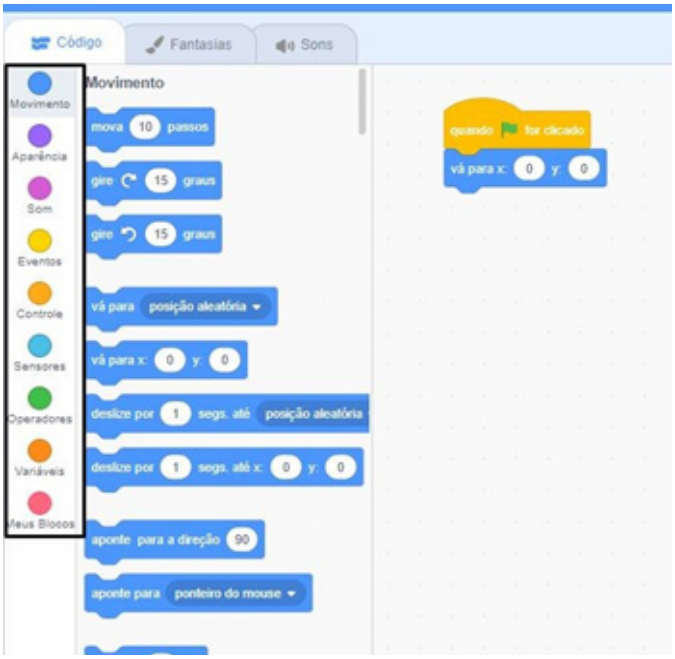
Figura 2.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

O palco apresentado na Figura 2 é o local onde visualizaremos o nosso código funcionando. É onde veremos o resultado do uso de todos os blocos na área de script. A área de atores e cenários é onde poderemos editar a parte gráfica de nosso projeto, focando mais nas artes dos personagens e do cenário.

Figura 3.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Um último ponto que ainda não mencionamos são os grupos de blocos, que estão apresentados na Figura 3. Existem 10 grupos que classificam os blocos do Scratch para melhor organização do ambiente, que são: Movimento, Aparência, Som, Caneta, Variáveis, Eventos, Controle, Sensores, Operadores e Mais Blocos.

2.1. Inicialização

Dentro do modo de edição, devemos inserir o bloco que inicializará qualquer projeto. Este bloco você encontrará dentro do grupo de blocos “Eventos”. É o primeiro bloco deste grupo que está identificado como “Quando ... for clicado”.

Figura 3.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Após identificar o bloco, pressione o botão esquerdo do mouse sobre o bloco e arraste-o para direita, para uma posição parecida com a que este mesmo bloco está posicionado na Figura 2.

2.2. Primeiro Projeto

Faremos com que o gato do Scratch diga a mensagem “Hello” na tela por 1 segundo. Para isso, você deve fazer a inicialização do projeto como foi ensinada no tópico 2.1.

Vá para o grupo de blocos “aparência” e arraste o primeiro bloco deste grupo - (“diga ... por ... segundos”) - de forma com que se encaixe embaixo do bloco de inicialização. Este bloco faz com que a mensagem que você deseja seja exibida na tela por um determinado

período. Você pode determinar a mensagem e o tempo. Para modificar, clique com o botão esquerdo sobre o campo que você deseja alterar.

Figura 4.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

2.3. Execução do primeiro projeto

Mude a mensagem e o tempo para que os campos estejam da forma você deseja. Neste exemplo, usaremos a mensagem “Olá Mundo!” por 1 segundo. Para executar (fazer funcionar sua programação), deve-se clicar na bandeira de execução, então ocorrera no palco a execução do código (sequência de blocos) de nosso projeto, como demonstrado na Figura 5.

Figura 5.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

3. SALVANDO PROJETO

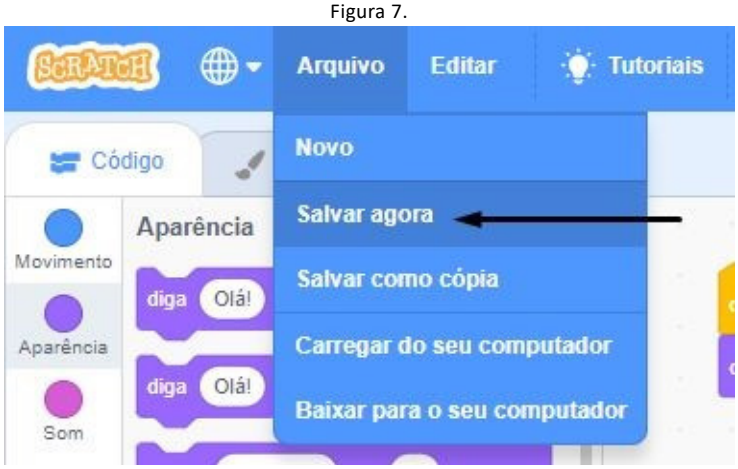
O Scratch permite salvar o projeto para que você possa sair da conta e editar sem recomeçar do zero, para salvar seu projeto você precisará:

- Estar conectado com sua conta no site;
- Dar um nome ao seu projeto (Figura 6).
- Clicar em arquivo (Figura 7);
- Clique em salvar agora (Figura 7);

Figura 6.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Figura 7.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

A Figura 8 mostra como acessar o seu projeto após salva-lo, você deve realizar os seguintes passos:

- Clicando no seu perfil (Canto superior direita);
- Selecionar Minhas coisas;

Figura 8.

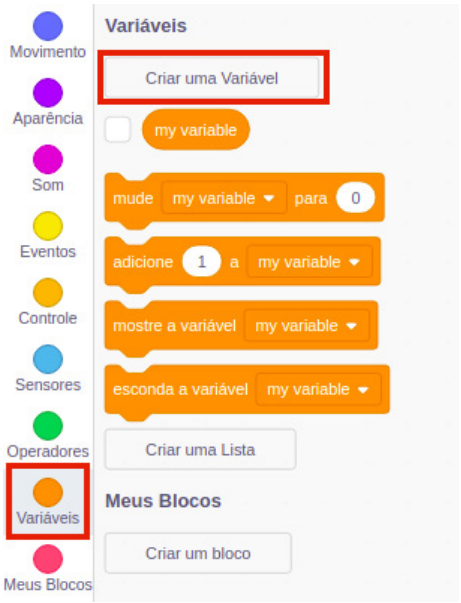
Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

4. VARIÁVEIS

Computador processa dados ...importante lembrar. Toda informação, na forma de dados, é armazenada na memória. Esta é organizada de maneira que cada dado fica num local onde existe um identificador (nome).

Quando se está programando em qualquer linguagem, é preciso saber armazenar os dados na memória. Para isso ser feito, usamos o conceito de variável em Scratch. Para usar as variáveis, você precisa acessar o menu das variáveis, clicar em “Criar uma variável” e nomeá-la. Você pode escolher se ela será manipulada em todos os atores ou apenas onde foi criada. Vale ressaltar que já vem uma variável padrão chama “my variable”.

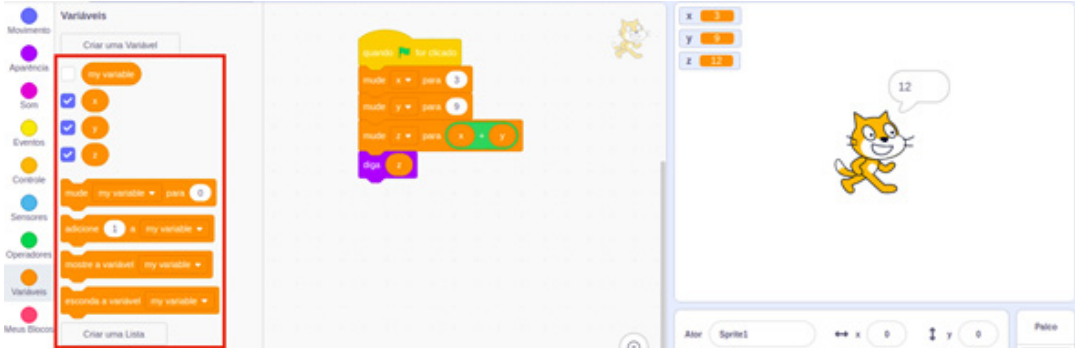
Figura 9.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Após a criação de uma ou mais variáveis, aparecerão blocos que nós usaremos para alterar os valores armazenados. As alterações são feitas nos blocos “mude” e “adicione”, e também temos um bloco de representação para cada variável criada, então podemos usá-las para alterar outra variável, como no caso da variável Z do exemplo apresentado na Figura 10, onde usamos X e Y para alterá-lo com o auxílio de um operador aritmético.

Figura 10.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

4.1. Tipagem

Em algumas linguagens de programação precisamos informar qual o tipo de dado que você está querendo armazenar na variável, se é um número inteiro, um número real, uma letra ou até mesmo uma palavra. Podemos escrever qualquer coisa no campo que se refere a variável, contudo existe uma pequena exceção dentro do Scratch porque sempre que queremos representar um número real devemos usar ao invés da vírgula um ponto.

