

# Aula06 - Procedimentos



## CAÇA MOSQUITO

Sérgio Carlos Portari Júnior

# O QUE VAMOS APRENDER?

## JOGO CAÇA MOSQUITO

### OBJETIVO

Caçar o mosquito voando pela tela

### INFORMAÇÕES

Mosquito terá 3 vidas.

Jogador tem 10 segundos para eliminar mosquito.

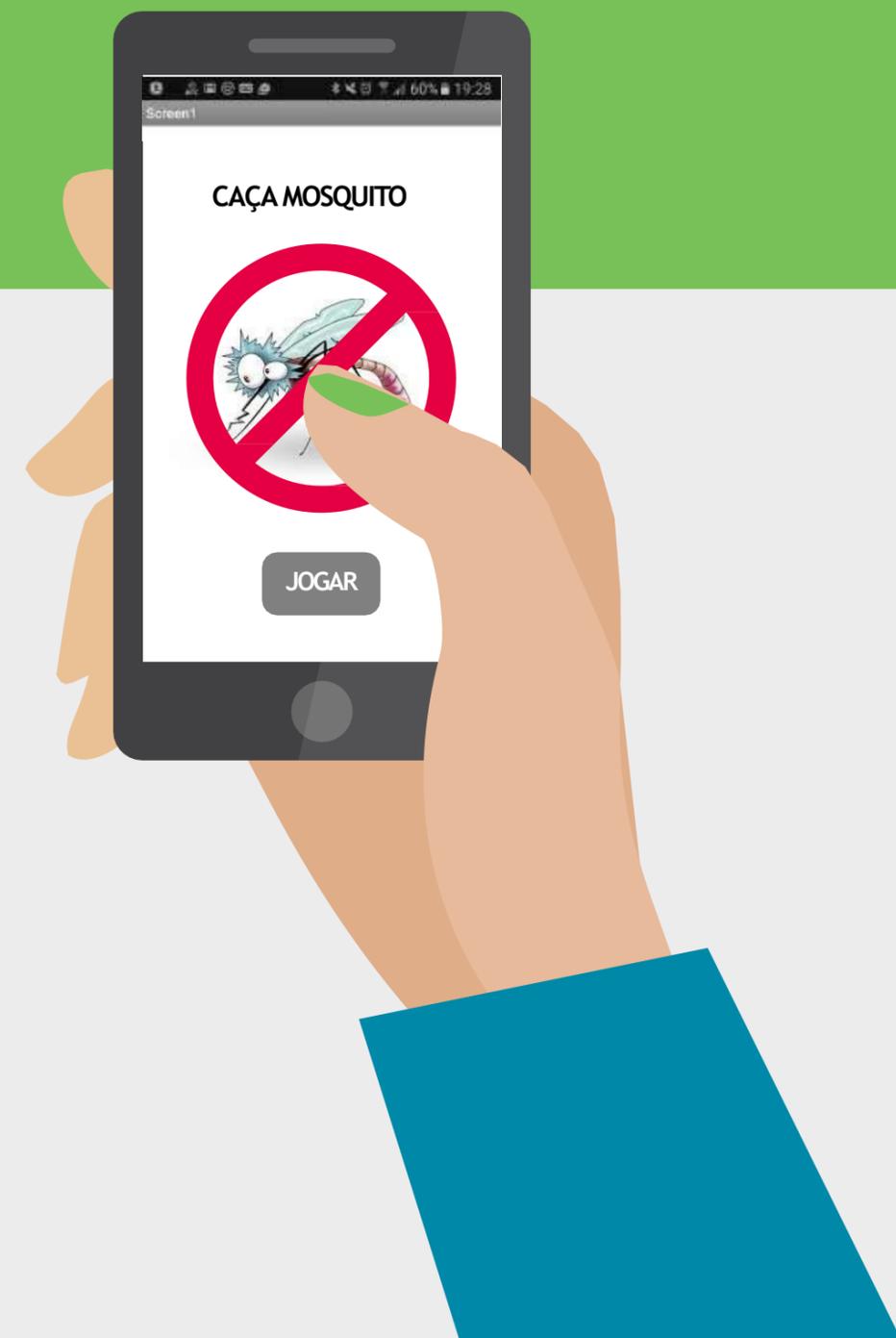


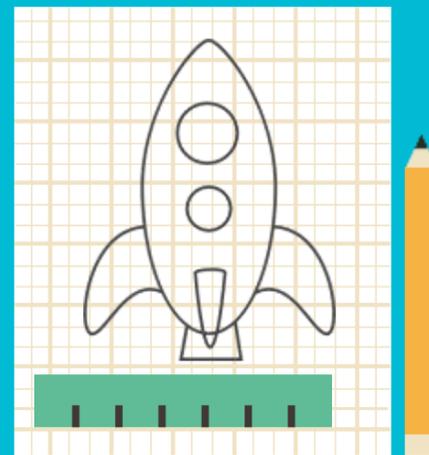
Fazer um mosquito se mover pela tela.

Adicionar vida ao mosquito.

Adicionar um tempo para matar o mosquito.

Incluir níveis de dificuldade ao jogo.





# CRIANDO UM NOVO PROJETO

# CRIANDO UM NOVO PROJETO

Vamos iniciar criando um novo projeto no AppInventor.

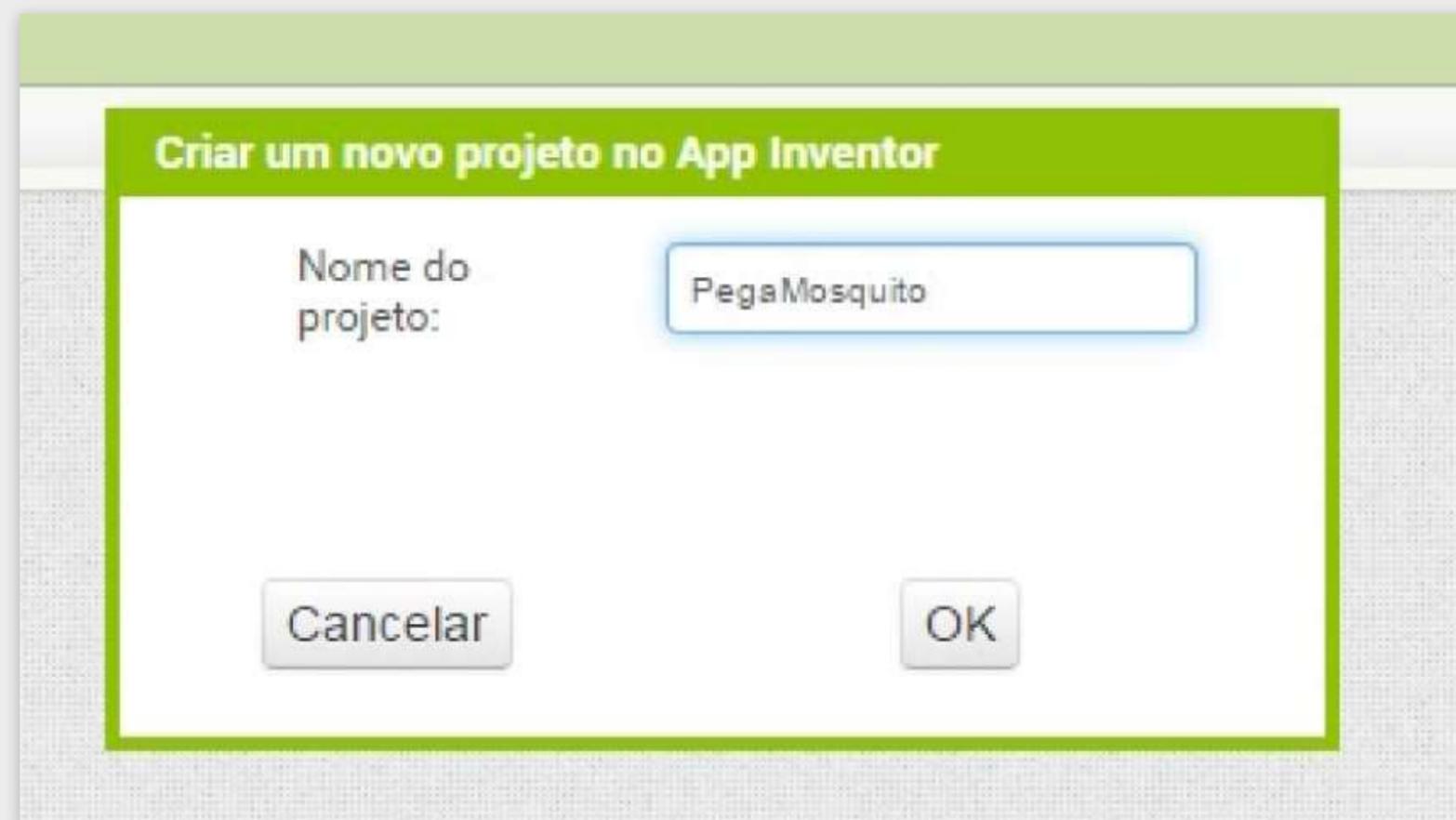


The screenshot displays the MIT App Inventor 2 Beta web interface. At the top left is the MIT App Inventor logo (a green Android head) and the text "MIT App Inventor 2 Beta". To the right are three menu items: "Projetos", "Conectar", and "Compilar", each with a downward arrow. Below the header is a green bar containing three buttons: "Iniciar novo projeto ..." (highlighted), "Apagar Projeto", and "Publicar na Galeria". Underneath is a section titled "Meus Projetos" with a light green background. Below this title is a table with two columns: "Nome" and "Data de Criação".

Nome	Data de Criação
------	-----------------

# CRIANDO UM NOVO PROJETO

Vamos definir o nome do novo projeto como **“PegaMosquito”** e clicarem OK.

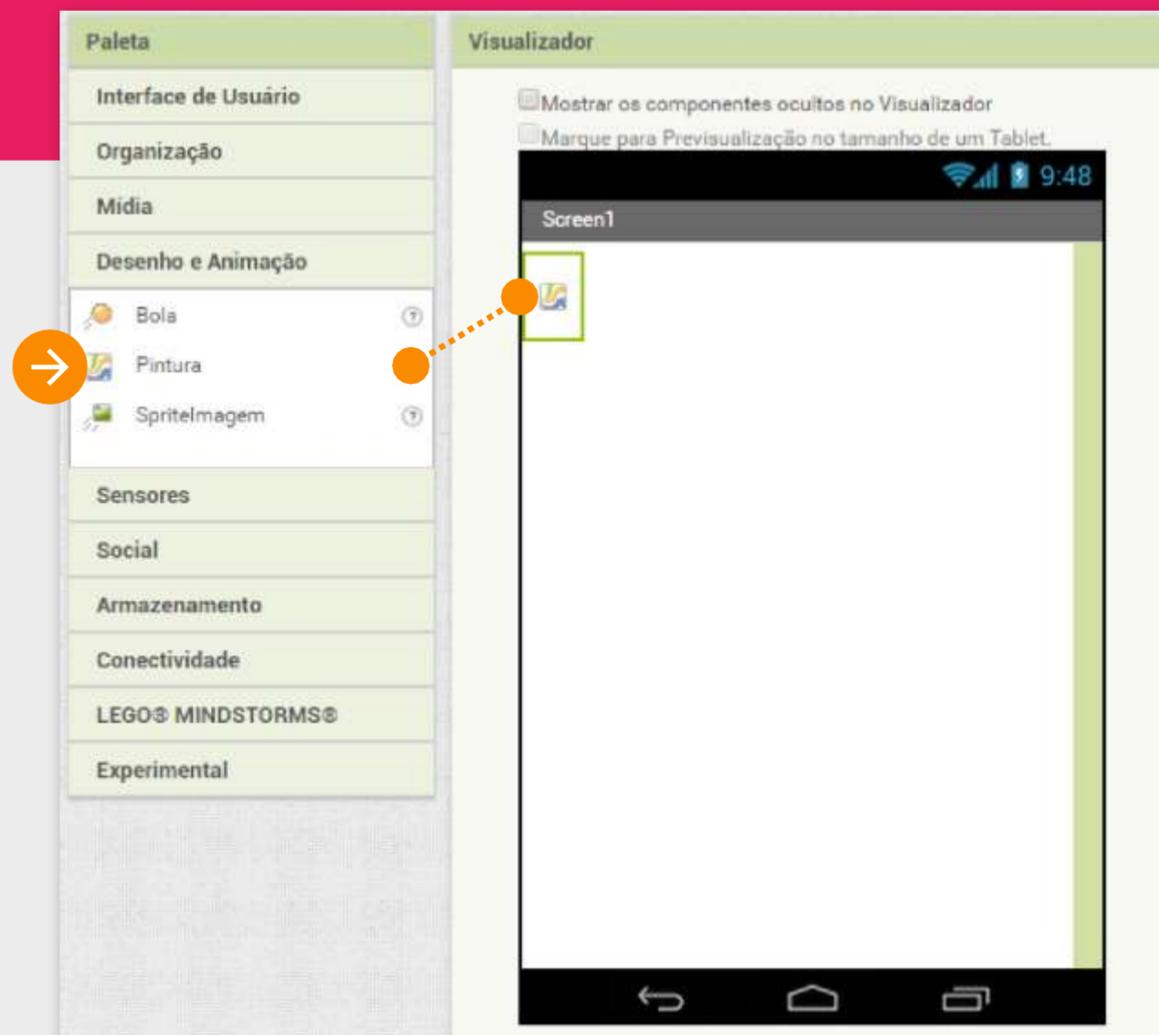


The image shows a dialog box titled "Criar um novo projeto no App Inventor". Inside the dialog, there is a label "Nome do projeto:" followed by a text input field containing the text "PegaMosquito". At the bottom of the dialog, there are two buttons: "Cancelar" on the left and "OK" on the right.

