

Aprendendo a Programar

ProCubo

Começando

Seja muito bem vindo ao curso de programação do ProCubo. O objetivo deste material é te apresentar as principais ideias que um programador precisa saber. Focaremos em fundamentos da programação e te ensinaremos a criar programas divertidos. Ao concluir esse curso, você será capaz de programar jogos, criar animações e fazer muitas outras coisas que parecem impossíveis sem as ferramentas que serão apresentadas.

A você, que estreia agora no universo da programação, a nossa mais sincera admiração. A decisão de buscar aprender algo novo exige coragem. Acredite, nesse começo você entrará em contato com ideias aparentemente simples e diretas, mas são exatamente essas ideias que formam a base para os feitos tecnológicos mais complexos que você pode imaginar: computadores, celulares, inteligência artificial e o sistema operacional de um foguete utilizam os mesmos princípios – e são esses princípios que você aprenderá agora. Aperte o cinto e aproveite a viagem.

Para quem escrevemos o material?

Esse material foi escrito para pessoas que desejam começar a programar de maneira fácil e intuitiva, independentemente do seu nível de escolaridade e sua idade. No último módulo, nós damos dicas de outros recursos que te ajudarão a aprofundar seus conhecimentos e continuar sua jornada no mundo da programação.

O que faremos?

Esse material é dividido em 4 módulos. Cada um deles discute um conceito fundamental da programação e te dará exemplos de como e quando usá-lo. Ao final de cada módulo, dedicamos uma seção para que você coloque a mão na massa. Iremos programar juntos, aos poucos, um pequeno jogo utilizando as ideias ensinadas durante o material. Nesse jogo, um gato (controlado pelo jogador) deve coletar maçãs para fazer pontos. Porém, uma bola desgovernada o atrapalha, ameaçando derrubar todas as maçãs caso eles se toquem. Uma imagem do jogo é mostrada na Figura 1.



Figura 1: Na versão final do jogo, um gato coleta maçãs enquanto evita tocar na bola.

Nós dividimos o material dessa maneira para que você aprenda e coloque em prática uma nova ideia toda aula. É importante que você tente programar o jogo por conta própria, mas sugestões de como o jogo pode ser feito são mostradas no final do material. Note que quando programamos existem múltiplas formas de se atingir o mesmo objetivo.

Notação

Ao longo do material usaremos algumas caixas coloridas para te ajudar a entender o conteúdo e indicar onde estão os pontos mais importantes:

Caixas azuis serão usadas para representar **ideias** fundamentais.

Caixas verdes serão usadas para **dicas**.

Usaremos caixas amarelas como sinal de **cuidado**.

1 Introdução: falando com máquinas

É possível falar com máquinas? Para responder essa pergunta, vamos investigar a maneira como humanos se comunicam. Imagine uma conversa entre você e uma pessoa de outro país, cada uma falando uma língua diferente. Será difícil de vocês se entenderem, não é mesmo? Agora imagine que vocês estejam utilizando um aparelho capaz de traduzir tudo o que é falado. Que alívio! Dessa forma vocês poderiam conversar e interagir. Nesse exemplo, perceba que a tradução entre as linguagens é fundamental para comunicação.

O desafio quando tentamos nos comunicar com máquinas é similar ao que você enfrentaria nessa conversa: computadores e humanos não falam a mesma “língua”. Por isso, precisamos traduzir nossas mensagens para um formato que eles sejam capazes de compreender. Assim como é possível nos comunicarmos com outras pessoas falando ou escrevendo em um determinado idioma, podemos nos comunicar com computadores através de **códigos**. Mas como isso funciona? Para entendermos, usaremos o Scratch.

Ideia: Para falar com uma máquina, é necessário traduzir a mensagem para uma linguagem que ela entenda. Para isso, escrevemos códigos.

1.1 O que é o Scratch?

Scratch é uma plataforma online que permite que você crie códigos a partir de blocos. Usando essa ferramenta, controlaremos um personagem (que chamaremos de Gato). Se comunicar com o Gato é parecido com falar com uma pessoa. Por exemplo, para pedir que alguém fale “Olá”, você diria algo parecido com a seguinte frase.

Diga “Olá”.

De maneira similar, podemos comandar que o Gato diga “Olá” utilizando o bloco da Figura 2. Perceba como é parecido com a forma como falamos.

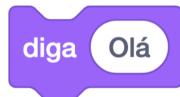


Figura 2: Como dizer “Olá” usando Scratch.

O Scratch também permite que usemos outros comandos. Por exemplo, podemos pedir para o Gato andar. Para isso, nós temos o bloco da Figura 3.



Figura 3: Como mover o Gato usando Scratch.