Figura 11.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

5. OPERADORES

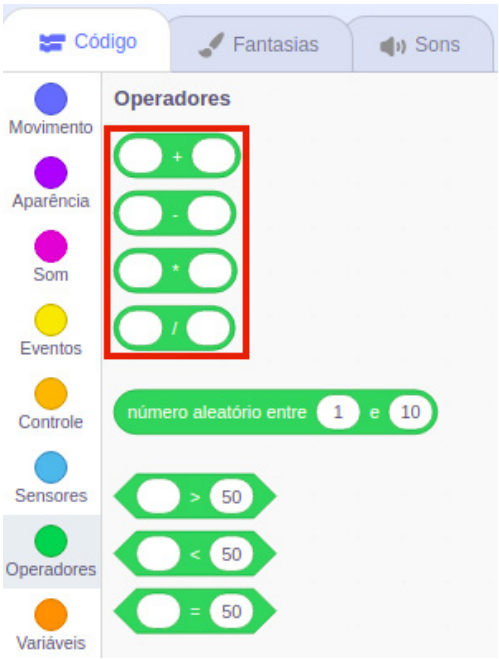
Na programação, os conhecimentos matemáticos ajudam muito na criação de nossos projetos.

No Scratch, temos o conjunto de operadores, que são todos os blocos verdes, esses blocos são correspondentes aos operadores que estudamos na Matemática. Para realizarmos operações, precisamos de algum bloco que represente e faça as ações do operador, como por exemplo somar conteúdos de duas variáveis, adicionar um valor a uma determinada variável e assim por diante, é o Scratch possui os seguintes tipos de operadores: aritméticos, relacionais e lógicos.

5.1. Operadores Aritméticos

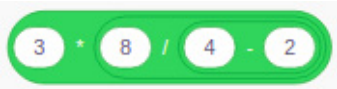
Na Figura 10 são somadas duas variáveis, para realizar operações aritméticas como a do exemplo precisamos acessar os operadores do Scratch. Os operadores são uma categoria de blocos, dentro desta categoria temos vários tipos de operadores, a Figura 12 apresenta os operadores aritméticos.

Figura 12.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Figura 13.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

A partir desses novos blocos nós podemos realizar várias operações matemáticas utilizando os principais operadores, e colocando um bloco dentro de outro da forma que quisermos, como a multiplicação junto de uma subtração no divisor apresentada na Figura 13, é possível conectar

vários operadores aritméticos ligados, ou até calcular médias - Figura 14.

Figura 14.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Apesar de a aritmética tradicional ser resumida a essas quatro operações, o Scratch também possui outras operações que as vezes não são tão conhecidas, mas que estão prontas no Scratch para casos de cálculos mais complexos, basta inserir os valores desejados ou as variáveis criadas. Podemos ver alguns exemplos na figura 15 como arredondamento, resto, módulo, raiz quadrada, logaritmo, entre outros.

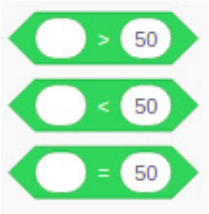
Figura 15.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

5.2. Operadores Relacionais

Os operadores relacionais são fundamentais para qualquer programação pois eles fazem as verificações nas nossas estruturas condicionais (será explicado dentro dos próximos tópicos). Eles nos ajudam a comparar as valores ou variáveis e informar se um elemento é maior, igual ou menor que outro.

Figura 16.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

5.3. Operadores Lógicos

Estes operadores são os mais conhecidos pois estão presentes no nosso cotidiano através de nossos diálogos e inferências, apesar de não percebermos. Quantas vezes não escutamos da família: “Se fizer todas as tarefas então pode brincar” ou dizemos aos amigos “Se eu estiver liberado do trabalho e tiver dinheiro para gasolina então eu vou para praia no fim de semana”.

Se observarmos atentamente as expressões acima:

“Se fizer todas as **tarefas** então pode **brincar**”
“Se eu estiver **liberado do trabalho** e tiver **dinheiro** para gasolina então eu vou **para praia no fim de semana**”.

Podemos reescrever assim:

- Tarefa vamos substituir por X;
- brincar vamos substituir por Y

Podemos escrever desta forma:

- Se X então Y
  ...bem mais simples, não é?

Agora substituindo:

- **liberado do trabalho** substituir por W;
- **tiver dinheiro para gasolina** substituir por K;
- **vou para praia no fim de semana** substituir por Z.

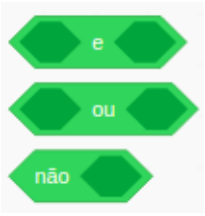
Podemos escrever desta forma:

- Se W e K então Z.

Estes “conectores” que ligam as condições aos seus efeitos são chamados de operadores lógicos. O que queremos é testar e atribuir um valor para sabermos se é V (verdadeiro) ou falso (F), chamados de proposição ..isto algo que estamos propondo e queremos saber se acontece ou não.

Quando a gente avalia logicamente dizemos que o resultado será V ou F. Os operadores lógicos do Scratch são E, OU, NÂO, cada um possui um conjunto de regras para que a gente saiba que resultado vamos receber. A seguir explicamos melhor isto.

Figura 17.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Estes operadores são chamados de binários porque eles só produzem dois resultados: V (verdadeiro) ou F (falso).

5.3.1. Operador AND (E)

Operador lógico AND (E) funciona usando as seguintes regras expressas nesta tabela ... que a gente chama detabela verdade:

X	Y	Conclusão
V	V	V
V	F	F
F	V	F
F	F	F

Figura 18.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Observe que o resultado do seu teste só dará verdadeiro se ambas as proposições (X, Y) que estão sendo avaliadas são verdadeiras. Se X ou Y possuem o valor falso, a conclusão é falsa.

Digamos que nosso X é “Eu tenho um carro ”, o Y é “Tenho gasolina” e a nossa Resposta(conclusão) é “Irei de carro”. Não adianta termos o carro e não termos a gasolina, e também não adianta termos a gasolina e não possuir um carro, portanto só é possível ir de carro se possuir os dois por isso precisamos que as duas condições sejam verdadeiras.

5.3.2. OR (OU)

Operador OU funciona diferente, quando pelo menos uma das proposições forem V ... o resultado é V.

Só dará resultado F quando ambas forem F.

X	Y	Conclusão
V	V	V
V	F	V
F	V	V
F	F	F

Figura 19.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Digamos que nosso X é “Eu tenho uma caneta”, o Y é “Tenho lápis” e a nossa Resposta é “Posso escrever”. Se tenho uma caneta eu posso escrever, se tenho um lápis também é possível escrever, só não é possível se eu não possuir nenhum desses, então se um deles for verdadeiro é verdade pois já podemos escrever.

5.3.2. NOT (NÃO)

Operador lógico NÂO é dito unitário porque ele troca o resultado original da proposição.

x	Resultado
V	F
F	V

Figura 20.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

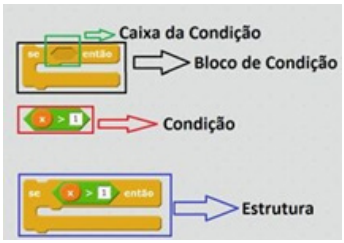
Assumindo uma condição x que possui a seguinte afirmação: “Tenho aula hoje”. Com o efeito do operador NÃO sobre nossa variável ela é modificada para “Não tenho aula hoje”, ou seja, assumimos a valoração contrária.

6. CONDICIONAL

Na programação existem partes do programa onde vamos precisar que em função de determinadas situações ocorrerem vamos desejar fazer ações distintas dependendo de uma condição determinada. Nós chamamos isso de **estruturas condicionais**, que podem nos ajudar a verificar vários casos e continuar o programa dependendo da condição. Para criar estas estruturas usamos os operadores lógicos que mostramos na seção 5.

Para usar este tipo de estrutura em Scratch você irá encontrar os blocos de condicional no conjunto de blocos “Controle”. Na figura 21 temos a condição e o bloco condicional, para termos nossa estrutura basta arrastar a condição até a caixa de condição.

Figura 21.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

6.1. IF (Se)

Esse é o condicional mais básico, ele poderá estar no meio do seu código, porém só será executada a sequência de blocos dentro dele se sua condição for verdadeira, caso contrário o algoritmo continua e ignora o que ocorreria dentro do “IF”.

Figura 22.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

No exemplo da Figura 23 uma variável que informa uma idade, e caso o número seja menor que 18 é retornada uma mensagem dizendo “é menor de idade”. E temos também um outro condicional para o caso de o número ser maior de 17, logo a mensagem retornada é “É menor de idade”.

Figura 23.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

6.2. IF ... Else (Se ... Senão)

Esse condicional possui duas opções de execução de acordo com a condição inserida, se esta for respeitada serão executados os blocos dentro do “se”. Caso a condição não seja respeita será executada a sequência de blocos dentro do “senão”.