# DEFININDO A INTERFACE



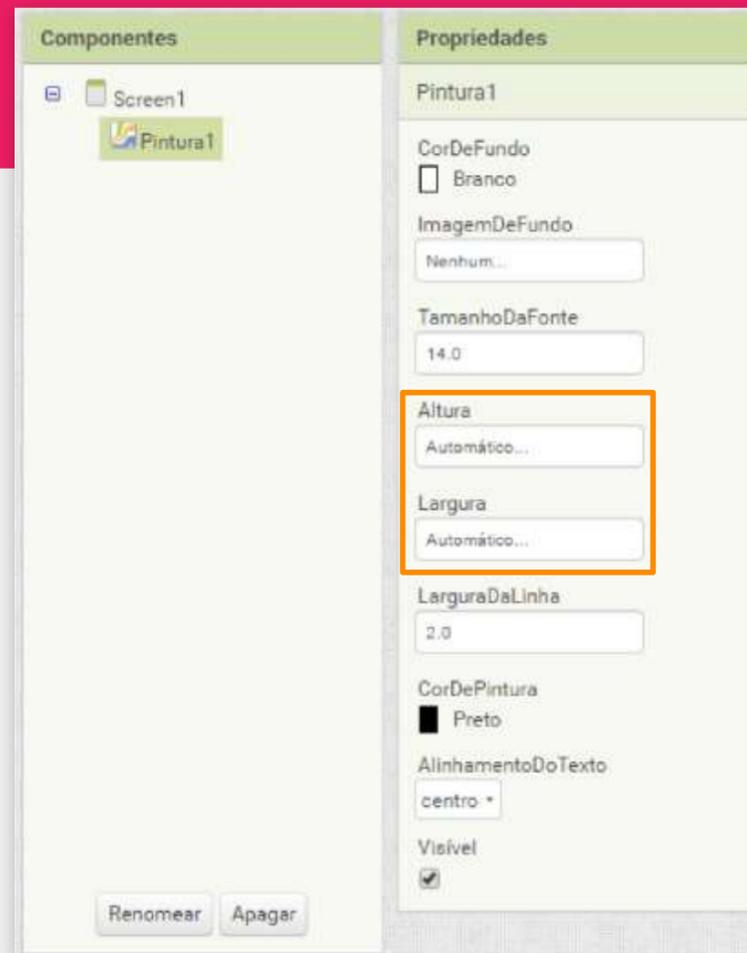
# DEFININDO A INTERFACE



Vamos começar definindo a nossa interface.

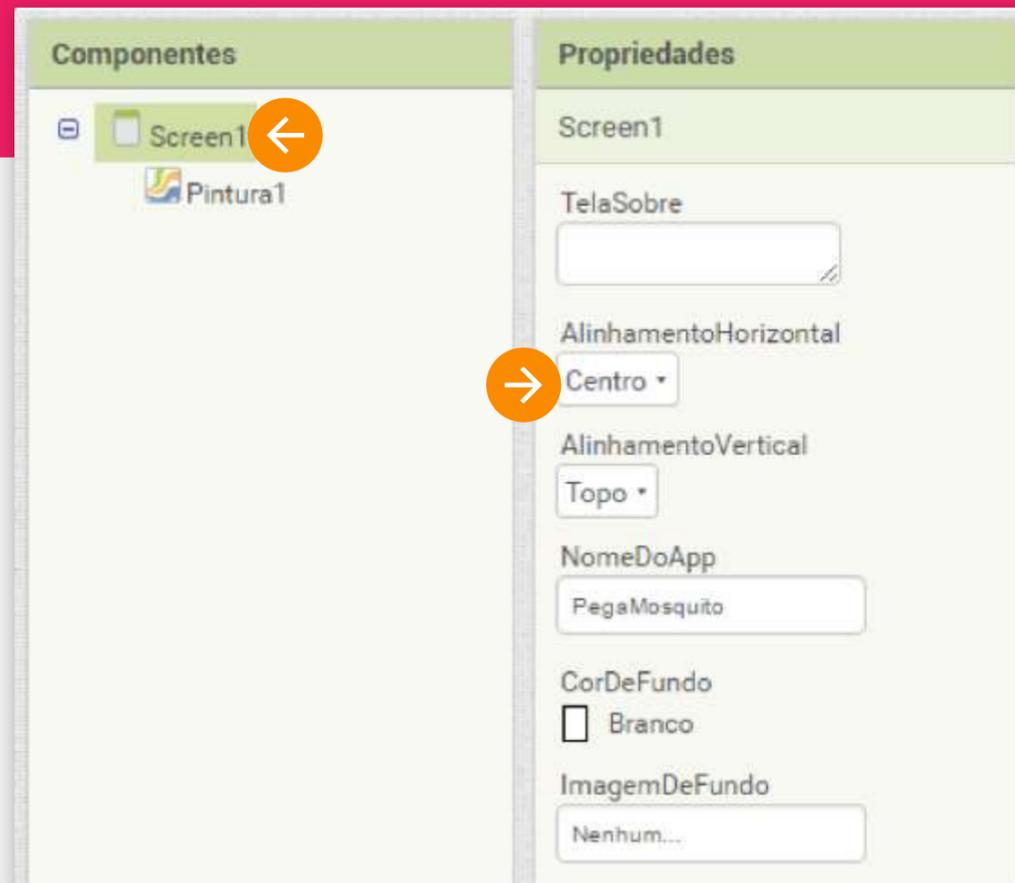
Primeiro vamos clicar na paleta **“Desenho e Animação”** e arrastar o componente **“Pintura”** para dentro da nossa tela (no visualizador).

# DEFININDO A INTERFACE

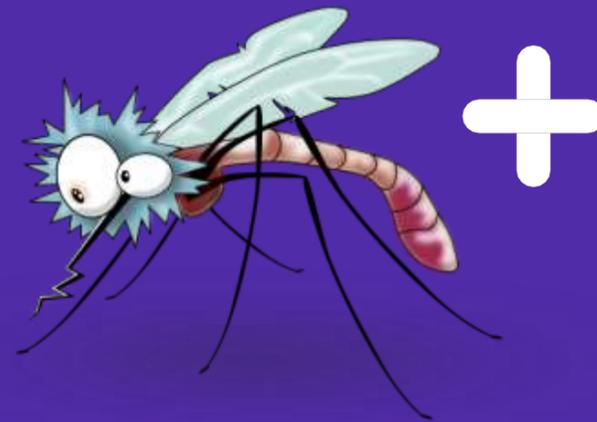


Vamos alterar as propriedades da pintura para que ela possua altura de 300 pontos e largura de 300 pontos.

# DEFININDO A INTERFACE



Para alinhar a pintura no centro da tela, vamos selecionar o componente "Screen1" e na propriedade "AlinhamentoHorizontal" vamos selecionar "Centro".

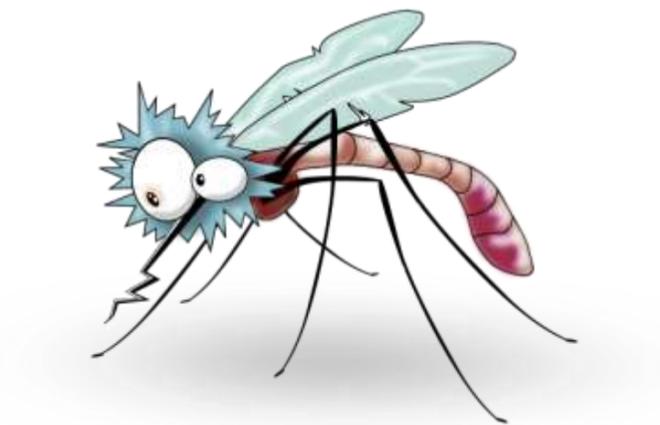


ADICIONANDO  
0  
MOSQUITO

# ADICIONANDO O MOSQUITO

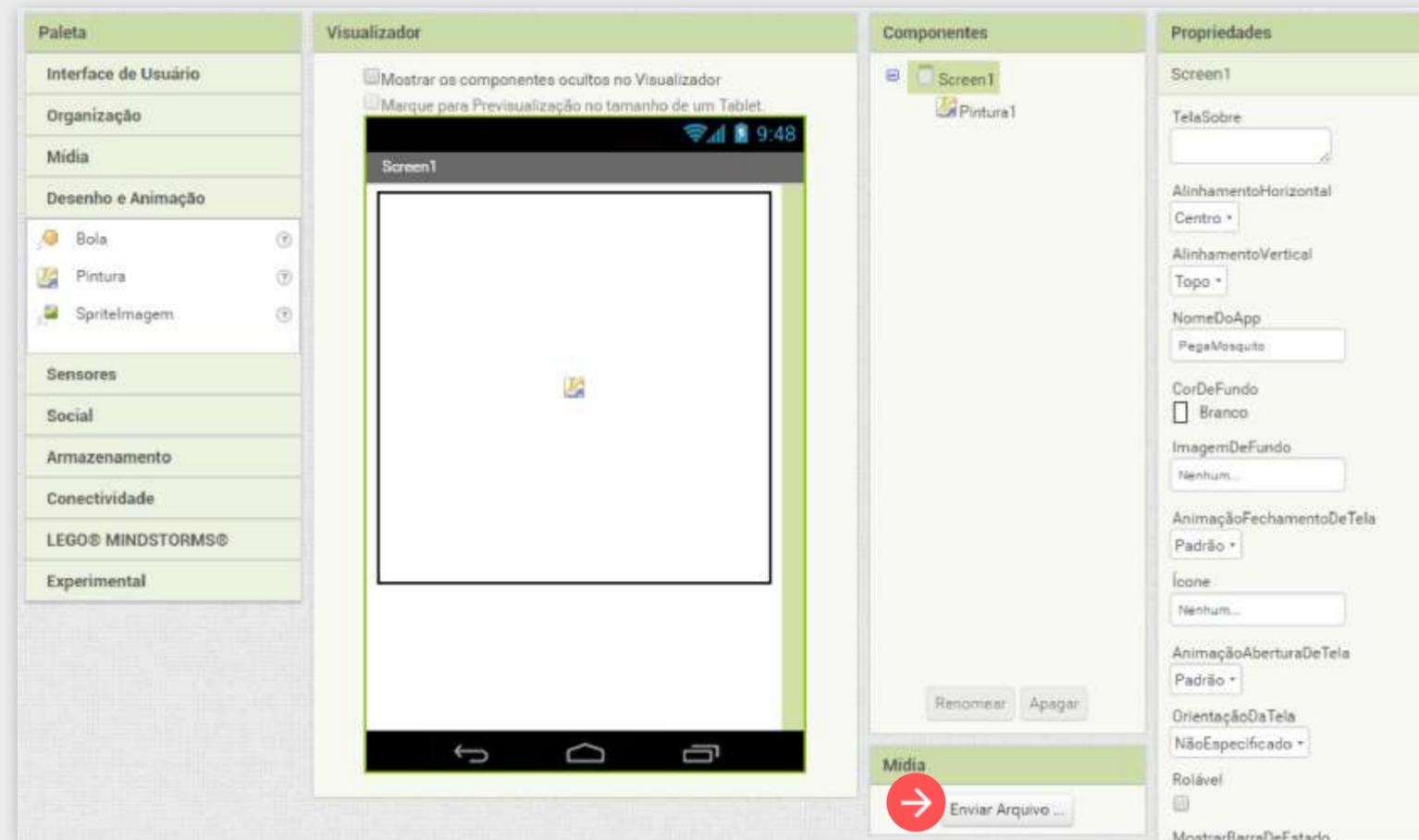
Primeiro baixe a imagem do mosquito aqui:

<http://www.sergioportari.com.br/wp-content/uploads/2019/03/mosquito.jpg>



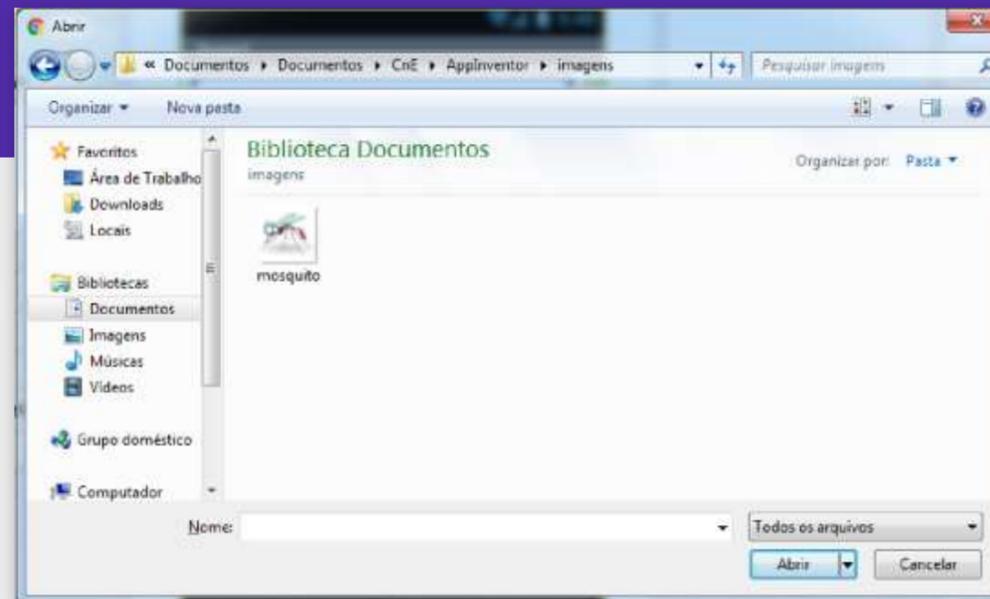
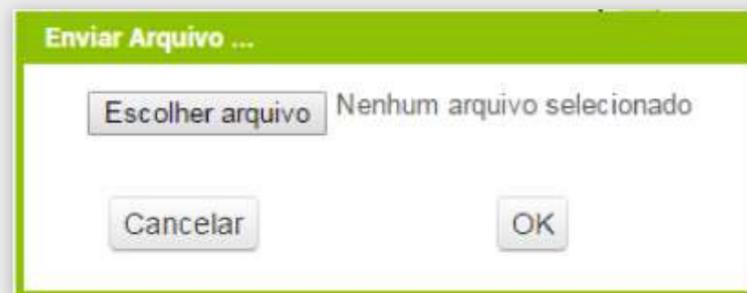
# ADICIONANDO O MOSQUITO

Insira a figura do mosquito clicando no botão “**Enviar arquivo...**” em Mídia.