Ao longo do nosso curso, vamos entrar em contato com muitos blocos. Cada um realiza um comando diferente e contém um texto explicando o que faz. Cada bloco também possui um formato específico que te ajudará a entender como ele funciona e onde melhor se encaixa. Se tudo isso parecer confuso, fique tranquilo. Vamos aprender pouco a pouco como criar códigos usando o Scratch ao longo desse módulo.

Ideia: No Scratch, cada bloco representa uma porção de código diferente, e tem sua própria finalidade.

1.2 Seu primeiro programa em Scratch

A melhor forma de aprender a usar o Scratch é colocando a mão na massa. Ou seria, a mão nas teclas? Para começar, acesse: <https://scratch.mit.edu/>. Depois de ser direcionado ao site, clique em “Comece a Criar”. Caso esteja em dúvida, você deve procurar algo parecido com a Figura 4A. Ao clicar em “Comece a Criar”, você será direcionado à página mostrada na Figura 4B.



Figura 4: (A) Página inicial do Scratch, onde você deve clicar em “Comece a Criar”. (B) Tela principal do Scratch, onde os blocos estão.

À esquerda da sua tela você verá vários círculos coloridos, cada um com um nome. Cada círculo corresponde a uma categoria de comandos (códigos). Ao clicar em algum desses círculos, você encontrará vários blocos de cores correspondentes à categoria selecionada. Cada bloco colorido representa um comando (código). Agora que você sabe onde encontrar os comandos, você poderá começar a utilizá-los para controlar o personagem (Gato) que está na parte direita da tela. Para isso, combinaremos os blocos, empilhando um em cima do outro!

Dica: Para usar um bloco, clique em cima dele e arraste para o meio da tela.

O que você deve estar se perguntando é: quais blocos devemos usar? A resposta depende do que queremos que o Gato faça. Em nosso primeiro exemplo, diremos “Olá”. Para isso precisaremos de 2 blocos. Primeiro, usaremos um bloco que faz com que o nosso programa inicie. Veja na Figura 5. Perceba que a bandeira presente no bloco é a mesma bandeira localizada em cima do Gato.



Figura 5: Bloco “quando a bandeira for clicada”.

O segundo bloco você já conhece (veja a Figura 2). Para que o programa funcione, nós arrastamos os dois blocos para o centro da tela e os empilhamos da maneira mostrada na Figura 6.

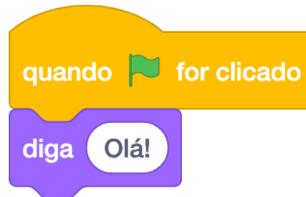


Figura 6: Código para dizer “Olá”.

Agora, clique na bandeira verde acima do Gato e veja o que acontece. Pode parecer algo muito simples por enquanto, mas esse é um dos primeiros passos para que possamos criar programas mais complexos. Nossa próximo passo é dar movimento ao Gato. Como? Continue lendo!

1.3 Andando para frente com o Gato

Continuando nossa jornada, precisamos aprender a dar movimento ao nosso personagem. Como fazemos isso? Vamos pensar nas etapas necessárias para fazer o personagem andar para a frente (lado direito da tela). Queremos que o Gato se move quando algum botão do computador for acionado. Seria ainda mais intuitivo se esse botão fosse a seta que aponta para a direita, não é? A boa notícia é que podemos fazer isso usando dois blocos.

Primeiro, arraste o bloco mostrado na Figura 7 para o meio da tela. Depois, troque (clicando no botão no bloco) “espaço” por “seta para direita”. Esse bloco criará uma ação quando clicarmos na seta para a direita. Para controlarmos qual ação queremos que o Gato realize, precisamos de outro bloco.

```
when space key is pressed
  [when green flag is clicked
    [repeat (1)
      [when space key is pressed
        [move (10) steps
          [right arrow key]
        ]
      ]
    ]
  ]

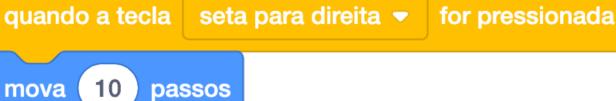
```

Figura 7: Bloco: “quando a tecla espaço for clicada”.

Para adicionar a ação de se movimentar, clique na seção de movimentos e selecione o bloco “mova 10 passos”. Perceba que é o mesmo bloco que vimos na Figura 3. Ao empilhar os blocos, você deve obter a combinação de comandos mostrada na Figura 8. Note que ao ler os blocos você obtém a frase:

Quando a tecla seta para direita for pressionada, mova 10 passos.

É parecido com a forma como falamos, não é? Essa é a ideia do Scratch e torna essa linguagem mais fácil de ser aprendida.



```
when space key is pressed
  [when green flag is clicked
    [repeat (1)
      [when right arrow key is pressed
        [move (10) steps]
      ]
    ]
  ]

```

Figura 8: Código para mover o Gato para a direita.