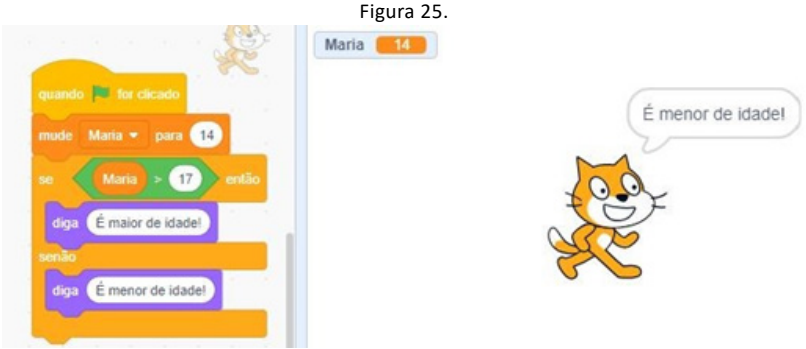
A partir desses novos blocos nós podemos realizar várias operações matemáticas utilizando os principais operadores, e colocando um bloco dentro de outro da forma que quisermos, como a multiplicação junto de uma subtração no divisor apresentada na Figura 13, é possível conectar

Figura 24.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

O exemplo da Figura 25 de forma semelhante à Figura 23, a diferença é que usando o bloco “if else” só precisamos definir um único condicional.

Figura 25.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

7. REPETIÇÃO

A repetição também é outro conceito importante e que na programação chamamos de **iteração**. Cada ciclo de iteração (repetição) é denominado de **laço** ou loop. Este conceito ajuda a realizarmos a mesma sequência de comandos mais de uma vez, ao invés de escrevermos várias vezes algo que faz a mesma coisa. Isso faz com que nosso projeto tenha menos linhas de código e podemos manter um laço executando uma ação até que uma condição se torne verdadeira.

Sim... para podermos criar estrutura iterativas (repetições) vamos usar os conceitos relacionados a testes.

Para usar este tipo de estrutura em Scratch você irá encontrar os blocos de condicional no conjunto de blocos “Controle”.

.

Figura 26.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Nas Figuras 26 e 27 vemos dois jeitos de contar de zero até cinco em Scratch, perceba que a Figura 26, precisamos utilizar dois blocos para cada vez que um novo número é somado. Na figura 27 utilizamos o laço de repetição evitando utilizar várias vezes os mesmos blocos, com isso a Figura 27, que possui um laço de repetição, tem um código com menos blocos.

Figura 27.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

7.1. Repetição por vezes

Neste tipo de laço será repetida a sequência de blocos pela quantidade de vezes que você determinar no próprio bloco de repetição, e após essa quantidade de repetições ser executado o algoritmo segue para o próximo bloco após a estrutura de repetição. Na figura 27 temos um exemplo deste tipo de estrutura.

Figura 28.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

7.2. Repetição por Condição

Neste tipo de laço será repetida a sequência de blocos até que a condição definida no bloco seja verdadeira, enquanto essa não for verdadeira a repetição será executada novamente. Após essa condição se tornar verdadeira o algoritmo segue para o próximo bloco após a estrutura de repetição.

Figura 29.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Na figura 30 vemos um exemplo da repetição por condição, onde fazemos com que a repetição só pare quando o “x” tiver valor 10, caso contrário adicionamos 1 ao seu valor atual.

Figura 30.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

7.3. Sempre

Neste tipo de laço será repetida a sequência de blocos enquanto o projeto estiver em execução, e como você pode ver na figura 31 não é possível encaixar nenhum bloco ao final dele somente dentro.

Figura 31.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Na figura 32 temos um exemplo do “sempre”, que faz com que sempre que o projeto esteja sendo executado o Sprite troque de fantasia .

Figura 32.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

8. CONTROLADORES

Estes blocos nos ajudam a controlar o tempo do nosso projeto, com eles podemos esperar algo acontecer ou esperar por um determinado tempo.

Também existe um controlador para encerrar o projeto ou algum determinado script.

Figura 33.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Na Figura 34 temos um exemplo onde é necessário esperar por um determinado tempo, este breve código adiciona 1 na variável x, e sempre após essa unidade ser somada será informado o valor da variável na tela. O bloco “espere 1 seg” faz com que seja possível ler o valor informado, caso contrário a mudança de valores na tela aconteceria de forma muito rápida. Neste exemplo é apresentado o bloco que atrasa a execução durante um tempo, mas como vimos na figura acima, também existe um bloco que pausa a execução até que uma condição seja respeitada.

Figura 34.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Na Figura 35 é possível visualizar a utilização do bloco “pare”, neste exemplo é adicionado uma unidade na variável x a cada iteração. Utilizamos o bloco “se” para verificar se o valor de x é igual a 50, caso esta condição seja verdadeira a execução do código é encerrada. O bloco “pare” permite encerrar todos os scripts ou scripts específicos, neste exemplo é apresentado o uso do bloco “pare” para todos os scripts.

Figura 35.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

9. MOVIMENTOS

A principal vantagem do Scratch é proporcionar um ambiente simples para criação de aplicações, como por exemplo jogos e animações, tendo isto como objetivo precisamos dar sensação de movimento para nossos projetos, e podemos fazer isso com o grupo de blocos denominado “Movimentos”, que é apresentado na Figura 36.

Figura 36.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

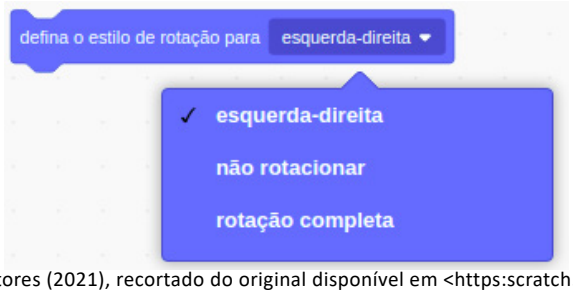
Neste tópico não será ensinado a maneira correta de estruturar uma movimentação, a estruturação de uma movimentação será mostrada no tópico 12, aqui será feita apenas a apresentação dos blocos para sabermos as suas funcionalidades.

9.1. Estilos de Rotação

O estilo de rotação é o que determinará como o seu personagem irá se movimentar, e como isto acontecerá e isto irá alterara forma que seu projeto será estruturado.

Para escolher o tipo de rotação, utilizamos o bloco que está sendo mostrado na Figura 37, que chamamos de “mude o estilo de rotação”.

Figura 37.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Estilos de rotação:

- O estilo “rotação completa”, faz com que o personagem gire, alterando sua direção em 360 graus.

- O estilo “esquerda-direita” faz com que o personagem se direciona apenas para os lados.

- O estilo “não rotacionar” não muda a rotação do personagem, mas muda a direção que ele andara fazendo com que ele ande direcionado do mesmo jeito sempre.

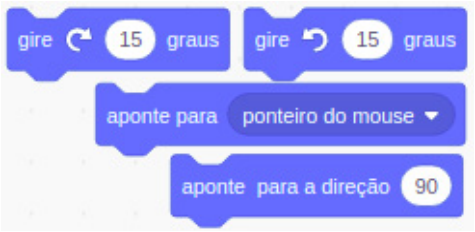
9.2. Direção

Após escolher o tipo de rotação que você usará para o personagem no seu projeto, você poderá controlar a rotação dele a partir de blocos feitos para isso.

Os dois primeiros blocos da figura 38 são modificadores de rotação, ou seja, vão girando conforme o desejado, que neste caso é uma rotação de 15 graus para qualquer lado;

O terceiro bloco da Figura 38 faz com que seu personagem se direcione para o ponteiro do mouse ou até mesmo para outro personagem do projeto. O quarto bloco também modifica a direção, porém para um grau específico, e podem ser outros graus além dos quatro padrões que vem no bloco. Caso queira outro grau você apenas precisa digitar o de seu desejo.

Figura 38.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

9.2.1. Bloco de Mover por Passos

Este bloco fará com que seu personagem de passos, porém ele funciona conforme a sua direção. Colocando uma quantidade de passos positivo ele avançará conforme sua direção, porém se for colocada uma quantidade de passos negativa ele avançará na direção contraria da sua direção.

Figura 39.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

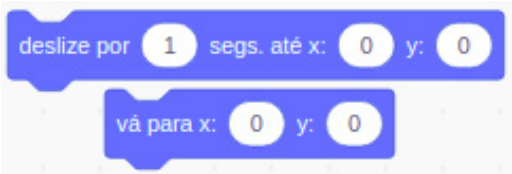
9.3. Coordenadas

O Scratch é um ambiente com um plano de duas dimensões por isso podemos tratar o jogo a partir de um plano cartesiano comum, controlando os personagens por x e y.

O primeiro bloco da Figura 40 fará com que o personagem do projeto deslize até da coordenada atual até a informada conforme o tempo que você deseja;

O segundo bloco da figura 40 faz com que no momento que você chama o bloco no algoritmo ele instantaneamente altera o personagem para coordenada informada;

Figura 40.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

A Figura 41 apresenta os blocos para pegarmos as coordenadas atuais, e inserirmos esta informação em outros blocos;

Figura 41.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Assim como existe os modificadores de rotação para controlarmos a rotação durante o algoritmo, a Figura 42 apresenta modificadores de coordenada.

Figura 42.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

No primeiro bloco da figura 42 você pode alterar instantaneamente a posição apenas em relação ao eixo x;

No segundo bloco você pode alterar instantaneamente a posição apenas em relação ao eixo y;

Com o terceiro bloco da figura 42 podemos simular passos, pois se adicionarmos um número positivo ele irá para a esquerda e se for um número negativo ele irá para a direita.