# ADICIONANDO O MOSQUITO

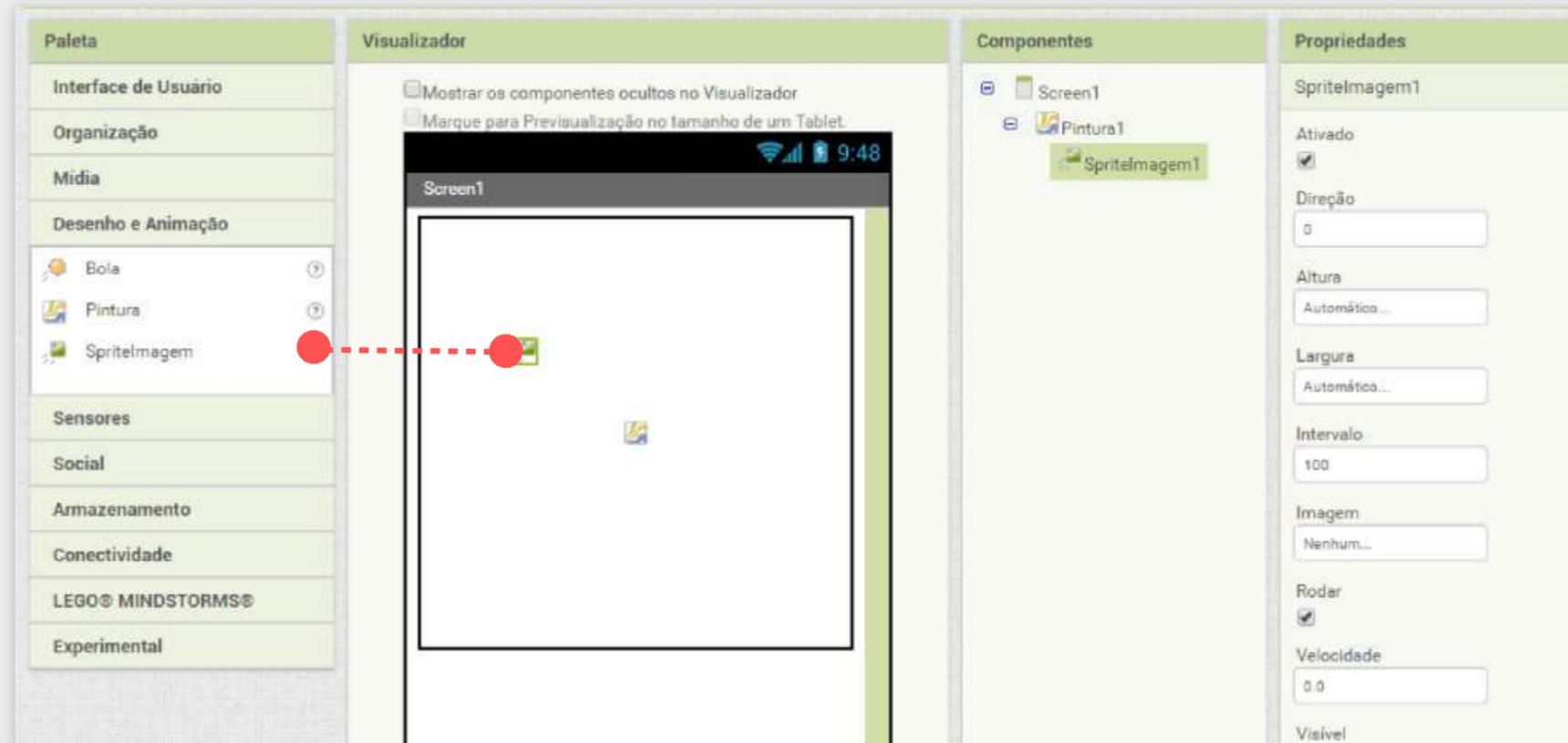
Clique no botão **“Escolher arquivo”** e selecione o arquivo com nome **“mosquito.png”** que você baixou.



# ADICIONANDO O MOSQUITO

Para colocarmos o mosquito na nossa tela, precisamos usar um tipo de imagem chamada de **“Sprite”**.

Selecione o componente **“SpriteImagem”** na Paleta de **“Desenho e Animação”** e arraste para a **“Pintura”**.

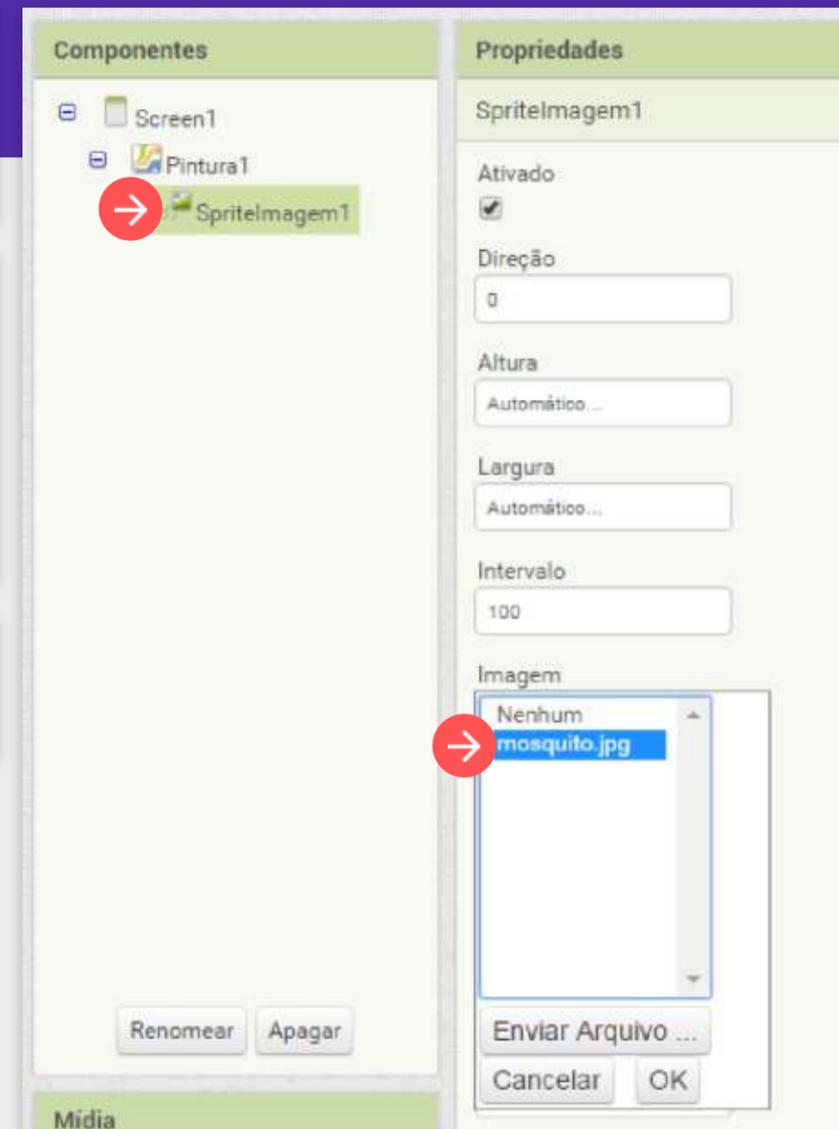


# ADICIONANDO O MOSQUITO

Agora devemos definir a imagem dessa Sprite.

Para fazer isso, nas Propriedades do componente **"SpriteImagem1"**, clique no campo **"Imagem"** e selecione o arquivo **"mosquito.png"**.

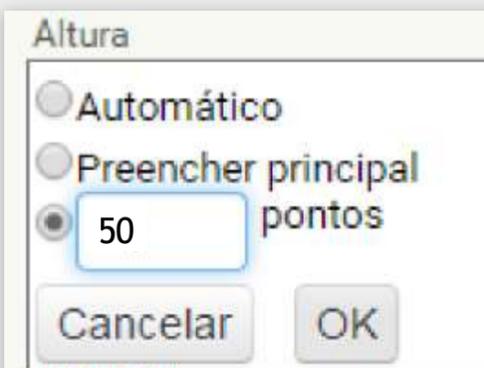
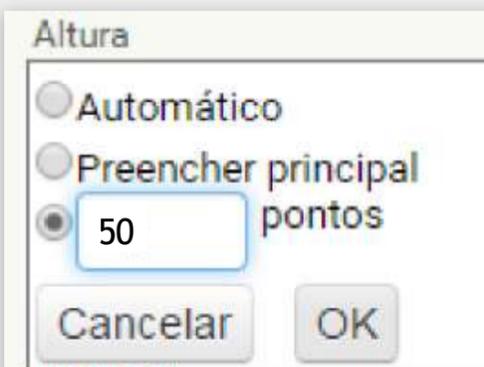
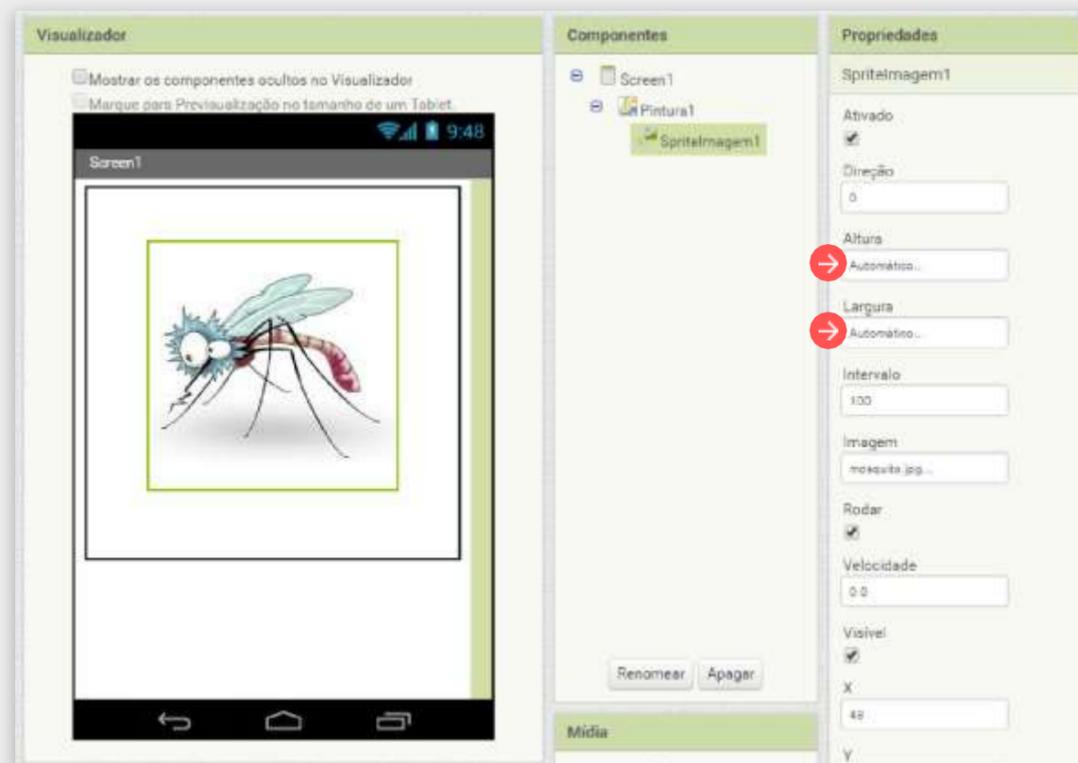
Clique em OK



# ADICIONANDO O MOSQUITO

Agora vamos redimensionar a imagem para que ela fique do tamanho desejado.