Para testar o movimento do Gato, pressione a seta para a direita do seu teclado. Perceba que nosso personagem se move apenas em uma direção (sempre para a direita). Para nosso jogo, também queremos que o Gato se move para trás (esquerda), para cima e para baixo. Criar código para podermos mover o Gato em todas as direções é a primeira etapa do nosso jogo. E é o que faremos a seguir!

1.4 Projeto: criando um jogo - primeiros passos

Como você já sabe, ao longo de nossos módulos criaremos um jogo com os conceitos apresentados. Antes de mais nada, é necessário que você **crie uma conta** no Scratch para poder salvar o progresso do seu código. Para isso, siga o tutorial que está no apêndice desse livro. Volte aqui depois de criar sua conta.

Nesse módulo inicial, daremos movimento ao nosso personagem. Para isso, utilizaremos blocos e técnicas que você já conhece. Por exemplo, usamos o comando “mova 10 passos” para deslocar o Gato para a direita. Na Figura 9, preste atenção no efeito desse código.

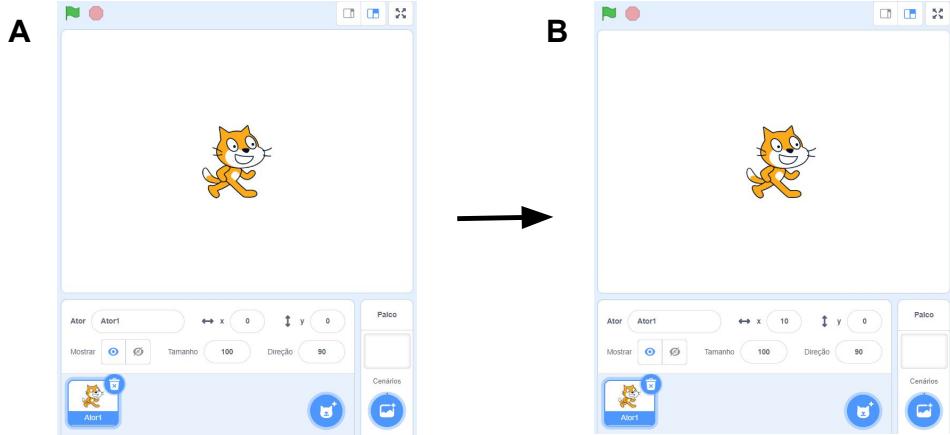


Figura 9: Gato antes e depois do bloco “mova 10 passos” ser aplicado.

Você percebeu o sutil efeito dessa movimentação nos números que aparecem abaixo do cenário? O que esses números representam e como podemos manipulá-los? Para entender, observe a Figura 10 e siga nossa explicação.



Figura 10: Indicadores para valores de x e y .

Perceba que ao lado da letra x existe uma flecha dupla na direção horizontal. Isso acontece porque o valor de x representa o quanto para esquerda ou para direita da tela o personagem se encontra. Note também que ao lado da letra y existe uma flecha dupla na direção vertical, indicando que o valor de y representa o quanto para cima ou para baixo da tela o personagem se encontra. A caixa a seguir resume essas ideias.

Ideia: O valor de x representa a posição horizontal do gato e o valor de y representa a posição vertical do Gato.

Agora que você compreende o significado das letras x e y , faça a seguinte adaptação em seu código. Troque o bloco “mova 10 passos” por “adicone 10 a x ”. Isso resultará no bloco mostrado na Figura 11.



Figura 11: Bloco de movimentação usando “adicone 10 a x ”.

Ao usar esses blocos, você deve perceber que obtemos o mesmo efeito que no código anterior. Tente mudar o valor de 10 para -10. Tentou? O gato está se movendo para o outro lado! Isso nos leva a uma conclusão importantíssima, descrita na caixa abaixo.

Ideia: Valores positivos e negativos têm efeitos contrários na movimentação do personagem na tela.

Para investigarmos a movimentação vertical, troque o bloco “adicone 10 a x ” pelo bloco “adicone 10 a y ”. Perceba que o gato agora se move para cima. Troque 10 por -10 e ele irá para baixo. Pegou a ideia? Se o conteúdo ainda estiver confuso, tentamos resumir os controles a seguir.

- Adicionar valores positivos a x gera um movimento para a direita;
- Adicionar valores negativos a x gera um movimento para a esquerda;
- Adicionar valores positivos a y gera um movimento para cima;
- Adicionar valores negativos a y gera um movimento para baixo;

Sabendo disso, podemos programar controles para o Gato usando as 4 setas do seu computador. Esse é o primeiro passo para completarmos nosso jogo. Por isso, **você será responsável** por implementar o código. Caso tenha dúvidas, deixaremos uma possível solução no apêndice.