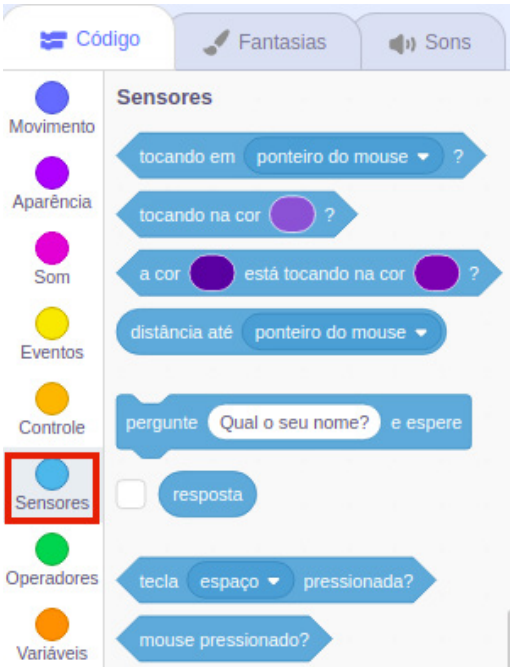
Com o último a ideia é a mesma do anterior, porém suas modificações no plano serão para cima e para baixo.

10. SENSORES

Sensores são blocos que permitem que nosso código verifique se alguma situação ocorreu, como verificar se dois personagens estão encostados ou se alguma tecla está sendo pressionada, mas também permite extrair informações específicas de nosso projeto como saber as coordenadas do mouse ou pedir alguma informação ao usuário do nosso projeto.

Os sensores nos auxiliam na tomada de decisões verificando o estado um elemento do projeto, no Scratch temos um grupo de blocos organizados com todos os sensores. Não apresentaremos todos os blocos deste grupo, apenas os principais que farão já darão noção para termos conhecimento dos demais.

Figura 43.



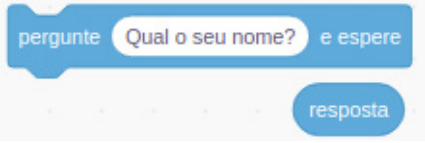
Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

10.1. Scanner

Na programação existe um termo que chamamos de Scanner que nos ajuda a extrair informações de que está utilizando o programa, normalmente no scanner o algoritmo tem uma pergunta já programada e o usuário responde e esta resposta é armazenada no programa.

Em nosso ambiente de programação não é muito diferente, porém já existem blocos prontos que nos ajudam a filtrar dados como números e palavras neste tipo de estrutura iterativa com o usuário.

Figura 44.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

O primeiro bloco da figura 44 possui a pergunta que direcionamos a quem está utilizando nosso projeto, e quando nosso o algoritmo chegar neste bloco a tela de execução fará a pergunta que definimos e só será possível continuar o algoritmo após a pergunta ser respondida.

O segundo bloco da figura 44 vemos o bloco que guarda o que foi respondido no bloco de scanner (bloco que faz a pergunta). O bloco da resposta deve ser imediatamente armazenado em alguma variável, como vemos na figura 45, pois ele não guarda os dados por muito tempo

O exemplo na figura 45 segue a mesma ideia dos apresentados no tópico 6.2, porém a principal diferença é que aqui usamos um scanner para sabermos a idade, pois antes a idade era um dado inserido no algoritmo e não durante a execução.

Figura 45.

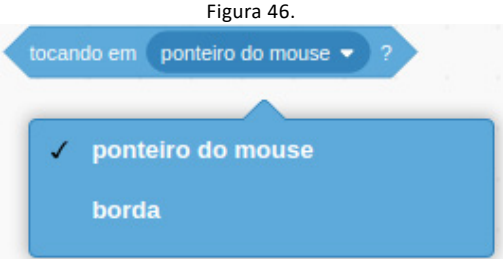
Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

10.2. Colisão

Teremos de verificar na maioria dos projetos se um personagem está tocando em algo ou alguém para podermos fazer mudanças na tela, e os sensores nos permitem fazer isso com um bloco. Normalmente utilizamos este bloco dentro de um bloco de condicional que vimos no tópico 6, mas também podemos usar dentro de alguns outros blocos. Esse bloco pode analisar se o personagem está tocando em objetos, como:

- No ponteiro do mouse;
- Na borda da tela;
- Em outro personagem.

Figura 46.

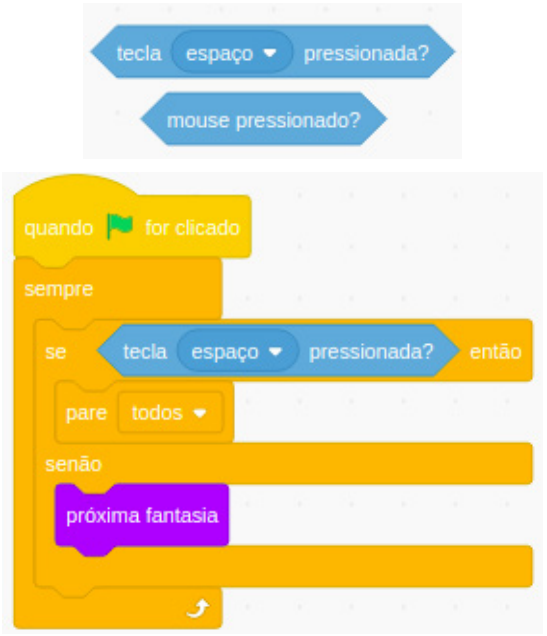


Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

10.3. Comandos

Estes sensores fazem a verificação do mouse ou das teclas, podemos determinar uma ação caso uma tecla seja pressionada ou o botão mouse, para isso usamos normalmente um bloco condicional como vemos na figura 47. O domínio desses sensores é essencial para fazer movimentação de personagens em jogos, pois as ações dos personagens ocorrem apenas se as teclas forem pressionadas.

Figura 47.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

10.4. Atributos (Informações)

Como foi dito na introdução de sensores, nós podemos obter informações dos personagens e até mesmo do palco através de sensores. Usamos o mesmo bloco apesar de que as informações para personagens e palco sejam diferentes.

Os atributos que temos dos personagens são os seguintes:

- Posição x;
- Posição y;
- Direção;
- Número da fantasia;
- Nome da fantasia;
- Tamanho;
- Volume;

Os atributos que temos para o palco são os seguintes:

- Número do pano de fundo;
- Nome do pano de fundo;
- Volume;

Figura 47.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

11. ATORES, FANTASIAS E PLANO DE FUNDO

Usamos os conceitos de fantasias e plano de fundo para personalizarmos o visual do projeto da maneira que acharmos melhor, o Scratch já possui muitas figuras pré-selecionadas que estão organizadas no site por categoria. A nossa

Programação (sequência de blocos) é feita para um ator que é representado na tela por figuras, e chamamos essas figuras de fantasias, é atrás dessas fantasias há um plano de fundo.

Porém também podemos importar figuras de nosso acervo pessoal para deixar o projeto com a nossa cara, com isso podemos baixar figuras da internet em nosso computador e utilizá-las no projeto. Além disso o site permite que usar figuras da câmera de nosso computador para personalização e também podemos desenhar no plano de fundo.

11.1. Atores (Personagem)

Chamamos de Atores no Scratch aquele objeto que será representado por figuras (fantasias) para executar nossa programação, como o gato que usamos na maioria dos exemplos anteriores, que é o ator padrão do nosso ambiente de programação. Neste tópico falaremos das formas de atores que podemos usar no Scratch, mostrando as maneiras diferentes da representação através de figuras.

Figura 48.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Nem sempre utilizaremos o gato que é um ator padrão, por isso devemos aprender a excluir um personagem do nosso projeto. Para apagar o gato devemos abrir o menu do ator, para fazer isso você deve clicar com o botão direito do mouse em cima do personagem que deseja excluir e clicar na opção “apagar” como vemos na figura 48.

Figura 49.

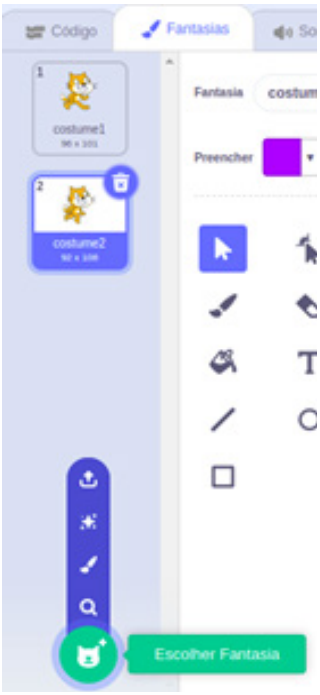
Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

As representações das quatro maneiras de adicionar um ator aparecem na figura 49, assim como as quatro maneiras de adicionar um plano de fundo aparecem na figura 50, as quais serão apresentadas nos tópicos a seguir. É importante ressaltar que mesmo após ter escolhido uma das maneiras que serão apresentadas nos tópicos seguir você pode adicionar outra fantasia no editor de qualquer modo que o Scratch permite, como vemos na figura 51.