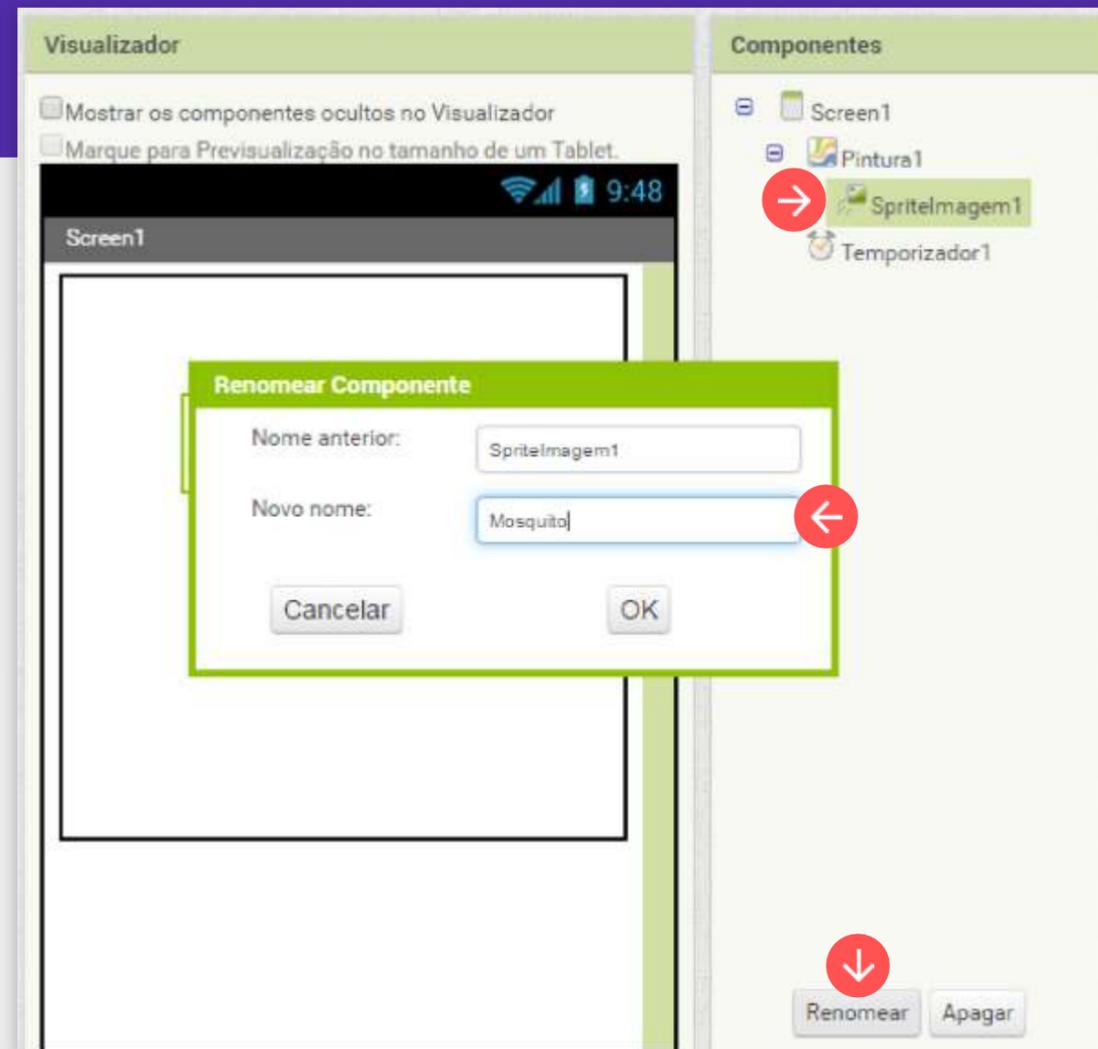
Mude a altura e largura do mosquito para 50.

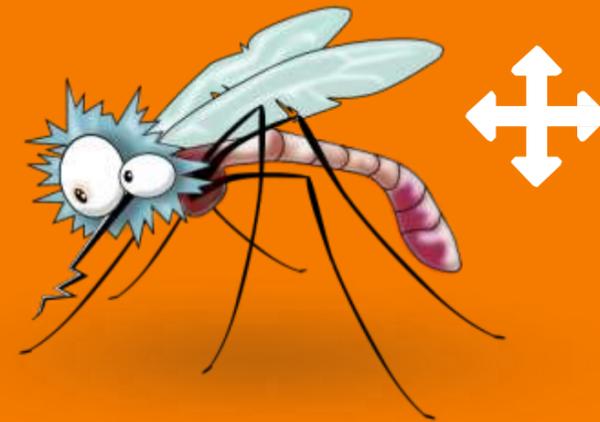


# ADICIONANDO O MOSQUITO

Para ficar mais fácil reconhecer que o componente Spritelmagem1 é a imagem do nosso mosquito, vamos mudar o nome dele para **“Mosquito”**.

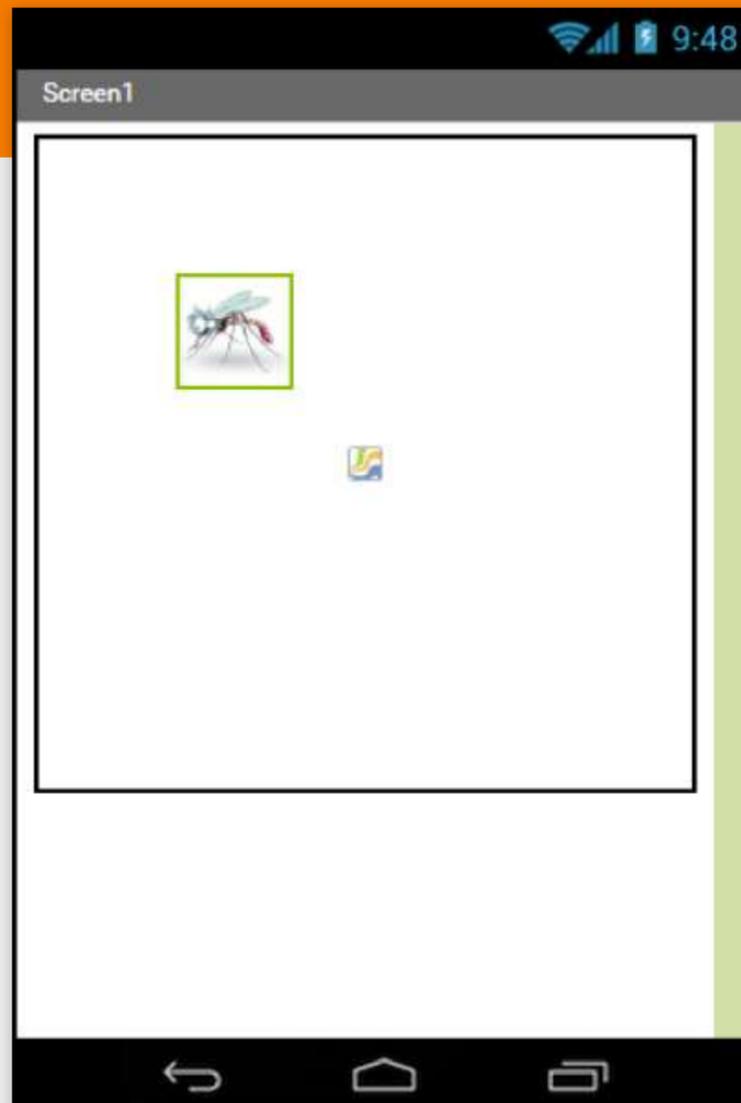
Selecione o componente e clique em Renomear. Digite **“Mosquito”** e clique em OK.





MOVER O  
MOSQUITO  
PELA TELA

# MOVER O MOSQUITO PELA TELA



Por enquanto só temos um mosquito parado na tela.

**AGORA VAMOS FAZÊ-LO SE MOVIMENTAR.**

# MOVER O MOSQUITO PELA TELA

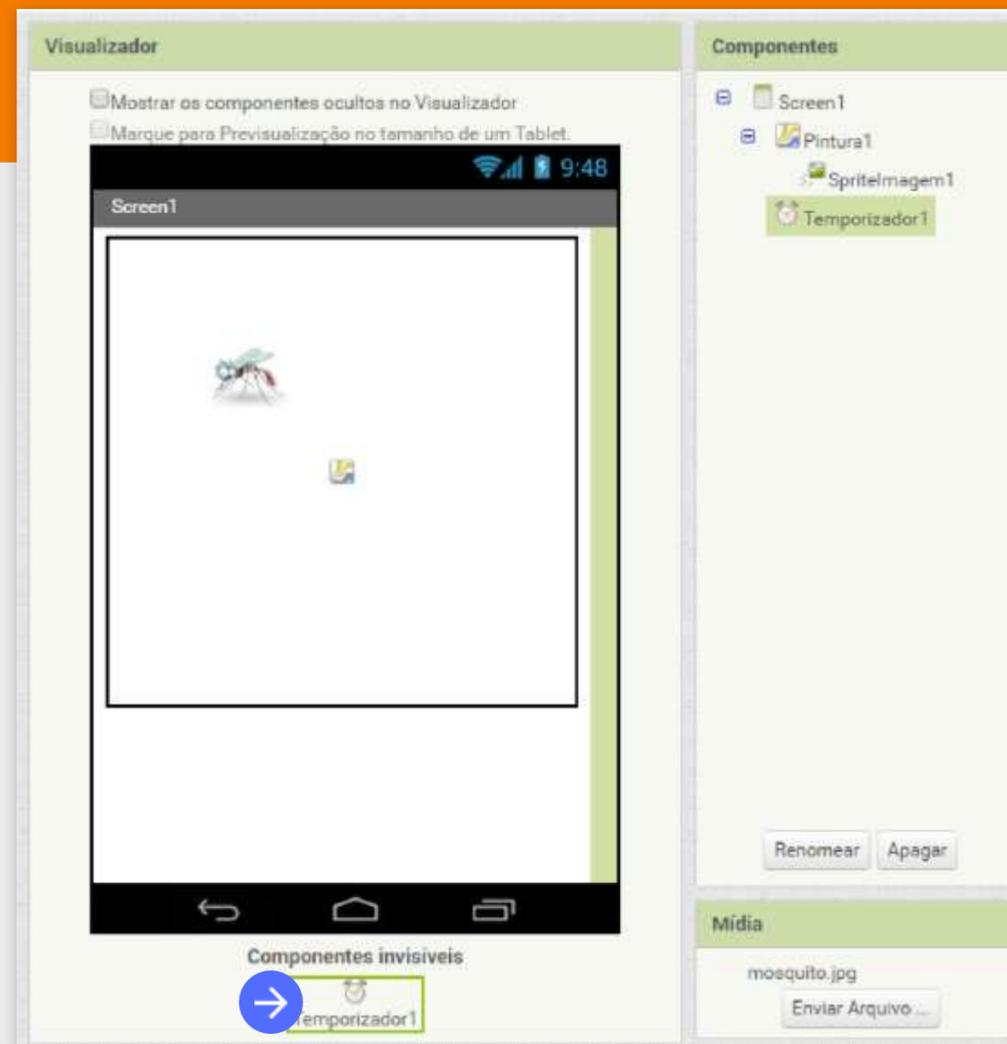
Para o mosquito se movimentar iremos utilizar o componente **Temporizador**. Este será responsável por mandar o mosquito se movimentar a cada meio segundo.

Devemos clicar em **Sensores** e arrastar o componente **Temporizador** até a tela.

The image shows the Android Studio interface. On the left, the 'Paleta' (Palette) window is open to the 'Sensores' (Sensors) category. The 'Temporizador' (Timer) component is highlighted with a blue circle. A dashed blue line connects this circle to the 'Visualizador' (Preview) window. In the 'Visualizador' window, the 'Temporizador2' component is being dragged onto the 'Screen1' canvas. The 'Screen1' canvas shows a mosquito icon and a timer icon. The status bar at the top of the 'Screen1' canvas shows the time 9:48.

# MOVER O MOSQUITO PELA TELA

O componente **Temporizador** é um componente invisível na tela, ele fica apresentado na parte inferior do bloco “**Designer**”.

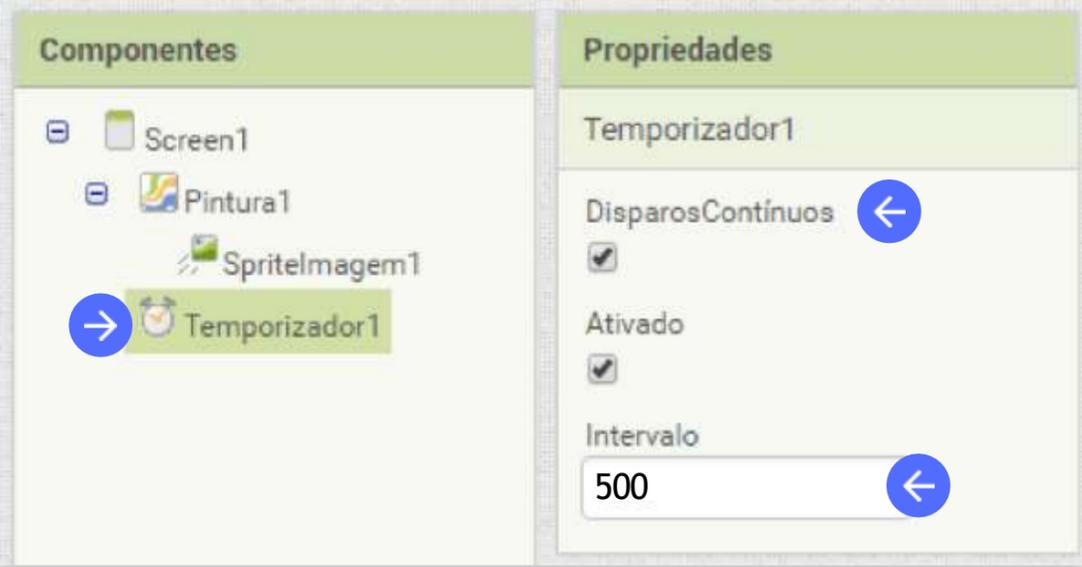


# MOVER O MOSQUITO PELA TELA

Selecione o componente **Temporizador1** e defina as suas propriedades.