Cuidado: Não se esqueça de salvar o seu projeto. O tutorial de salvamento está no apêndice.

Fim do módulo 1

2 Condicionais: se estiver frio, pegue um casaco

Talvez você ainda não dirija, mas provavelmente sabe como um semáforo funciona e o que cada cor representa para os motoristas.

Se a luz estiver verde, os carros podem andar.

Se estiver amarela, os motoristas devem ficar atentos.

Se estiver vermelha, eles devem parar.

Perceba o uso da palavra “se” representando uma **condição** relacionada à cor do semáforo. Situações em que condições estão presentes são muito comuns em nossas vidas. Você com certeza já escutou frases como: “se for comer, lave as mãos”, “se quiser aprender, estude”, ou “se estiver frio, pegue um casaco”. Acontece que condições também são muito comuns no mundo da programação. Para usá-las, precisaremos de **condicionais**. Nesse módulo, vamos entender melhor como usá-las e como elas tornarão nosso jogo mais interessante.

2.1 O que são condicionais?

A ideia por trás de condicionais é **verificar uma condição**. Em outras palavras, você precisa checar se algo é verdade e, caso seja, realizar uma ação. Por exemplo, observe a frase:

Se estiver frio, pegue um casaco.

Nela, você precisa verificar se está frio. Se estiver frio, você pegará um casaco. Caso contrário, não. A decisão nesse caso será entre realizar ou não uma ação. Podemos também criar uma outra forma de lidar com condicionais, deixando-as mais interessantes. Pense na seguinte frase:

Se estiver frio, tome sopa. Senão, tome um picolé.

Note que nesse caso uma ação precisa ser tomada independentemente da condição ser verdadeira ou não. Mesmo se não estiver frio, você precisará fazer algo. Portanto, podemos dizer que condicionais são usadas para tomar decisões sobre o que fazer. Essa é uma ideia chave para entender o uso delas na programação.

Ideia: Condicionais exigem que você verifique uma condição e são usadas para tomar decisões.

Se você está se perguntando como usamos condicionais no Scratch, é hora de aprendermos.

2.2 Condicionais no Scratch

Para começarmos a usar condicionais no Scratch, focaremos no bloco indicado pela Figura 12. Ele é usado para realizar uma ação caso a condição escolhida seja verdadeira.

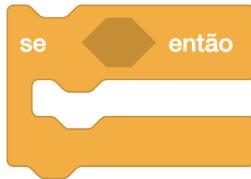


Figura 12: Bloco “se”, usado para testar condições.

Um exemplo de como utilizar o bloco é mostrado na Figura 13, na qual testamos se 1 é igual a 0 (sabemos que 1 não é igual a 0). Caso essa condição fosse verdadeira, o Gato diria “Olá”.



Figura 13: Usando o bloco “se” para verificar se 0 e 1 são iguais.

Nesse caso, o Gato não dirá “Olá!”. O interessante é que podemos usar um bloco parecido que nos deixa escolher que ação deve ser executada caso a nossa condição inicial não seja verdadeira. Veja a Figura 14:

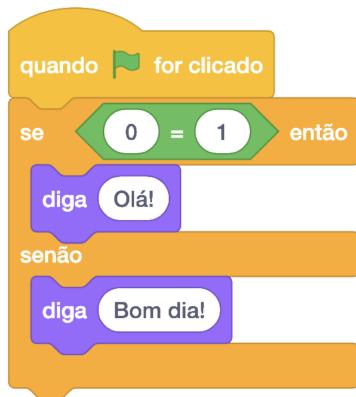


Figura 14: Usando o bloco “se” com “senão”.

Se usarmos esse bloco, o Gato dirá “Bom dia!”. Pegou a ideia? Agora que você já sabe usar esses dois importantíssimos blocos, está na hora de aplicá-los no seu jogo.

2.3 Projeto: criando um jogo - a maçã que foge

Cuidado! Lembre-se de acessar sua conta no Scratch e acessar o código que começamos a criar no módulo anterior!

Em nosso jogo, pretendemos fazer um gato coletar maçãs enquanto foge de uma bola. Agora que temos uma bagagem estabelecida com o conceito de condicionais, podemos pensar em escrever as linhas de código referentes ao gato pegar ou não as maçãs. Temos que desenvolver as seguintes ideias:

- Sempre que o gato encostar na maçã, ela deve ser movida para uma posição aleatória na tela, indicando que ela foi pega.
- Caso o gato não esteja encostando na maçã, ela deve permanecer parada.