Figura 50.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Figura 51.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

11.1.1. Atores e Fundos Padrões do Scratch

O ambiente disponibiliza atualmente cerca de 350 atores e 120 telas fundos para usarmos sem precisarmos procurar por uma figura fora do programa. Caso você não saiba editar figuras e não queira procurar alguma na internet você ainda tem a possibilidade de projetar algo de qualidade com a diversidade de atores já fornecida pelo Scratch.

Figura 52.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Para abrir os atores do ambiente você apenas precisa clicar no ícone destacado na figura 52 (“Selecione um ator”), caso queira ver os planos de fundos também siga a figura 52 (“Selecione um cenário”) e aparecerão todos os fundos do ambiente de forma parecida na figura 53 onde visualizamos os atores, se você já tem algo em mente pode procurar de acordo com as classificações que ficam na parte superior da tela na Figura 53.

Figura 53.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

11.1.2. Criar Atores ou Planos de Fundos no Scratch

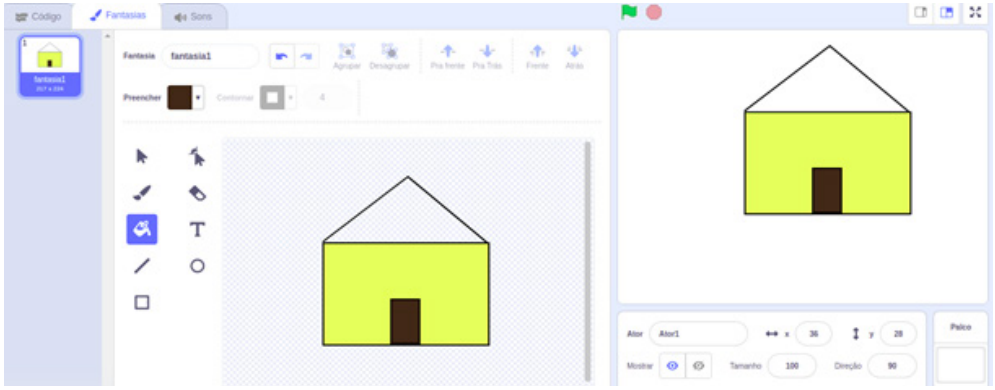
Este jeito de criar um ator nos permite desenhar o personagem da forma que quisermos, basta clicar no pincel da figura 54, que abrirá o editor do próprio ambiente Figura 55, para criarmos um personagem sem fundo branco (formato PNG). Porém também é possível criar um plano de fundo a partir do desenho, basta clicar na representação em destaque na figura 54 e o editor será aberto para criarmos nossa figura de fundo (formato JPEG).

Figura 54.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Apesar do ambiente ser muito bom para a programação ele não é um editor de figuras de ponta, então ele pode ser usado para criarmos nossas figuras, porém para termos um ator ou um plano de fundo com uma figura feita com boa qualidade de edição é aconselhável usar algum outro editor de figuras e salvar uma figura de fundo transparente ou um plano de fundo.

Figura 55.



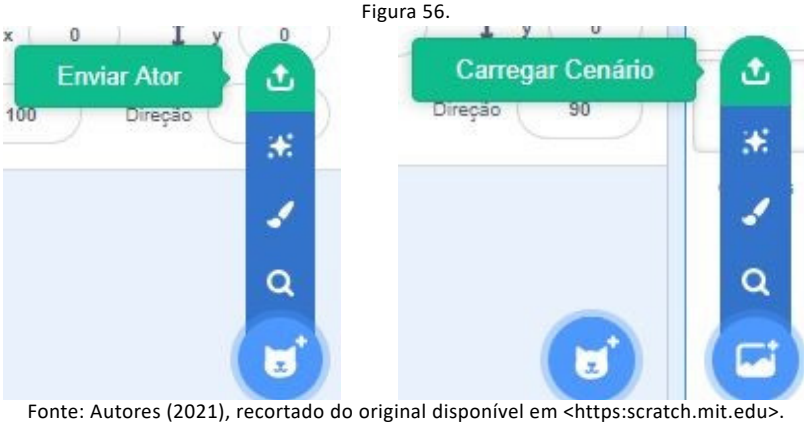
Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

11.1.3. Usar Figuras Para Atores ou Planos de Fundo no Scratch

Caso nenhum dos atores ou planos de fundo já existentes no ambiente seja de seu agrado você pode procurar figuras na internet ou pode usar figuras suas já editadas para montar o seu projeto, é importante ressaltar que caso você procure na internet alguma figura para representar seu ator você deve procurar no formato PNG (com fundo transparente), mas se você quer um plano de fundo o formato deve ser JPEG, isto também serve para as figuras que você criar em um editor de figuras.

Você deve saber em qual pasta de seu computador a figura está, porque quando clicar no ícone em destaque na figura 56 seu localizador de arquivos abrirá para você localizar o ator desejado em seu computador, ou seu plano de fundo. Este é o estilo mais usado para criar atores em Scratch pois além de você usar figuras que você editou, você também pode recriar jogos já existentes usando figuras da internet.

Figura 56.

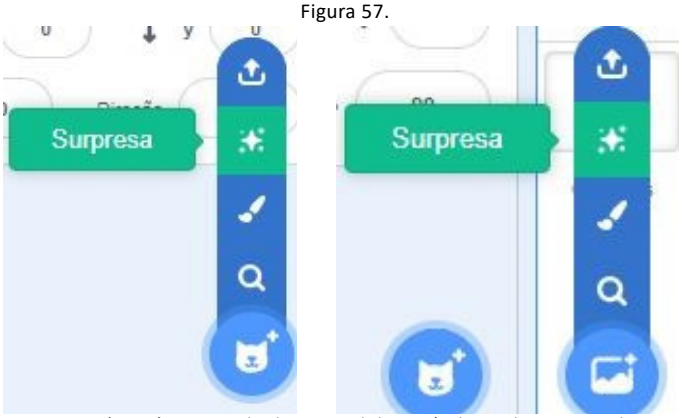


Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

11.1.4. Usar Atores ou Planos de Fundo Aleatórios no Scratch

Este é um modo usado para você criar um projeto rápido no Scratch, o modo surpresa é muito usado quando a lógica importa mais do que as figuras e a interface, assim o próprio ambiente já escolherá seus personagens e atores. Na figura 57 vemos como inserir um ator aleatório ou um fundo aleatório.

Figura 57.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

12. APARÊNCIA

O conceito de aparência em nosso ambiente determina que alguma alteração será feita em algum dos atores, isso nos permite fazer com que o ator mude seus aspectos iniciais durante a execução do nosso código como seu tamanho, sua cor, sua fantasia e outras características.

Neste tópico veremos alguns dos blocos do grupo denominado “Aparência”, não apareceram aqui os blocos que já foram mostrados anteriormente nos exemplos, veremos os blocos mais importantes que ainda não aparecem em nosso manual.

Os dois primeiros blocos da figura 58 mudam o estado de visibilidade do personagem, o primeiro bloco torna o ator visível caso ele esteja invisível. Já o segundo bloco faz o inverso, torna invisível caso esteja visível.

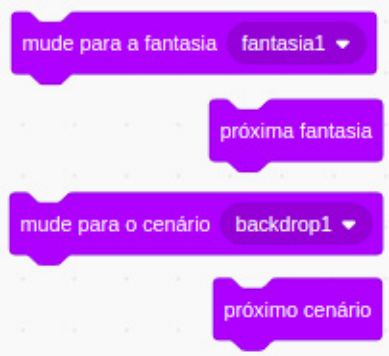
Os dois últimos blocos permitem que possamos mudar o tamanho do personagem.

Figura 58.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

O primeiro bloco da figura 59 altera a fantasia do ator para uma fantasia específica. O segundo altera a fantasia para a próxima da lista de fantasia do ator. O terceiro altera o plano de fundo para um plano específico, e o quarto altera o plano de fundo para o próximo da lista de fantasia de planos de fundo.

Figura 59.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

13. ANIMAÇÕES

Como foi visto no tópico 11 um ator pode possuir mais de uma fantasia, podemos enxergar isso de maneira mais clara na figura 60 que está com o editor aberto e o ator possui duas fantasias. A animação é o que dará impressão de movimento ao nosso personagem, mas também pode servir pra várias outras coisas como trocar a fantasia para um personagem que durante o projeto se abaixou, mudou de roupa, morreu, sofreu impacto, entre outros.

Figura 60.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

A função mais usada para este conceito é fazer com que o boneco simule que está caminhando, para isso você precisará de ao menos duas figuras as quais as pernas do ator estejam em posições diferentes, na figura a seguir usamos as duas fantasias da figura 61.

Figura 61.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

A sequência de blocos que vemos dá a ideia de movimento ao personagem como foi dito anteriormente, porém como usamos as fantasias da figura 60, que são apenas duas, a animação não fica muito real pois está sendo feita apenas troca de duas fantasias, quanto mais fantasias usarmos para fazer a progressão desse movimento nós tornaremos a nossa animação mais real.

Figura 62.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

Para tornar a nossa animação mais real podemos procurar na internet por Sprites daqueles personagens que queremos como vemos na figura 62, onde o personagem possui 7 figuras com as pernas em posições diferentes. Assim podemos usar um editor para cortar a figura em cada frame, que são as aparições do personagem na figura tendo uma postura diferente.