Com a opção **“DisparosContínuos”** o Temporizador vai ficar disparando repetidamente a cada Intervalo de tempo.

O **“Intervalo”** é o tempo em milissegundos que demora para o comando disparar, como queremos 0,5 segundo, deixamos como 500 milissegundos.



Componentes	Propriedades
Screen1	Temporizador1
Pintura1	DisparosContínuos <input checked="" type="checkbox"/>
Spritelimagem1	Ativado <input checked="" type="checkbox"/>
Temporizador1	Intervalo: 500

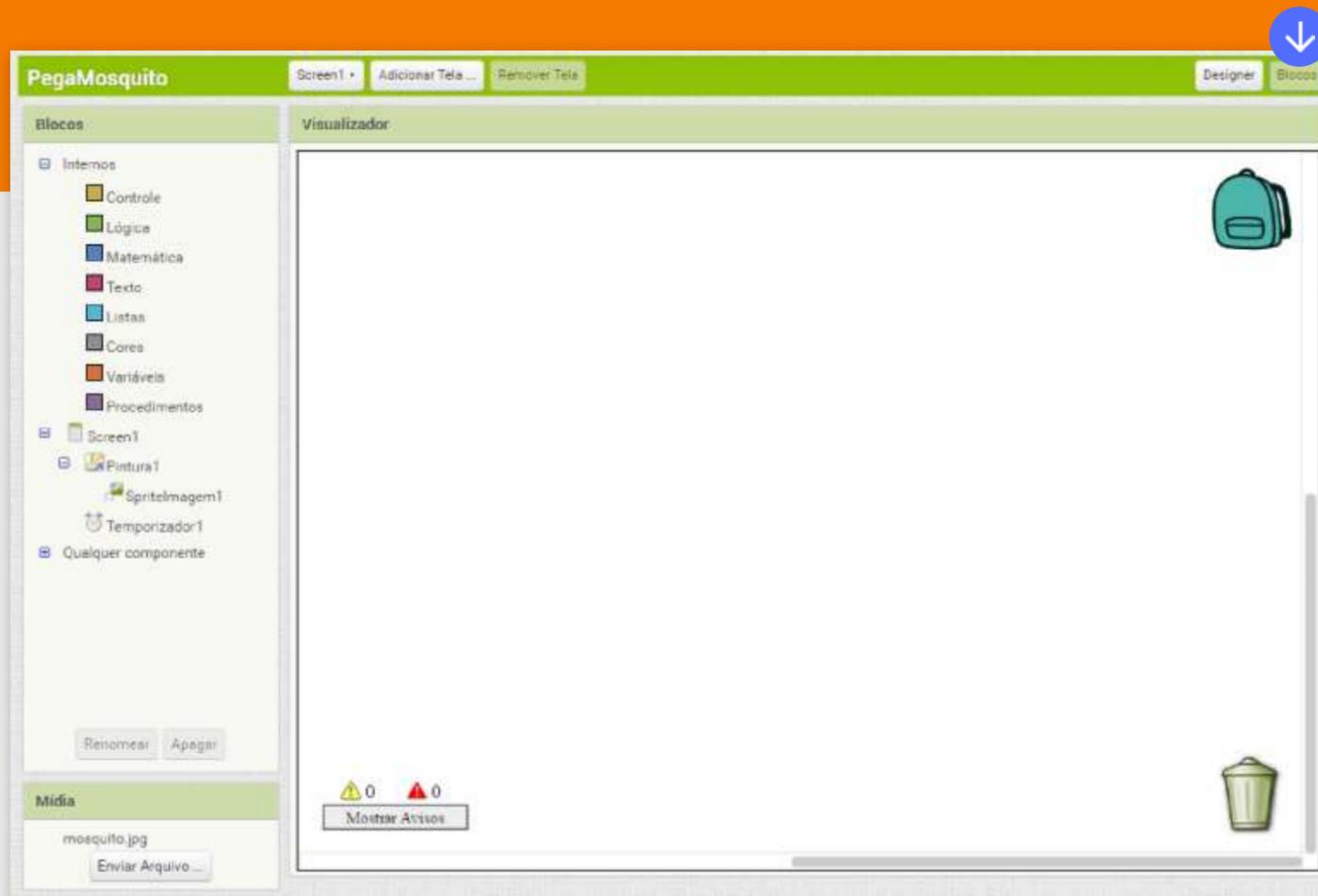
## DICA

Sempre que falarmos em selecionar o componente para ver suas propriedades, significa clicar no nome do componente na coluna de “Componentes”

# MOVER O MOSQUITO PELA TELA

Agora vamos programar o movimento do mosquito.

Até então estávamos trabalhando na tela de Design do nosso aplicativo, agora vamos passar para a tela de programação, ou **Blocos**.



# MOVER O MOSQUITO PELA TELA

Para mover o mosquito na tela, vamos criar um **Procedimento**, ou seja, um conjunto de comandos que vamos chamar de **MoverMosquito**.

Clique em **Procedimentos** na coluna de Blocos.

The image shows a screenshot of the Scratch programming environment. The interface is divided into two main panels: 'Blocos' (Blocks) on the left and 'Visualizador' (Viewer) on the right.

**Blocos (Blocks):** This panel contains a list of block categories. Under the 'Internos' (Internal) category, the following blocks are listed: Controle (Control), Lógica (Logic), Matemática (Mathematics), Texto (Text), Listas (Lists), Cores (Colors), Variáveis (Variables), and **Procedimentos** (Procedures). The 'Procedimentos' block is highlighted with a green background and a blue arrow pointing to it from the right. Below this, there are other categories: Screen1, Pintura1 (Paint) which includes Spritemagem1 (Sprite Image), and Temporizador1 (Timer). At the bottom, there is a 'Qualquer componente' (Any Component) category.

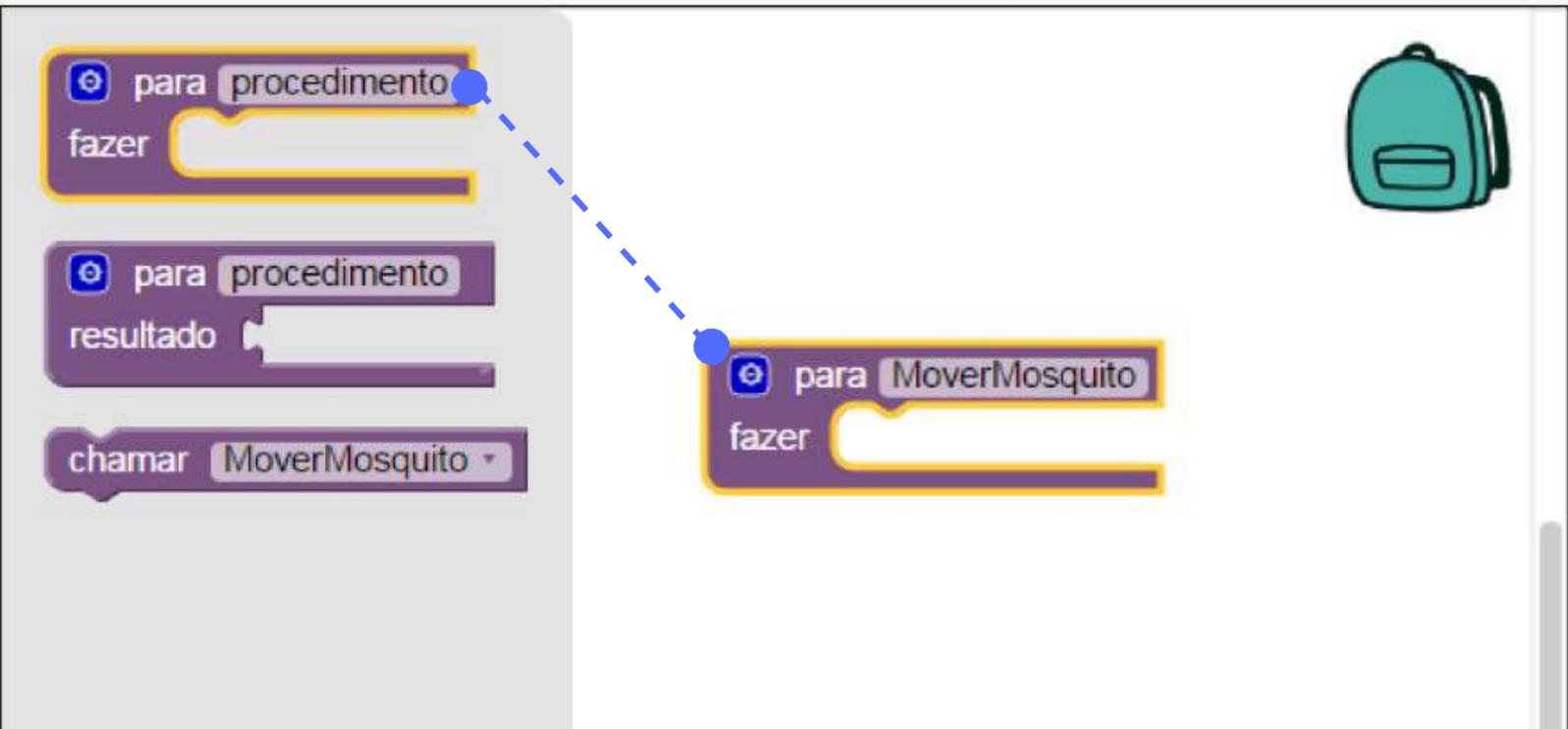
**Visualizador (Viewer):** This panel shows the visual representation of the code blocks. It contains two 'para procedimento' (for procedure) blocks. The first block has a 'fazer' (do) block attached to it. The second block has a 'resultado' (result) block attached to it.

# MOVER O MOSQUITO PELA TELA

Arraste o comando “**para procedimento fazer**” até a área do Visualizador.

Clique na palavra “**procedimento**” e substitua por “**MoverMosquito**”.

Visualizador



The screenshot shows a visual programming environment. On the left, a palette titled 'Visualizador' contains three blocks: a 'para procedimento fazer' block, a 'para procedimento resultado' block, and a 'chamar MoverMosquito' block. On the right, a workspace contains a 'para MoverMosquito fazer' block. A dashed blue line connects the 'procedimento' text in the top-left block to the 'procedimento' text in the workspace block. A green backpack icon is in the top-right corner of the workspace.

# MOVER O MOSQUITO PELA TELA

Esse procedimento deve trocar a posição do Mosquito toda vez que for ativado.

Para fazer isto, vamos utilizar a função de trocar a posição **X** e **Y** do componente **Spritemagem**, ou seja, do “Mosquito”.

Clique no **Mosquito** na coluna de Blocos.

The screenshot shows the Scratch IDE interface for a project named "PegaMosquito". The "Blocos" (Blocks) panel on the left lists various categories, with "Mosquito" selected under "Screen1". The "Visualizador" (Visualizer) panel on the right displays the script area with four event blocks for the Mosquito component:

- quando Mosquito .ColidiuCom** (when Mosquito collides with):
  - outro (other)
  - fazer (do)
- quando Mosquito .Arrastado** (when Mosquito is dragged):
  - xlInicial, ylInicial, xAnterior, yAnterior, xAtual, yAtual (variables)
  - fazer (do)
- quando Mosquito .AlcançouBorda** (when Mosquito reaches the edge):
  - borda (edge)
  - fazer (do)
- quando Mosquito .Arremessado** (when Mosquito is thrown):
  - x, y, velocidade, direção, xvel, yvel (variables)
  - fazer (do)
- quando Mosquito .ToqueParaBaixo** (when Mosquito is touched from below):
  - x, y (variables)
  - fazer (do)

The "Visualizador" panel also shows a preview of the Mosquito component and a trash icon.

# MOVER O MOSQUITO PELA TELA

Arraste o comando “ajustar Mosquito.X para” para dentro do procedimento “MoverMosquito”.

## DICA

Como a lista de comandos do Mosquito é muito grande, você terá que descer a barra de rolagem até o fim para achar o comando utilizado.