Está claro que precisamos verificar uma condição: **o gato encosta ou não na maçã?** Então sem mais delongas, arregace as mangas e bora programar! Nossa primeiro passo é incluir o personagem Maçã! Para incluir um novo personagem, clique no botão azul no canto inferior direito (com um símbolo de um cachorro). Uma tela semelhante ao que mostramos na Figura 15 deve aparecer. Clique na maçã (Apple, em inglês).

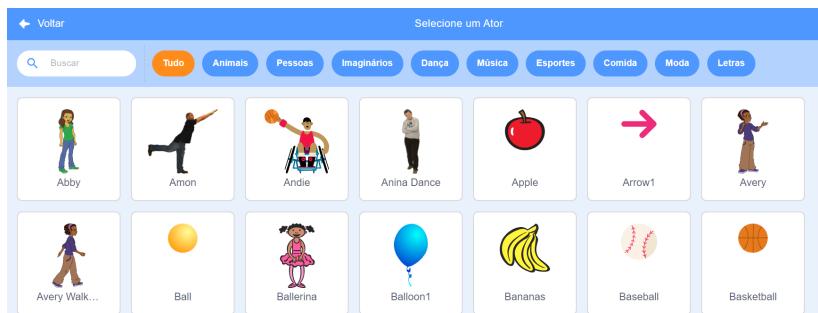


Figura 15: Tela para escolher um novo personagem.

Chegou o momento de criarmos as linhas de código para a maçã. Perceba que o código para a Maçã fica em uma página diferente do código para o Gato. Você pode selecionar o personagem sendo programado clicando no seu avatar (localizado abaixo da tela do jogo). Por exemplo, na Figura 16 estamos editando o código do Gato.

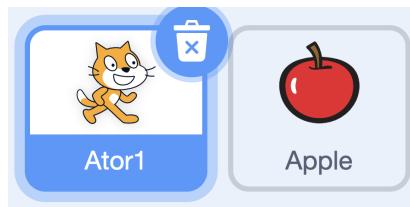


Figura 16: Selecionar o personagem sendo editado. Nesse caso, estamos editando o código para o Gato.

Ao contrário do que é feito na Figura 16, selecione a Maçã para editar. Depois, posicione os blocos mostrados abaixo. Nós queremos que o programa fique constantemente verificando se o gato encosta na maçã. Por isso, vamos usar o bloco “sempre”. Não se preocupe muito com ele por enquanto, o significado e a ideia por trás desse bloco serão apresentados no módulo seguinte quando aprendermos sobre “loops”.



Figura 17: Código inicial para a maçã

Dentro dessa estrutura deveremos colocar uma condicional que verifica se o gato encosta ou não na maçã. Agora a tarefa é sua: procure por blocos que façam sentido e verifique se suas ideias funcionam! Depois de quebrar a cabeça, verifique a resposta proposta no apêndice. Lembre-se que em programação raramente existe uma única solução para um problema.

Cuidado! Não se esqueça de salvar seus códigos, vamos utilizá-los nos próximos capítulos!

Fim do módulo 2

3 Loops: repetindo a mesma ação

Um segundo conceito que nós vamos explorar é o de repetição. Digamos que você está programando um robô para pegar 3 fatias de bolo para você. Uma maneira de fazer isso é falando:

Pegue uma fatia de bolo.

Pegue uma fatia de bolo.

Pegue uma fatia de bolo.

Repetitivo, não é mesmo? Talvez você esteja pensando “eu poderia pedir para ele pegar 3 fatias de bolo uma única vez”. Bem, isso envolve repetir uma ação, o que aprenderemos a fazer com loops. Com loops poderemos dizer a seguinte frase:

Repita 3 vezes: pegue uma fatia de bolo.

Perceba que loops nos permitem **repetir** algo diversas vezes automaticamente, o que é muito comum no nosso cotidiano. Essa ideia tornará nosso jogo muito mais interessante e nosso código muito mais prático. Se você ainda está confuso sobre o que são loops, fique tranquilo. Vamos, com calma, entender esse conceito.

3.1 O que são loops?

Loops são usados na programação para repetir uma ação, assim como mostramos anteriormente. Nós podemos usá-los em 3 situações:

Ideia:

1. Para repetir algo um número específico de vezes. Exemplo: ande 5 passos.
2. Para repetir algo até que uma condição seja atendida. Exemplo: ande até chegar em casa.
3. Para repetir algo para sempre. Exemplo: ande para sempre.

Por exemplo, no último módulo quando programamos nosso jogo, queríamos que a Maçã desaparecesse **sempre** que o Gato a tocasse. Por isso, usamos a terceira forma de loop (repetir uma ação para sempre). Perceba que sempre, nesse contexto, significa “até que o jogo acabe”.