Na figura 63 a ideia é similar ao modo que já vimos anteriormente de fazer uma animação, porém agora podemos usar o bloco “espere...” para delimitar um tempo entre as trocas de fantasias. Dessa forma podemos fazer com que o personagem pareça estar caminhando ou correndo, para parecer que ele esteja andando é necessário deixar o tempo de maior duração dentro do bloco “espere...”, caso queira que ele corra deixe um tempo de menor duração entre as trocas de fantasia.

Figura 63.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

Caso nem todas suas fantasias sejam para fazer aquela animação você não poderá usar os modos ensinados anteriormente, você deverá fazer como na figura 65 que utilizamos apenas as fantasias que queremos para fazer a animação, pois a última das fantasias da figura 64 não serve para uma animação de caminhada e sim para animação de voar.

Figura 64.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

Figura 65.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

13.1. Animação no Plano de Fundo

Muito dos casos não basta apenas animar o personagem, então você deverá animar o plano de fundo para que o ambiente fique mais vivido, a ideia é semelhante a animação do ator pois você deve ter mais de um plano de fundo diferente e ficar trocando estes planos de fundos em uma determinada ordem que dará a sensação de movimento ao plano.

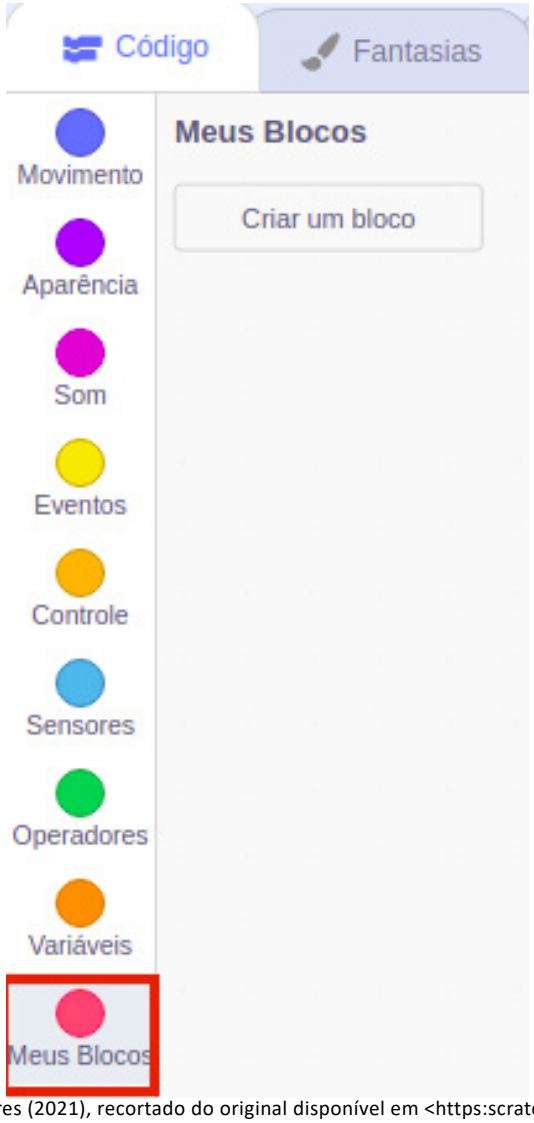
Figura 66.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

14. MAIS BLOCOS

Neste tópico aprenderemos a criar um bloco a partir dos outros já existentes, pois as vezes precisamos de uma funcionalidade que ainda não existe em forma de bloco no Scratch, então precisamos criá-la a partir dos blocos que já nos foram fornecidos, porém usar na sequência de blocos principal do nosso projeto deixaria esta sequência muito longa, por isso criaremos um bloco em um lugar separado e o utilizaremos no meio desta sequência.

Figura 67.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Existe um grupo de blocos no Scratch apenas para criar blocos, esse grupo é chamado de “mais blocos” e é o último grupo. Após selecionar este grupo você deve clicar em “criar bloco” como é mostrado na figura 68 e abrirá uma janela para você nomear o bloco que irá criar, então teremos o nosso novo bloco dentro de “mais blocos” porem devemos descrever sua funcionalidade no bloco “defina ...”. Após definir o nosso novo bloco podemos usá-lo quando quisermos no código e quantas vezes forem necessárias.

Figura 68.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

No exemplo da Figura 69 criamos um bloco chamado scanner para filtrar a idade do usuário do nosso projeto e a partir desse dado dizemos se ele é maior de idade ou não, neste caso parece desnecessário criar um bloco pois teríamos apenas dois blocos a mais na nossa sequência de blocos principal, porém é necessário entender que poderíamos fazer o código maior e usar mais vezes esse bloco criado.

Figura 69.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

O exemplo da figura 70 mostra como economizamos blocos na sequência principal quando criamos um novo bloco, pois todos os que foram usados na definição de

"conferindo" poderiam ser utilizados onde se localiza este bloco. A ideia do projeto abaixo é fazer a brincadeira "Adivinhe o número", onde existe um número que deverá ser adivinhado e o chamamos de "numeroCerto" que neste caso é definido como 7, mas poderia ser qualquer outro de 1 até dez, o usuário do projeto vai "chutar" um número quando o bloco "conferindo" for executado e isso se repetirá até que ele acerte.

Figura 70.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

15. MOVIMENTAÇÕES BÁSICAS

Para que seja feita a construção de um jogo é necessário conseguir controlar alguém ou algo durante a execução de nosso projeto, uma das principais formas de fazer isso é podendo manipular os movimentos de algum ator e neste tópico abordaremos a construção da estrutura que permita que o usuário controle algum personagem do projeto. Mostraremos aqui as principais estruturas de movimentação, porém dependendo do seu desejo você precisará criar sua própria estrutura pois existem muitas formas de movimentar e você deve entender este capítulo da forma certa, ao invés de decorar as estruturas, pois assim você terá o conhecimento para criar a movimentação que precisa.

Para criar estas estruturas é fundamental que você tenha compreendido os capítulos que apresentam condicional, repetição e movimentos pois não é possível construir as ideias deste capítulo sem a noção destes tópicos que são a base das nossas estruturas. Cada movimentação tem um estilo de rotação específico, então uma revisada no tópico 9.1 ajudará a compreender o porquê das escolhas de cada bloco. Como a intenção atual é apresentar as estruturas, não iremos mostrar as estruturas com a animação junto, porém você aprendeu nó tópico 13 e poderá adicionar ao seu projeto para ficar mais completo.

Figura 71.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

A figura 71 mostra um tipo de movimentação bem primitivo que utiliza apenas duas teclas, porém é utilizada em muitos jogos de estilos Pong, Temple Run (para pc), Space Invaders, etc. O estilo de rotação neste caso não importa, pois usamos as coordenadas e não a direção do ator.

- Primeiramente fixamos uma posição para cada vez que o jogo iniciar, com o bloco “vá para ...”;
- Após isso inserimos a repetição através do bloco “sempre”, que é composto por dois condicionais e estará sendo rodado a todo momento de execução;

- O primeiro condicional verifica se a tecla ‘s’ está pressionada para movimentarmos o ator para baixo;
- O segundo condicional verifica se a tecla ‘w’ está pressionada para movimentarmos o ator para cima.

Figura 72.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

A figura 72 mostra um tipo de movimentação comum que é bastante utilizada para simular carros, pois utilizamos o estilo de rotação “em todas direções”(o primeiro mostrado no tópico 9.1) que permite ao ator girar em torno de seu eixo para troca de direção.

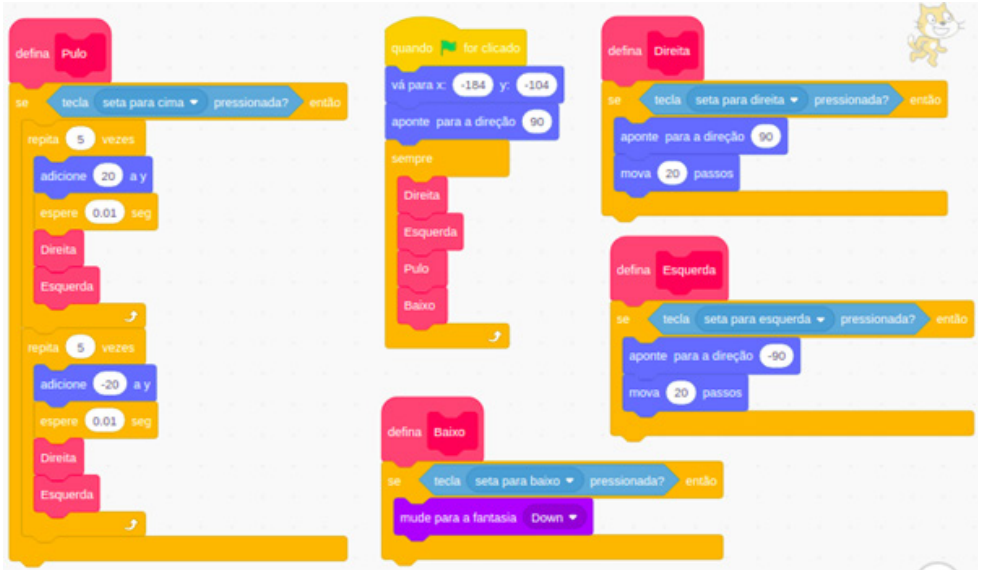
- Primeiramente fixamos uma posição e depois uma direção com os blocos “vá para...” e “aponte para a direção...”, para cada vez que o jogo começar;