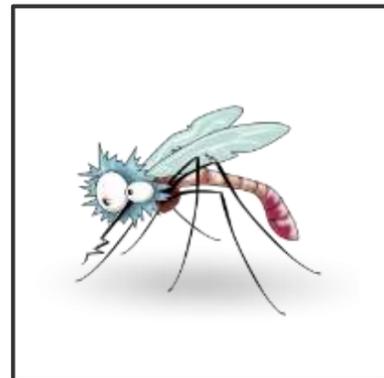
The screenshot shows the Scratch IDE interface for a project named 'PegaMosquito'. The 'Visualizador' pane displays a script for the 'MoverMosquito' procedure. The script consists of several blocks: 'ajustar Mosquito . Visível para', 'Mosquito . Largura', 'ajustar Mosquito . Largura para', 'Mosquito . X', 'ajustar Mosquito . X para', 'Mosquito . Y', 'ajustar Mosquito . Y para', 'Mosquito . Z', 'ajustar Mosquito . Z para', and 'Mosquito'. A blue dashed line indicates the 'ajustar Mosquito.X para' block being dragged from the bottom of the list into the 'para MoverMosquito' loop. The 'Blocos' pane on the left shows a list of categories, including 'Internos', 'Controle', 'Lógica', 'Matemática', 'Texto', 'Listas', 'Cores', 'Variáveis', and 'Procedimentos'. The 'Screen1' and 'Pintura1' folders are expanded, showing the 'Mosquito' component.

# MOVER O MOSQUITO PELA TELA

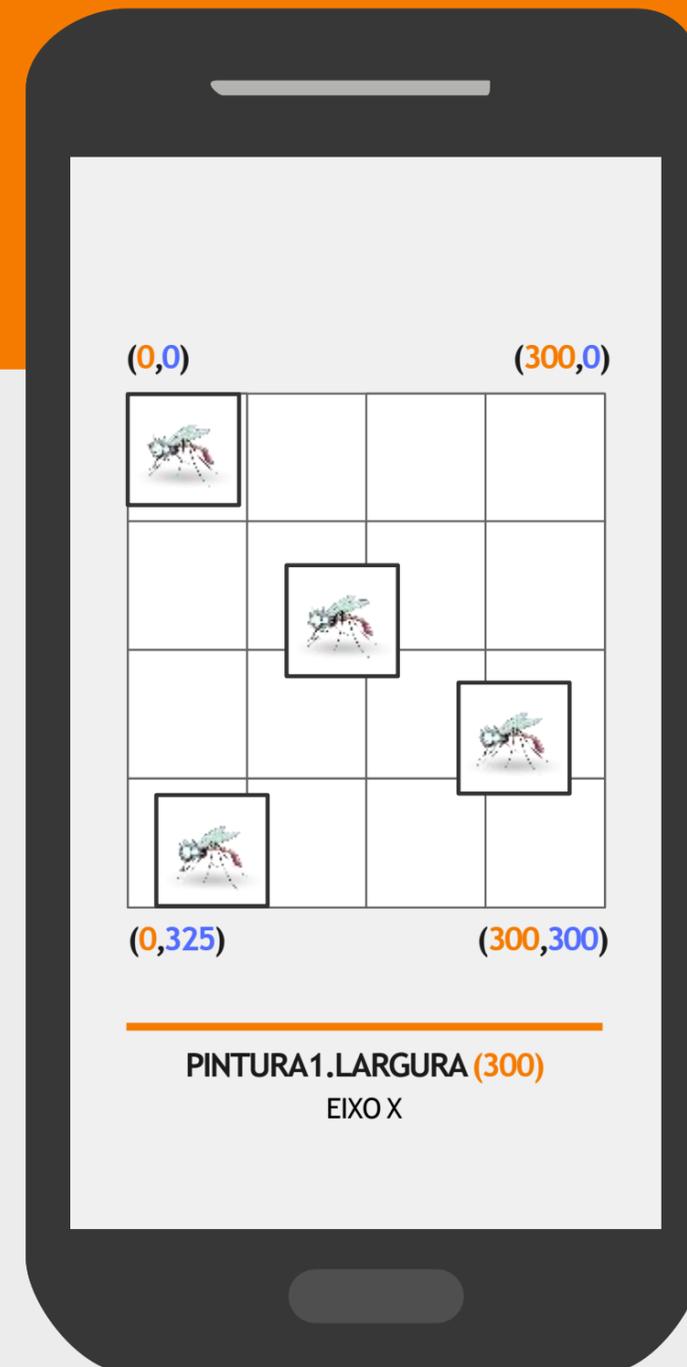
Para que posição queremos mover o mosquito?

A posição deve ficar entre **0** (zero) e **270** (tamanho da tela menos o tamanho do mosquito)

MOSQUITO.ALATURA (30)



MOSQUITO.LARGURA (30)



# MOVER O MOSQUITO PELA TELA

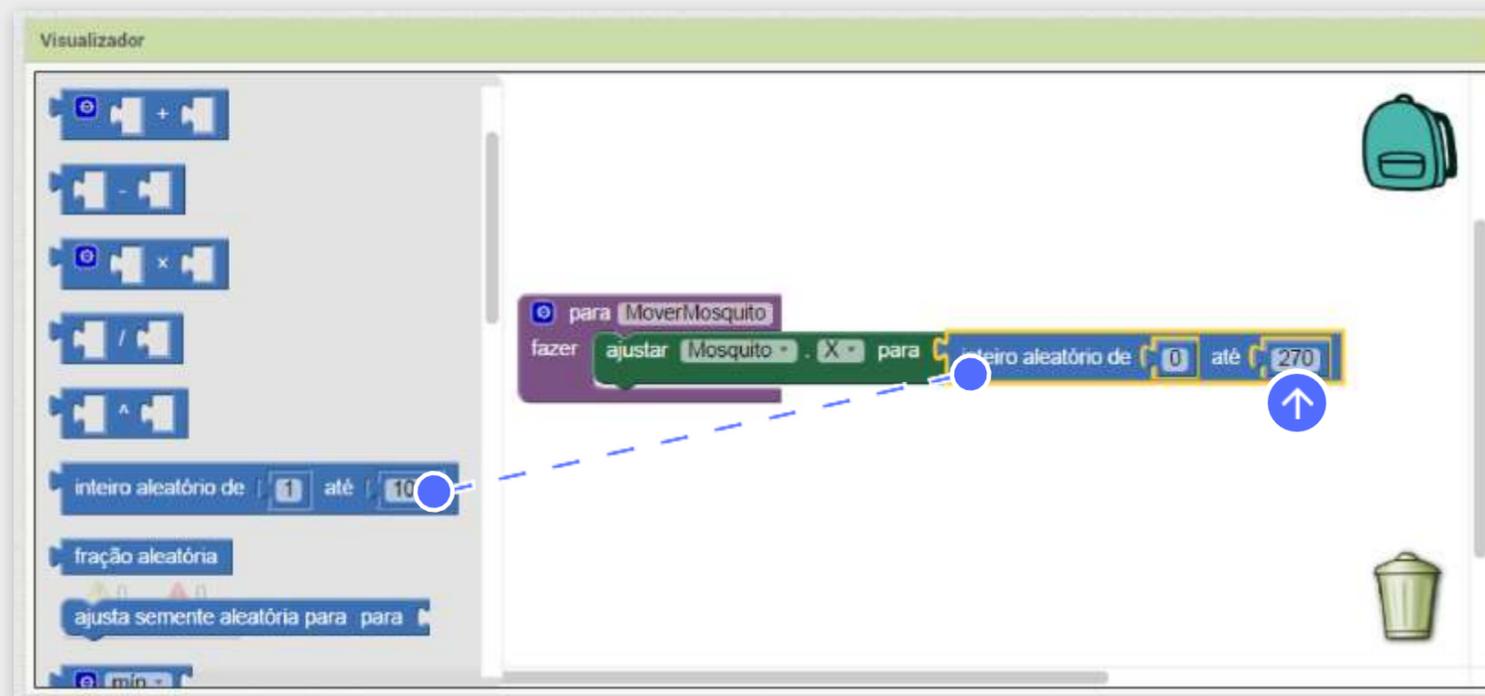
Como queremos que o Mosquito se mova para uma posição “sorteada”, vamos usar o bloco matemático “inteiro aleatório de 1 até 100”.



The image shows a programming environment with two main panels: 'Blocos' (Blocks) on the left and 'Visualizador' (Viewer) on the right. The 'Blocos' panel is organized into categories: Internos (Internal), Controle (Control), Lógica (Logic), Matemática (Mathematics), Texto (Text), Listas (Lists), Cores (Colors), Variáveis (Variables), and Procedimentos (Procedures). Under 'Internos', there are sub-categories: Screen1, Pintura1 (Paint), Mosquito, and Temporizador1 (Timer). The 'Visualizador' panel shows a script for the Mosquito object. The script starts with a 'quando clicado' (when clicked) event block, followed by a 'mover + para Mosquito' (move + to Mosquito) block. Below this is a 'fazer ajustar Mosquito X para' (do adjust Mosquito X to) block. The 'inteiro aleatório de 1 até 100' (random integer from 1 to 100) block is highlighted with a yellow border, indicating it is the block being used to generate a random position for the mosquito.

# MOVER O MOSQUITO PELA TELA

Arraste o bloco de **inteiro aleatório** para completar o comando de **ajustar Mosquito.X** e preencha com os valores que queremos (0 e 270).



# MOVER O MOSQUITO PELA TELA

Já criamos nosso procedimento para mover o mosquito.

Agora precisamos dizer **quando** ele deve se mover, isto é, quando o **Temporizador1** disparar.

Vamos usar o bloco relacionado ao evento de Disparo do **Temporizador1**.

The image shows a block editor interface with two main panels: 'Blocos' (Blocks) on the left and 'Visualizador' (Viewer) on the right.

**Blocos Panel:** A tree view of categories. Under 'Internos', there are sub-categories: Controle, Lógica, Matemática, Texto, Listas, Cores, Variáveis, and Procedimentos. Under 'Qualquer componente', there are 'Screen1', 'Pintura1', 'Mosquito', and 'Temporizador1' (highlighted).

**Visualizador Panel:** A script area containing a sequence of blocks:

- A yellow 'quando Temporizador1 .Disparo' block with a 'fazer' loop.
- A purple 'chamar Temporizador1 .Add Days' block with 'instante' and 'quantity' options.
- A purple 'chamar Temporizador1 .Add Duration' block with 'instante' and 'quantity' options.
- A purple 'chamar Temporizador1 .Add Hours' block with 'instante' and 'quantity' options.
- A purple 'chamar Temporizador1 .Add Minutes' block with 'instante' and 'quantity' options.

# MOVER O MOSQUITO PELA TELA

## EVENTO

Um evento é algo que acontece, por exemplo, quando pressionamos um botão. O despertar de um alarme também pode ser considerado um evento.



# MOVER O MOSQUITO PELA TELA

A cada 0,5 segundos serão executadas as instruções que estão dentro do bloco de Disparo do **Temporizador1**.

Clique em “Procedimentos”, e arraste o bloco “**chamar MoverMosquito**” para dentro do bloco de Disparo.

The image shows the Scratch IDE interface. On the left is the 'Blocos' (Blocks) panel, and on the right is the 'Visualizador' (Viewer) panel.

**Blocos Panel:**

- Internos
  - Controle
  - Lógica
  - Matemática
  - Texto
  - Listas
  - Cores
  - Variáveis
  - Procedimentos** (highlighted)
- Screen1
- Pintura1
  - Mosquito
- Temporizador1
- Qualquer componente

**Visualizador Panel:**

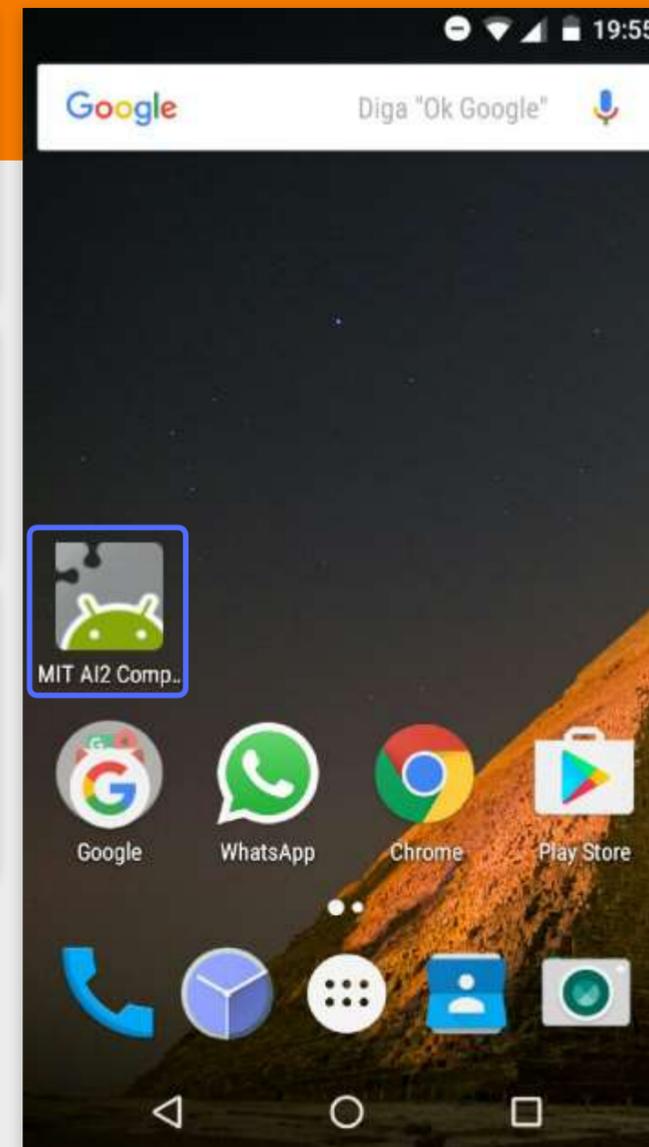
- para procedimento (loop block)
- fazer (do block)
- ajustar Mosquito . X para (set block)
- quando Temporizador1 .Disparo (when timer ticks block)
- fazer chamar MoverMosquito (call block)

# MOVER O MOSQUITO PELA TELA

Pronto! Agora nosso mosquito deve estar se movendo sozinho na tela.