Dica: Loops indicam repetição e podem existir em 3 diferentes formas.

Note que, dependendo da situação, você precisará escolher qual tipo de loop você quer em seu código, o que pode ser desafiador. A melhor maneira de entender a diferença entre eles é testando. Agora que sabemos a teoria, está na hora de colocar em prática os conceitos aprendidos!

3.2 Loops no Scratch

Nós aprenderemos como usar os três tipos de loop usando blocos no Scratch. Vamos começar pela ideia de repetir uma ação por um número predefinido de vezes. Imagine que gostaríamos que o Gato andasse para frente 5 vezes esperando 1 segundo entre cada movimento. Podemos fazer isso com o código a seguir:



Figura 18: Usando loops com um número determinado de repetições.

Perceba que nesse exemplo nós usamos o bloco de repetir (por 5 vezes) a ação de ir para frente. Também esperamos 1 segundo entre as ações para que você possa visualizar o que acontece. Como é a primeira vez que mostramos esses blocos, experimente mudar os valores do número de repetições e do tempo entre cada movimento.

O que podemos fazer se quisermos andar até chegar em uma posição exata? Por exemplo, digamos que nosso personagem sairá do meio da tela (posição com $x = 0$ e $y = 0$) e desejamos que ele ande até a posição onde $x = 50$ e $y = 0$. Como podemos fazer isso sem calcular quantos passos precisamos dar? Veja o seguinte bloco de código.



Figura 19: Usando loops até que uma condição seja atendida.

Nesse caso, nossa condição é ter $x = 50$. Outra observação importante é que se você comparar o resultado dos loops, estamos fazendo o mesmo movimento. A única diferença é que usamos blocos diferentes. Isso nos leva a uma conclusão importante, escrita na caixa abaixo:

Ideia: É possível realizar a mesma ação usando loops diferentes.

O último loop que veremos é o que repete uma ação para sempre. O bloco será o mesmo que utilizamos quando estávamos fazendo nosso jogo no módulo anterior. Para testá-lo, use o seguinte código:



Figura 20: Usando loops para sempre realizar uma ação.

Nesse caso, o Gato irá andar até que o programa pare de rodar. Loops para sempre serão importantes na implementação do nosso jogo, uma vez que eles permitirão controlar o movimento de um novo personagem. Ficou curioso? Então vamos ver como adicioná-los em nosso jogo.

3.3 Projeto: criando um jogo - o rival que não para de nos perseguir

Cuidado! Lembre-se de acessar sua conta no Scratch e acessar o código que estamos construindo desde o primeiro módulo,

Anteriormente, programamos a função para o Gato coletar maçãs. Para isso, utilizamos loops sem nos preocuparmos com os detalhes, lembra? Abra o progresso do jogo que está salvo e perceba o uso do bloco “sempre”.

Pois bem, onde mais precisamos implementar loops em nosso jogo? Que tal programarmos uma bola em que o gato deve evitar tocar enquanto tenta coletar as maçãs? Para isso, vamos primeiro criar o personagem Bola usando a imagem “Beachball”. Caso não se lembre de como adicionar um personagem, dê uma espiada na figura 15, localizada na seção 2.3.

Depois de selecionar a tela referente ao personagem Bola (refresque a memória de como completar essa etapa olhando a figura 16 na seção 2.3), vamos escrever o código para gerar seu movimento perpétuo. Hora de pensar nas propriedades que a bola deve possuir em nosso jogo:

- Em primeiro lugar, precisamos posicionar a bola em um ponto inicial afastado do gato.
- Precisaremos de um loop do tipo “sempre” para criar o movimento perpétuo da bola.
- Dentro do bloco sempre, precisaremos colocar um código que faz a bola se movimentar.
- Dentro do bloco sempre, deveremos avaliar se a bola encosta na parede, para então fazê-la retornar..

Posição inicial da bola

Sabemos que a posição inicial do gato será onde $x = 0$ e $y = 0$. Vamos posicionar a bola em um ponto afastado do gato, utilizar o comando “Vá para X e Y”, utilizando valores de X e Y bem distantes de 0 (onde sabemos que o gato estará).

Porém, isso dá origem a um pequeno problema: se a posição inicial da bola for sempre a mesma, teremos sempre o mesmo movimento de bola e um jogo muito repetitivo.

Tente solucionar o problema de dar uma pequena aleatoriedade à posição da bola.

Pensou? Nós sugerimos a seguinte solução (que não é a única)



Figura 21: Posições randômicas da bola

Movimentação da bola

Ótimo, agora temos que nos concentrar no loop de movimento. Queremos obter um loop em que a bola se mantém em movimento e, ao bater na parede, retorna.