- Após isso inserimos a repetição através do bloco “sempre”, que é composto por quatro condicionais e estará sendo rodado a todo momento de execução;

- Nos dois primeiros condicionais verificamos se a seta direcional para cima ou a seta para baixo do teclado foram pressionadas, então faremos que o personagem se mova positivamente(para frente) ou negativamente(para atrás), usamos o bloco “mova” que não utiliza coordenadas, mas a direção do ator;

- Nos dois segundos condicionais verificamos se a seta direcional para direita ou a seta para esquerda do teclado foram pressionadas, então faremos que o personagem gire para a direita ou para a esquerda dependendo da tecla que foi pressionada, usamos os blocos “gire ...”;

Figura 73.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>

A figura 73 mostra um tipo de movimentação mais avançado que utiliza o conceito de mais blocos para que não fique tudo desorganizado em uma única sequência de blocos. Apesar de variar bastante essa movimentação de acordo com sua necessidade, ela é usada para movimentar seu ator principal em jogos arcade como Mario Bros.

- O estilo de rotação neste caso é o “Esquerda-Direita”, pois o ator só deve se direcionar para estes lados;

- Primeiramente fixamos uma posição e uma direção(normalmente apontando para a direita) para cada vez que o jogo iniciar, com os blocos “vá para ...” e “aponte para a direção...” ;

- Após isso inserimos a repetição através do bloco “sempre”, que é composto por blocos que iremos criar a seguir e estará sendo rodado a todo momento de execução;

- Dentro da repetição(“sempre”) temos que criar o primeiro bloco que se chama “direita”. Na definição do bloco vemos que temos um condicional que verifica se a seta para a direita foi pressionada, se essa condição acontecer direcionamos o ator para direita (com o bloco “aponte para ...”), que é 90 graus e movemos ele (com o bloco “mova’);

- Após criar “direita’ devemos criar o bloco “esquerda”, que tem a mesma estrutura que o anterior, porém devemos direcioná-lo para esquerda colocando -90 graus no bloco “aponte para ...”;

- A criação do bloco “baixo” é fácil, você precisa ter lido os tópicos sobre aparência e fantasias, porque apenas trocaremos a fantasia do personagem para alguma que ele esteja agachado, faremos isso com o bloco “mude para a fantasia ...”;

- A parte complicada fica na criação de “pulo”, que é onde faremos o salto do ator, caso a seta para cima do teclado tenha sido pressionada acontecerá este pulo da seguinte forma:

Nós usaremos o laço “repita” para fazer ele subir, então repetiremos 5 vezes o processo de adicionar um valor positivo em y(com o bloco “adicione a ... y”) e depois usaremos o bloco “espere 0.01 seg” para nos ajudar a controlar o tempo. Após isso usamos os blocos recém-criados “direita” e “esquerda”, para que o ator possa se mover no ar e assim chamamos o “repita”;

Como o ator subiu, devemos desce-lo e faremos isso de maneira semelhante a que fizemos com o último laço de “repita”, a única diferença é que no bloco “adicione a ... y” não colocaremos um valor positivo, deverá ser o mesmo valor anterior porém negativo;

16. MENSAGENS

Desde o início do manual até o atual momento nós aprendemos a trabalhar com o Scratch, programação, com a manipulação de um personagem e do plano de fundo, porém um jogo ou uma animação que são ambientados por um único personagem nos dão poucas opções para criatividade, então precisamos aprender a fazer os personagens interagirem entre si através do uso de blocos.

Para fazer esta interação entre atores usaremos um sistema de mensagens que permitirá que um personagem diga ao outro quando executar uma ação, isso nos permitirá identificar colisões, conversas, tiros e outras coisas. Esses blocos podem ser encontrados no grupo de blocos intitulado “Eventos”, que já conhecemos porque lá está o bloco de inicialização de projetos, logo não mostramos aqui os blocos já vistos e nem os que não são triviais para nossa didática.

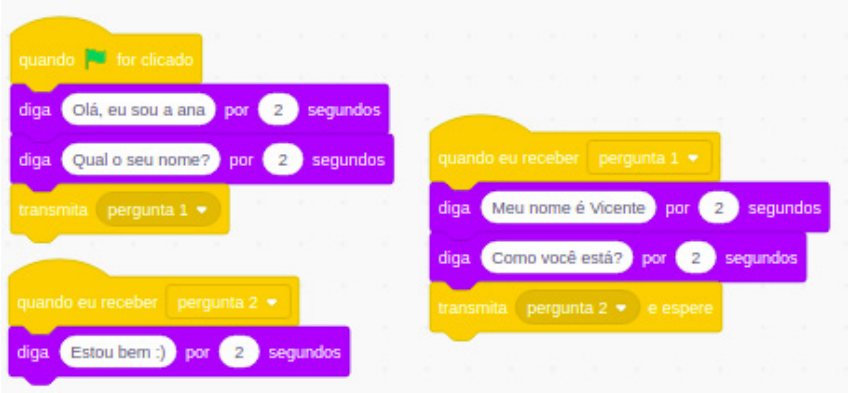
Figura 74.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Na Figura 74 vemos os principais blocos que usaremos para transmitir mensagens, para criar uma mensagem nova basta ir em “nova mensagem” como vemos no segundo bloco (também existe esta opção no primeiro bloco).

O segundo bloco, chamado de “Envie ...”, envia a mensagem que desejamos, basta inserir na parte do algoritmo que desejamos. Quando esta mensagem for enviada, a comunicação será estabelecida pois começará a execução do primeiro bloco, chamado de “Quando receber...”, então a sequência dos blocos inseridos neste serão executados.

Figura 75.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

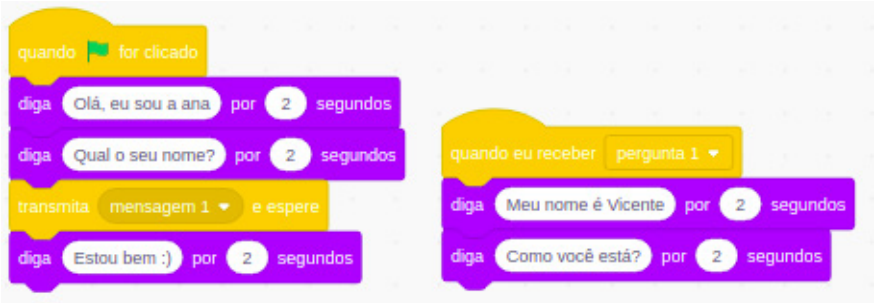
A Figura 75 é um código que implementa o diálogo de dois atores. Após iniciar o projeto, o primeiro ator se apresenta com o bloco “diga...”, logo após ele pergunta o nome do outro personagem.

O bloco seguinte (“envie...”) é o que nos permitirá continuar o projeto com o outro ator.

Quando a mensagem é enviada pelo primeiro ator, o segundo recebe esta através do bloco “Quando receber...”. Então ele responderá seu nome e perguntará ao primeiro ator “como você está?”, logo envia outra mensagem para que o ator um possa responder a pergunta. Portanto voltamos ao primeiro ator que recebe a nova mensagem e responde.

Este é a maneira mais comum de trocar mensagens, mas também podemos trocar mensagens de acordo com a Figura 76. Onde você pode notar que só é enviada uma mensagem, porque usamos um bloco de envio diferente, chamamos de “envie... a todos e espere”, após o segundo ator receber a mensagem ele ainda executa seus blocos, mas não precisa enviar outra porque o primeiro ator volta a executar o algoritmo quando o segundo ator finaliza sua sequência de blocos.