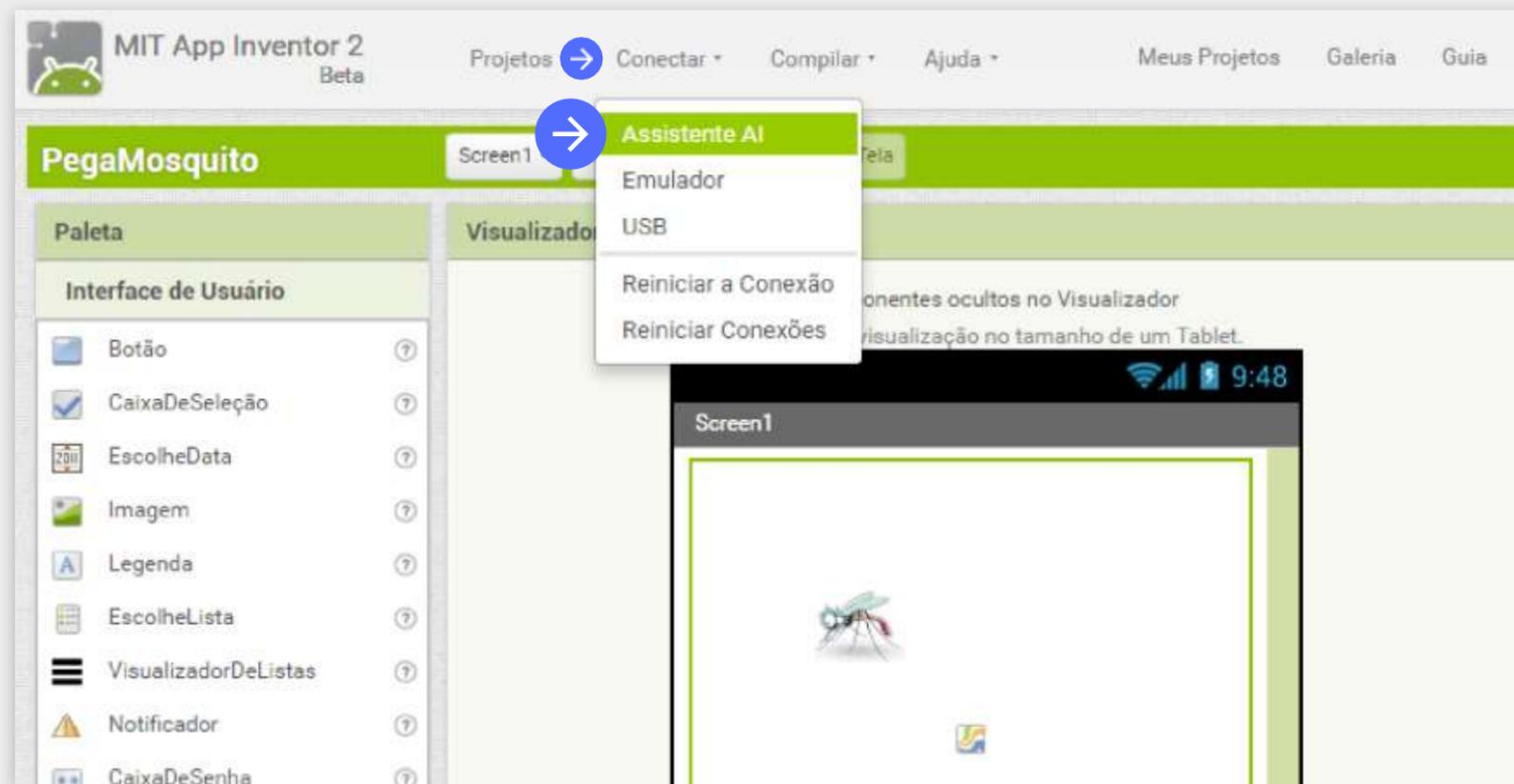
Vamos testar no celular como está ficando o jogo!

Pegue o seu celular e abra o aplicativo **“MIT AI2 Companion”**.



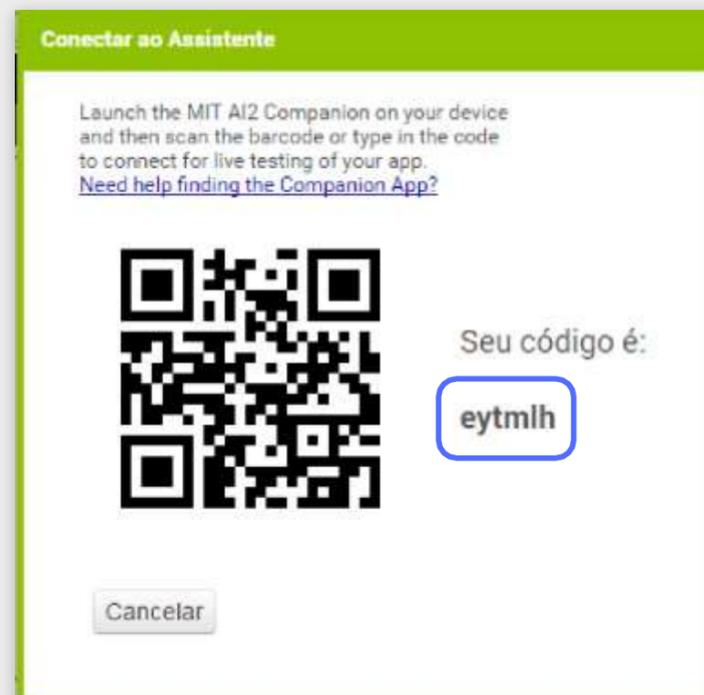
# MOVER O MOSQUITO PELATELA

No AppInventor clique em “Conectar” -> “Assistente AI”.



# MOVER O MOSQUITO PELA TELA

No aplicativo do celular clique em “scan QR code” e aponte a câmera para o símbolo da tela do computador. Pronto, o seu celular está conectado com o programa que temos no computador.



## MIT App Inventor 2

type in the 6-character code  
-or-  
scan the QR code

Six Character Code

connect with code

scan QR code

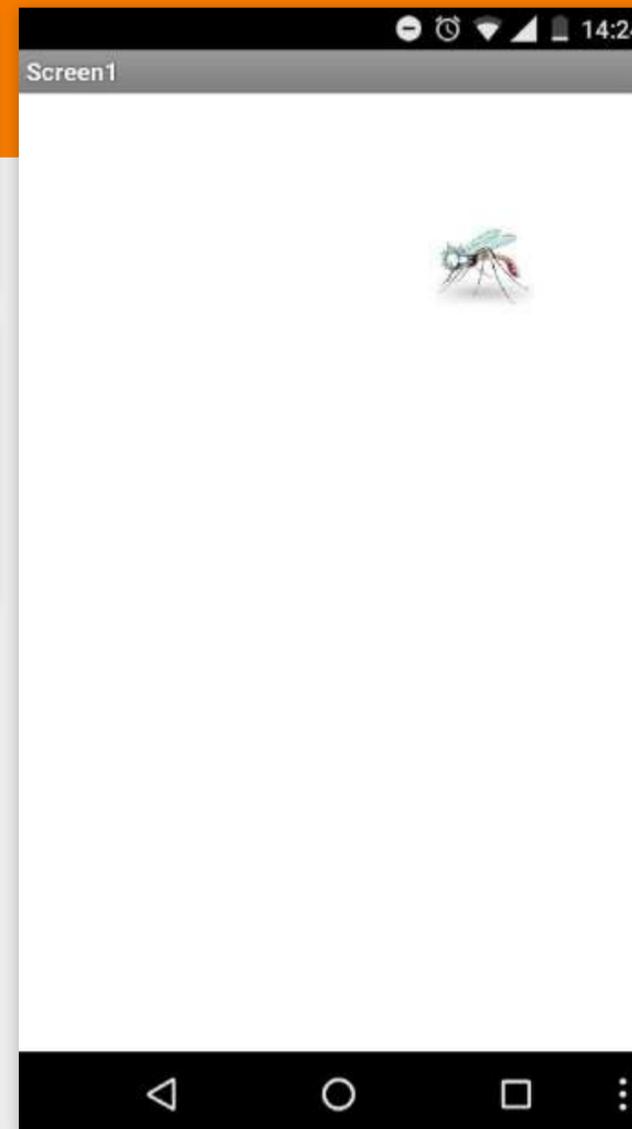
## DICA

Se o seu celular não tiver câmera para usar o “scan”, você pode digitar o código indicado e clicar em “connect with code”.

# MOVER O MOSQUITO PELA TELA

Você deve notar que o mosquito só está se movendo para os lados.

Temos que fazê-lo se movimentar em todas as direções, X e Y.

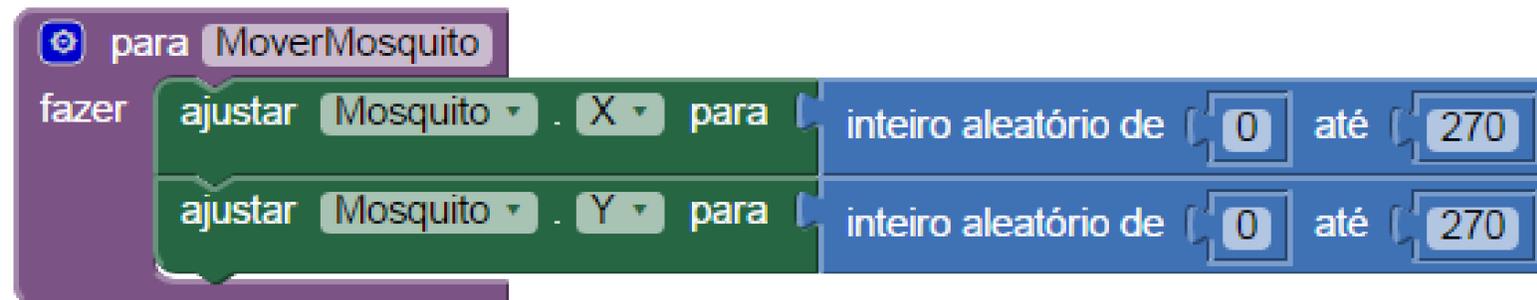


# MOVER O MOSQUITO PELA TELA

Vamos seguir os mesmos passos anteriores para alterar também a coordenada Y do **Mosquito**.

O procedimento **MoverMosquito** deve ficar como a imagem ao lado.

Note como o jogo é atualizado automaticamente no celular!



```
para MoverMosquito
  fazer
    ajustar Mosquito . X para inteiro aleatório de 0 até 270
    ajustar Mosquito . Y para inteiro aleatório de 0 até 270
```

The image shows a Scratch code block for a function named 'MoverMosquito'. It is a 'para' block with a 'fazer' loop. Inside the loop, there are two 'ajustar' blocks. The first block is 'ajustar Mosquito . X para inteiro aleatório de 0 até 270'. The second block is 'ajustar Mosquito . Y para inteiro aleatório de 0 até 270'.