- Para o movimento da bola, podemos usar o bloco “mova passos”
- Para o ato de retornar quando a bola atingir a parede temos um bloco pronto chamado “se tocar na borda, volte”

Tente fazer o código completo para o movimento da bola por conta própria! Deixaremos umas solução no apêndice.

Cuidado! Não se esqueça de salvar seus códigos, vamos utilizá-los nos próximos capítulos!

4 Variáveis: caixas, números e palavras

Imagine que você recebe uma bolacha e uma maçã, mas está sem fome agora. Para guardar sua comida para depois, você coloca a bolacha em uma caixa e a maçã em outra. A fim de saber o que tem dentro de cada caixa, você escreve “bolacha” em uma delas e “maçã” na outra. Assim você lembrará mais tarde qual caixa contém cada comida.

Nessa situação, você está guardando uma informação para usar depois. No mundo da programação, também é comum precisarmos guardar uma informação para outro momento. Como fazemos isso? Usando **variáveis**. Nesse módulo nós vamos explorar o que são variáveis e como podemos usá-las para melhorar nosso jogo.

Note que, diferentemente de loops e condicionais, variáveis não são operações e sim estruturas que usamos para nos ajudar. Se isso parece difícil, fique tranquilo. Te explicaremos o que são variáveis, como usá-las e quando elas serão úteis em nosso jogo.

4.1 Entendendo variáveis

Como vimos anteriormente, uma variável é usada para “guardar” um nome ou um valor. Podemos imaginá-las como uma caixa na qual colocamos algo que precisaremos depois. Por exemplo, podemos guardar nomes, números ou até um texto inteiro. Para usarmos uma variável nós damos um nome a ela. Uma boa prática é escolher algo que esteja relacionado ao valor da variável. Por exemplo, podemos guardar a palavra “azul” na variável “minha cor preferida”.



Figura 22: Variável “minha cor preferida”, guardando o valor “azul”

Ideia: A parte interessante das variáveis é que temos liberdade para manipular o que está guardado nelas.

Por exemplo, digamos que você mude de ideia e que sua cor preferida agora é verde, não mais azul. Você pode atualizar o valor na caixa para “verde”, assim como na figura a seguir. Perceba que ao mudar o conteúdo de uma variável, seu nome é preservado. Ou seja, alteramos a cor guardada na

caixa de “azul” para “verde” mas o nome da caixa permaneceu o mesmo.

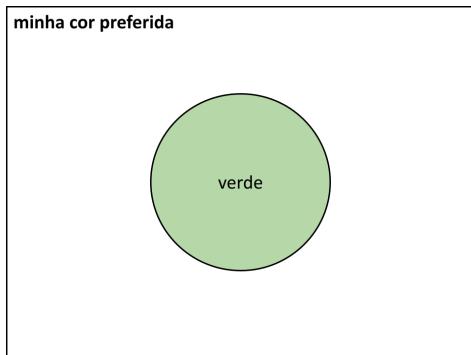


Figura 23: Variável “minha cor preferida”, guardando o valor “verde”

Ideia: Variáveis são usadas para armazenar informações e podemos mudar o valor guardado nelas.

Além disso, podemos utilizar o valor guardado em uma variável para armazenar outras informações mais elaboradas. É possível, por exemplo, criar uma variável que armazena a idade do programa Scratch.

Dica: Para fazer isso, você pode criar um programa que atualiza a idade a cada ano que passa. Basta criar uma variável que subtrai 2007 (ano de criação do Scratch) do ano atual.

Com o que já dissemos, você conhece as principais ideias por trás de variáveis. Assim como outros conceitos que vimos anteriormente, a teoria por trás de variáveis é muito simples. Na prática, porém, precisamos saber utilizá-las da forma correta. Por isso vamos aprender a implementá-las no Scratch.

4.2 Variáveis no Scratch

Para usarmos uma variável no Scratch, nós primeiro precisamos criá-la. Para fazer isso, na parte dos blocos, vá para a seção “Variáveis” e clique em “Criar uma Variável”. Depois, digite “Nome” no espaço para o nome da sua variável. Após essas etapas, temos uma nova variável! A Figura 24 mostra o passo a passo.



Figura 24: Guia ilustrado de como criar uma nova variável no Scratch. Lembre-se que você pode escolher qualquer nome para sua variável.

Dica: Quando nomeando variáveis, escolha nomes que te ajudem a lembrar o que elas representam. Por exemplo, para criar uma variável que guarda a sua idade, chame-a de “Minha idade”.