Figura 76.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

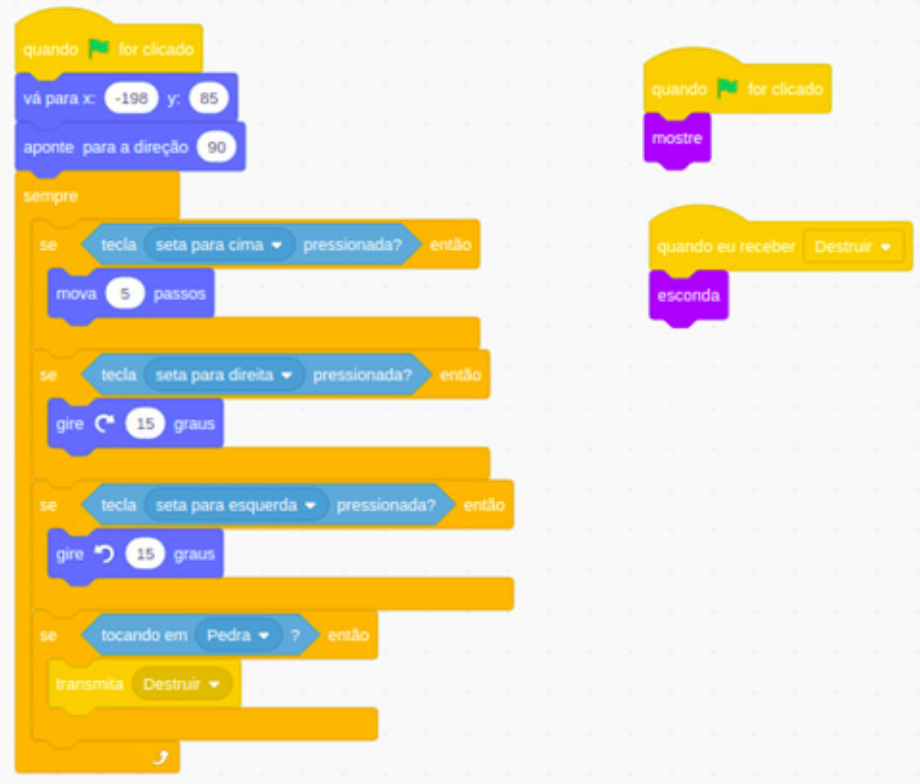
17. COLISÕES

O intuito deste tópico é apresentar colisões, pois existem projetos que você deseja identificar a colisão entre dois atores e realizar uma ação e para fazer isso utilizaremos os conceitos que vimos nos últimos tópicos. Usaremos os conceitos de mensagens para informar o outro personagem que houve uma colisão.

Como vemos na figura 77, você implementa uma movimentação de acordo com a que deseja, mas ao analisar nosso último bloco de condicional - “se ... então” -, utilizamos o sensor “tocando em ...” para identificar se este ator está tocando outro, caso esta condição seja respeitada ele envia uma mensagem para que outro ator reaja a colisão.

Em nosso exemplo a ideia é esconder o ator que foi tocado, então na figura 77 usamos o bloco esconder para realizar o que desejamos. Este é um exemplo de colisão, porém você precisará montar seus blocos de acordo com a sua necessidade, mas aqui lhe mostramos como funciona este conceito que é bem útil para utilizarmos em jogos.

Figura 77.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

18. PROJÉTEIS

A criação de jogos é ocasionalmente feita com uma trama entre herói e vilão, então precisamos aprender a fazer com que haja troca de golpes, poderes, ou algo semelhante, pois é um embate é precisamos que haja um enfrentamento. Neste tópico mostraremos como disparar projéteis de um personagem, tendo o intuito de acertar seu inimigo.

É importante afirmar que precisamos usar no mínimo um total de quatro atores, porque além do "herói" e do "vilão", cada um deverá ter um ator para seu projétil, fogo, raio de luz, etc, que é o objeto a ser arremessado. Mostramos como se faz apenas o projétil de um deles, pois o outro será de maneira semelhante, logo você saberá estruturar. Como foi citado em vários tópicos, você precisará adaptar o que aprenderá neste tópico conforme a necessidade de seu projeto.

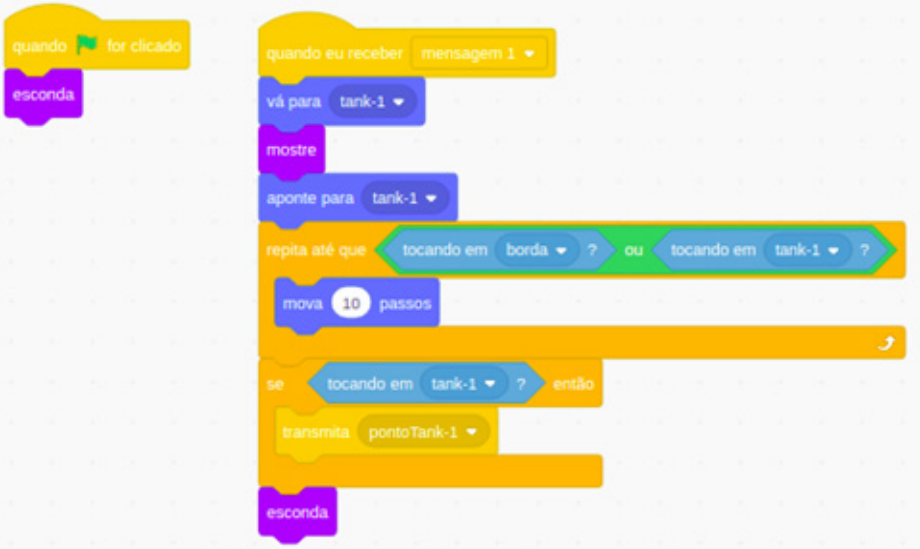
Figura 78.

Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Na figura 78 apresentamos uma movimentação que já mostrados no tópico deste assunto, a diferença desta para as anteriores é o condicional que vemos no final da nossa estrutura de repetição.

Neste condicional verificamos se a tecla espaço foi pressionada, caso isso tenha acontecido enviaremos uma mensagem que será recebida pelo nosso projetil e fara um tiro, como mostra a figura 79, pois estamos trabalhando em teoria com um projeto de dois tanques.

Figura 79.



Fonte: Autores (2021), recortado do original disponível em <https:scratch.mit.edu>.

Nosso projetil começa o projeto escondido como vemos na parte superior esquerda da Figura 79, porém o conceito se aplica quando recebemos a mensagem 1. Quando recebemos a mensagem enviada após a tecla espaço ser pressionada, ocorre o seguinte:

- Movemos o nosso projétil que ainda está escondido para as coordenadas do tank-1(o tank que irá atirar), com o bloco “vá para”;
- Então agora que ele já está junto com o tank, ele pode ficar visível e fazemos isso com o bloco “mostre”;
- Precisamos direcionar o projétil para a mesma direção que o tank-1 está apontando, então pegamos o sensor “direção de tank-1” e inserimos dentro do bloco “aponte para a direção ... graus”;
- Após esse bloco você vê na figura 89 uma estrutura de repetição, que movimenta o projétil até que ele toque na borda ou no tank-2;
- Então criamos um condicional depois da repetição, que escondera o projétil caso o tank-2 for tocado, e enviamos uma nova mensagem que será recebida pelo tank-2.
- Esta nova mensagem já mostramos no tópico sobre colisões, você poderá receber ela para que o tank-2 seja escondido, volte para sua posição inicial do jogo, ou faça qualquer outra coisa que você queira implementar no seu projeto.
- Se o tank-2 não tenha se chocado com o projétil nós também escondemos o projétil, pois mesmo assim ele tocou em alguma borda da tela de execução.
- Então usamos no final o bloco “esconda”.

19. ORGANIZANDO UMA OFICINA PASSO-A-PASSO

Agora que conseguimos fornecer uma boa noção de como o Scratch funciona, vamos colocar em pratica estes conhecimentos.

Disponibilizamos nesta seção um roteiro para você organizar uma oficina com Scratch considerando um público que não tem pratica de programação.

Para incentivar a pratica da programação criamos uma oficina cujo objetivo é criar jogos usando o Scratch. Evidentemente, que os passos aqui organizados vão permitir você criar o que desejar.

Encontro 1:

- Apresentação:
  - o Scratch;
  - o Projetos;

- Ambiente:
  - o Personagens (Ator/Fantasia);
  - o Fundo;
  - o Tipos de blocos;

- Varáveis (Seção 4);

- Operadores (Seção 5);

Neste link você pode encontrar todos os materiais de apoio utilizados para esta aula, contendo apresentações, exercícios e projetos exemplo.

Encontro 2:

- Condicional (Seção 6);

- Sensores (Seção 10);

- Animação (Seção 13);

Neste link você pode encontrar todos os materiais de apoio utilizados para esta aula, contendo apresentações, exercícios e projetos exemplo.

Encontro 3:

- Laço (Seção 7);

- Movimentação (Seção 15);

- Mais Blocos (Seção 14);

Neste link você pode encontrar todos os materiais de apoio utilizados para esta aula, contendo apresentações, exercícios e projetos exemplo.

Encontro 4:

• Interação de personagens (Seção 16)

• Colisão (Seção 17);

• Projeteis (Seção 18);

Neste link você pode encontrar todos os materiais de apoio utilizados para esta aula, contendo apresentações, exercícios e projetos exemplo.

Encontro 5:

Recomendamos que este seja um encontro de apoio para o desenvolvimento dos projetos.

Encontro 6:

Este é o momento de compartilhar, onde serão apresentados os projetos e os desafios durante esta jornada de programação. Alguns exemplos dos projetos compartilhados em nossas oficinas estão listados abaixo.

- Bola de neve;
- Flappy;
- Exemplo 3;
- Exemplo 4;

REFERÊNCIAS

BRACKMANN, Christian Puhlmann. **Desenvolvimento do pensamento computacional através de atividades desplugadas na educação básica**. Porto Alegre: PGIE-IFRGS. Tese de Doutorado. Disponível em: <https://lume.ufrgs.br/handle/10183/172208>. Acesso em 17 de julho de 2021.

Este livro contou
com apoio do
Conselho Nacional de
Desenvolvimento
Científico e Tecnológico (CNPq)
- Bolsa de Produtividade em Pesquisa – PQ
Processo: 312864/2020-5.

CNPq
Conselho Nacional de Desenvolvimento
Científico e Tecnológico