No Scratch, sua variável sempre começará com o valor 0 (zero). Para guardarmos um novo valor, precisaremos dos blocos da Figura 25. Perceba que nesse caso, nós trocamos o valor 0 que estava guardado na variável por “Lucas”. Essa troca é similar a que fizemos quando trocamos a nossa cor preferida de azul para verde.



Figura 25: Iniciando uma variável chamada “Nome” no Scratch.

4.3 Projeto: criando um jogo - Registrando a pontuação

Agora que você tem em mãos a ideia de variáveis, podemos dar os toques finais a nosso jogo. Até agora, temos praticamente todas as mecânicas importantes do nosso jogo - o gato consegue pegar maçãs, e já temos a bola que o perseguirá. O que está faltando? A pontuação! Os pontos aumentarão quando o Gato coletar maçãs e irão para 0 se a bola encostar nele.

Ideia: Deveremos incluir algo que consiga marcar quantas maçãs o gato conseguiu pegar!

Você deve desconfiar que vamos fazer isso utilizando uma variável. Vamos criar, portanto, uma variável e chamá-la de “Maçãs”. Caso você não lembre como criar uma variável, veja a Figura 24. Note também que devemos colocar para essa variável a opção “para todos os atores”. Devemos fazer

isso porque o valor de maçãs não depende apenas das maçãs, mas também se a bola encosta no gato.

Tudo bem, e o que queremos fazer com o valor da variável maçãs?

1. Queremos que o número de maçãs comece em zero toda vez que o jogo for iniciado
2. Queremos que o número de maçãs coletadas aumente em 1 (somar 1) para cada vez que o gato encostar na maçã
3. Queremos que o número de maçãs coletadas zere se a bola encostar no gato

Para a propriedade 1, não precisaremos avaliar nenhuma condição. Basta, portanto, mudar o valor da variável logo no começo do código.

Já para as propriedades 2 e 3 será necessário colocar nosso conhecimento de condicionais em prática. Tente solucionar esse problema por conta própria. Você precisará escolher corretamente as condições e associar a elas mudanças nos valores da variável “maçãs”.

Perceba! Para esse problema, você já está usando condicionais, variáveis e loops ao mesmo tempo! A maioria dos problemas reais de programação envolvem justamente a mistura dessas ideias que você já conseguiu colocar em prática. Parabéns!

Fim do módulo 4

5 Apêndice

5.1 informações para criar uma conta e salvar

5.2 Solução Jogo módulo 1

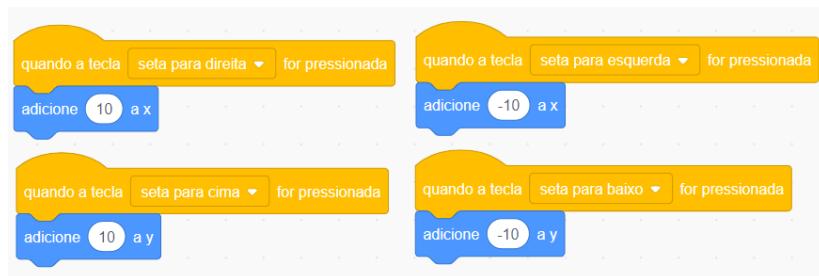


Figura 26: Movimentação completa.

5.3 Solução Jogo módulo 2



Figura 27: Solução módulo 2

5.4 Solução Jogo módulo 3



Figura 28: Solução do capítulo 3

5.5 Solução Jogo módulo 4

Para os comandos relativos ao personagem “Maçã”



Figura 29: Solução do capítulo 4 - parte da maçã

Para os comandos relativos ao personagem “Gato”

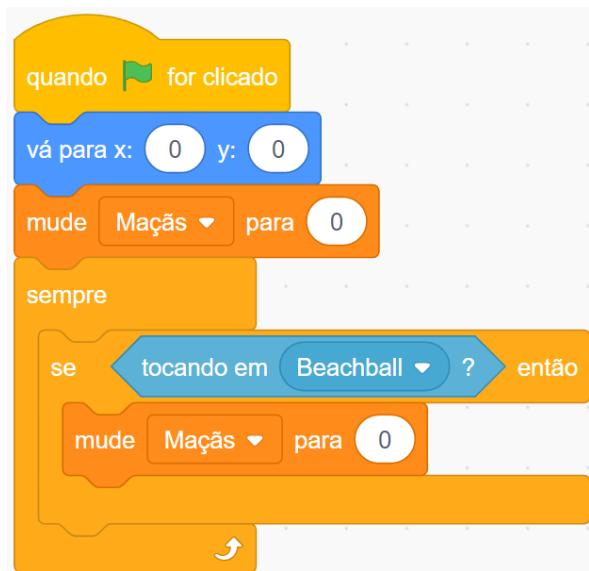


Figura 30: Solução do capítulo 4 - parte da gato