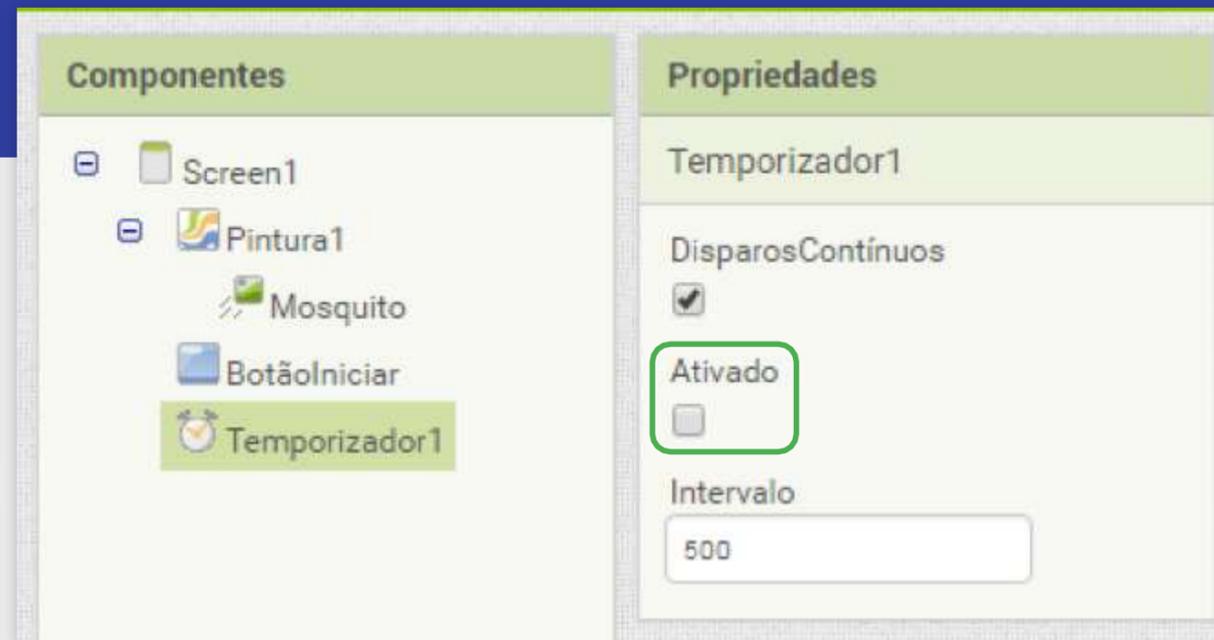
# INICIAR O JOGO



# INICIAR O JOGO

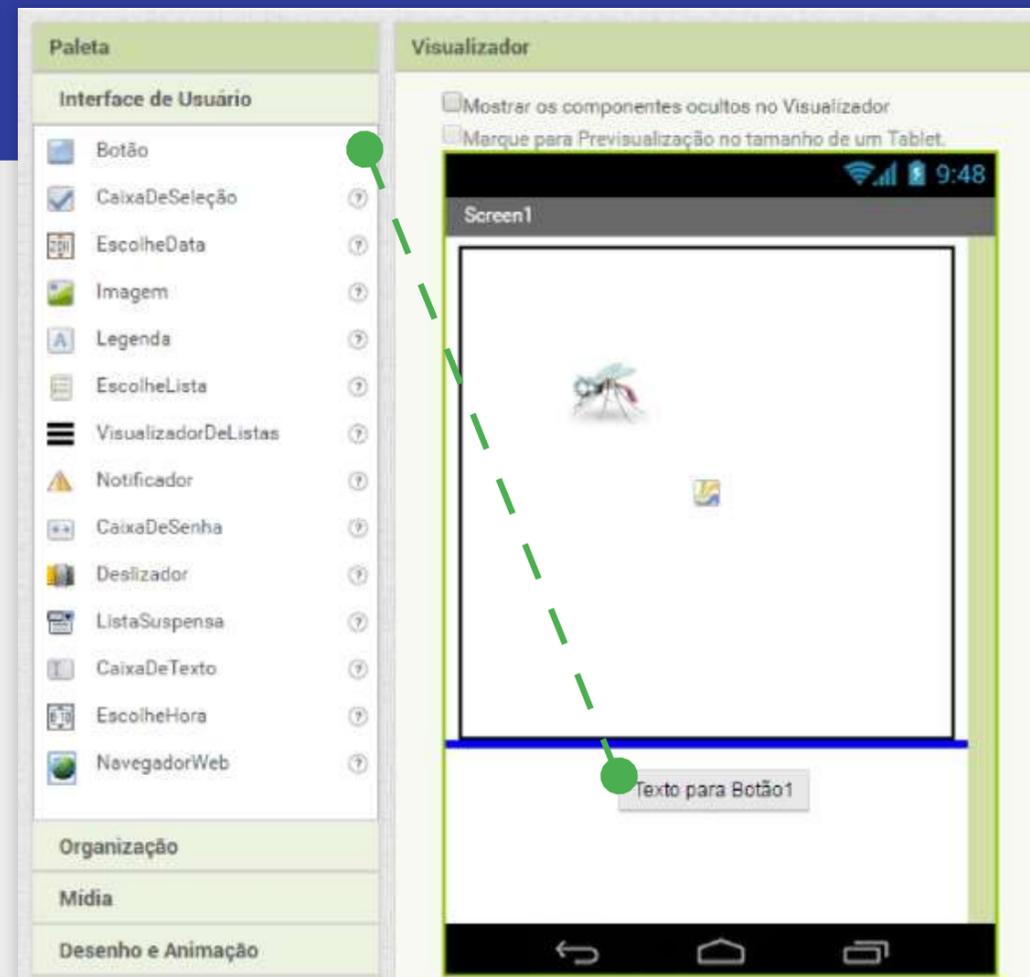
Primeiro, vamos voltar para a tela de “Designer” e desativar o **Temporizador1**. Assim, ele não vai mais disparar o movimento do mosquito até que o botão **Iniciar Jogo** seja clicado.

O mosquito está se movimentando a cada disparo do **Temporizador1**. Contudo precisamos que isso seja realizado somente quando formos jogar. Para isto, vamos inserir um botão de **Iniciar Jogo**.



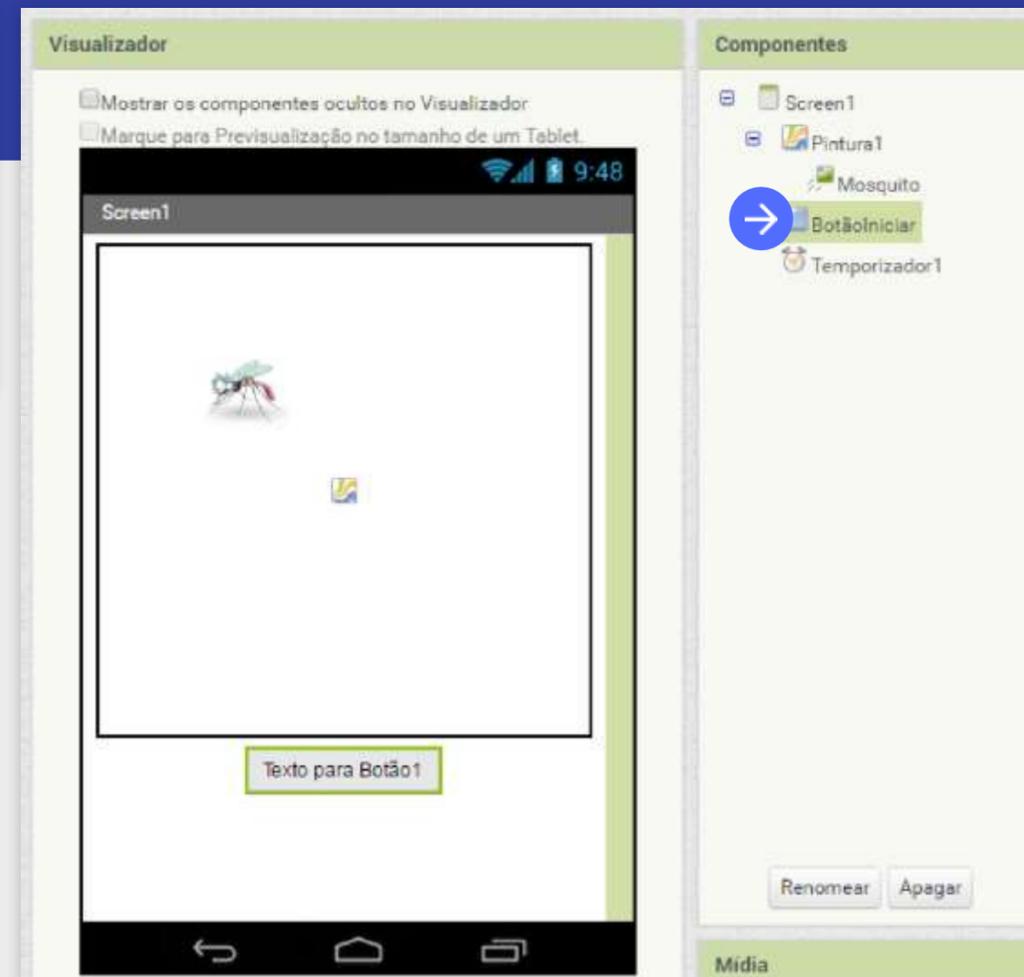
# INICIAR O JOGO

Na paleta de “Interface de Usuário”, arraste o **Botão** para a tela, preferencialmente embaixo da área da **Pintura1**.



# INICIAR O JOGO

Agora, da mesma maneira que renomeamos o **Mosquito**, vamos renomear o componente **-Botao1** para **Botaoiniciar**.



# INICIAR O JOGO

Também devemos trocar a propriedade do **BotãoIniciar** para ele apresentar o texto “Iniciar Jogo”.

Veja no seu celular se o botão está aparecendo corretamente!

The image shows a software development interface with three main panels: Visualizador, Componentes, and Propriedades.

- Visualizador:** Displays a mobile app preview. At the bottom, a button labeled "Iniciar Jogo" is highlighted with a blue circle and an arrow. Below the preview, there are labels for "Componentes invisíveis" and "Temporizador1".
- Componentes:** Lists the app's components: "Screen1", "Pintura1", "Mosquito", "BotãoIniciar" (highlighted with a green box), and "Temporizador1". Buttons for "Renomear" and "Apagar" are visible.
- Propriedades:** Shows the properties for the selected "BotãoIniciar" component. The "Texto" property is set to "Iniciar Jogo" and is highlighted with a blue circle and an arrow. Other properties include "CorDeFundo", "Ativado", "FonteNegrito", "FonteItálico", "TamanhoDaFonte", "FamiliaDaFonte", "Altura", "Largura", "Imagem", "Forma", and "MostrarFeedback".

# INICIAR O JOGO

Você deve ter verificado que ao clicar no botão nada está acontecendo, por isto, precisamos programar a funcionalidade de iniciar o jogo.

Volte para a tela de **Blocos**.

Vamos adicionar o bloco **quando Botaoiniciar.clique**. Este bloco se encontra nos blocos referentes ao **BotãoIniciar**.

The screenshot displays the Scratch programming environment. On the left is the 'Blocs' panel, which is organized into categories: Internos (Control, Logic, Math, Text, Lists, Colors, Variables, Procedures), Screen1 (Paint1, Mosquito, BotãoIniciar, Temporizador1), and Qualquer componente. The 'BotãoIniciar' block is highlighted in green. On the right is the 'Visualizador' panel, which shows a script for the 'BotãoIniciar' object. The script contains several event blocks: 'quando BotãoIniciar .Clique', 'quando BotãoIniciar .RecebeuFoco', 'quando BotãoIniciar .CliqueLongo', 'quando BotãoIniciar .PerdeuFoco', 'quando BotãoIniciar .ToqueParaBaixo', and 'quando BotãoIniciar .ToqueParaCima'. Each event block is followed by a 'fazer' block. The 'quando BotãoIniciar .CliqueLongo' block has a 'fazer' block containing 'ajustar [aleatório] para inteiro aleatório'. The 'quando BotãoIniciar .Clique' block has a 'fazer' block containing 'BotãoIniciar .CorDeFundo'.

# INICIAR O JOGO

Quando o **BotãoIniciar** for clicado vamos ativar o **Temporizador1** para o jogo começar.

Clique em **Temporizador1** e escolha o bloco ajustar **Temporizador1.Ativado**, como mostrado ao lado.

The image shows the Scratch IDE interface. On the left, the 'Blocos' (Blocks) panel is open, showing a tree view of the project's components. Under 'Internos', there are categories like 'Controle', 'Lógica', 'Matemática', 'Texto', 'Listas', 'Cores', 'Variáveis', and 'Procedimentos'. Under 'Screen1', there are 'Pintura1', 'Mosquito', and 'BotãoIniciar'. The 'Temporizador1' block is highlighted with a blue arrow. On the right, the 'Visualizador' (Scripts) panel shows the script for the 'BotãoIniciar' button click event. The script consists of a 'quando BotãoIniciar .Clique' event block followed by a 'fazer' block containing an 'ajustar Temporizador1 . Ativado para inteiro aleatório' block. A blue arrow points to the 'ajustar' block in the script.

# INICIAR O JOGO

Lembra quando desativamos o **Temporizador1** na tela de “Designer”?

A propriedade “Ativado” do **Temporizador1** pode ter dois valores, Verdadeiro, ou Falso.

**Blocs**

- Internos
  - Controle
  - Lógica**
  - Matemática
  - Texto
  - Listas
  - Cores
  - Variáveis
  - Procedimentos
- Screen1
- Qualquer componente

**Visualizador**

- verdadeiro
- falso
- não
- =
- e
- ou

ajustar Temporizador1 . Intervalo para 500

TEMPO ENTRE DISPAROS É DE 500 MILISEGUNDOS

ajustar Temporizador1 . Ativado para verdadeiro

SE A VARIÁVEL ATIVADO TIVER VALOR VERDADEIRO, VAIDISPARAR!

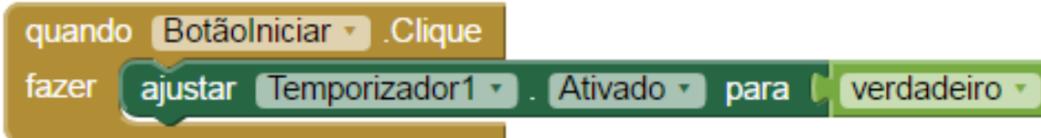
ajustar Temporizador1 . Ativado para falso

SE A VARIÁVEL ATIVADO TIVER VALOR FALSO, NÃO VAIDISPARAR!

# INICIAR O JOGO

Para que o nosso botão ative o **Temporizador1**, selecione o bloco de **Lógica** com ovalor **Verdadeiro**.

O comando do **BotãoIniciar** deve ficar assim:



```
quando BotãoIniciar .Clique  
fazer ajustar Temporizador1 . Ativado para verdadeiro
```

# INICIAR O JOGO

Teste o seu aplicativo e verifique que o mosquito se move após apertar no **Botão Iniciar!**



# VIDAS DO MOSQUITO

Agora vamos adicionar vida ao Mosquito.  
Ele vai iniciar o jogo com 3 vidas.

Cada vez que o(a) jogador(a) conseguir clicar no mosquito devemos diminuir a sua vida em 1. Ou seja, quando o(a) jogador(a) clicar 3 vezes sobre o mosquito ele(a) ganha o jogo.

# VIDAS DO MOSQUITO

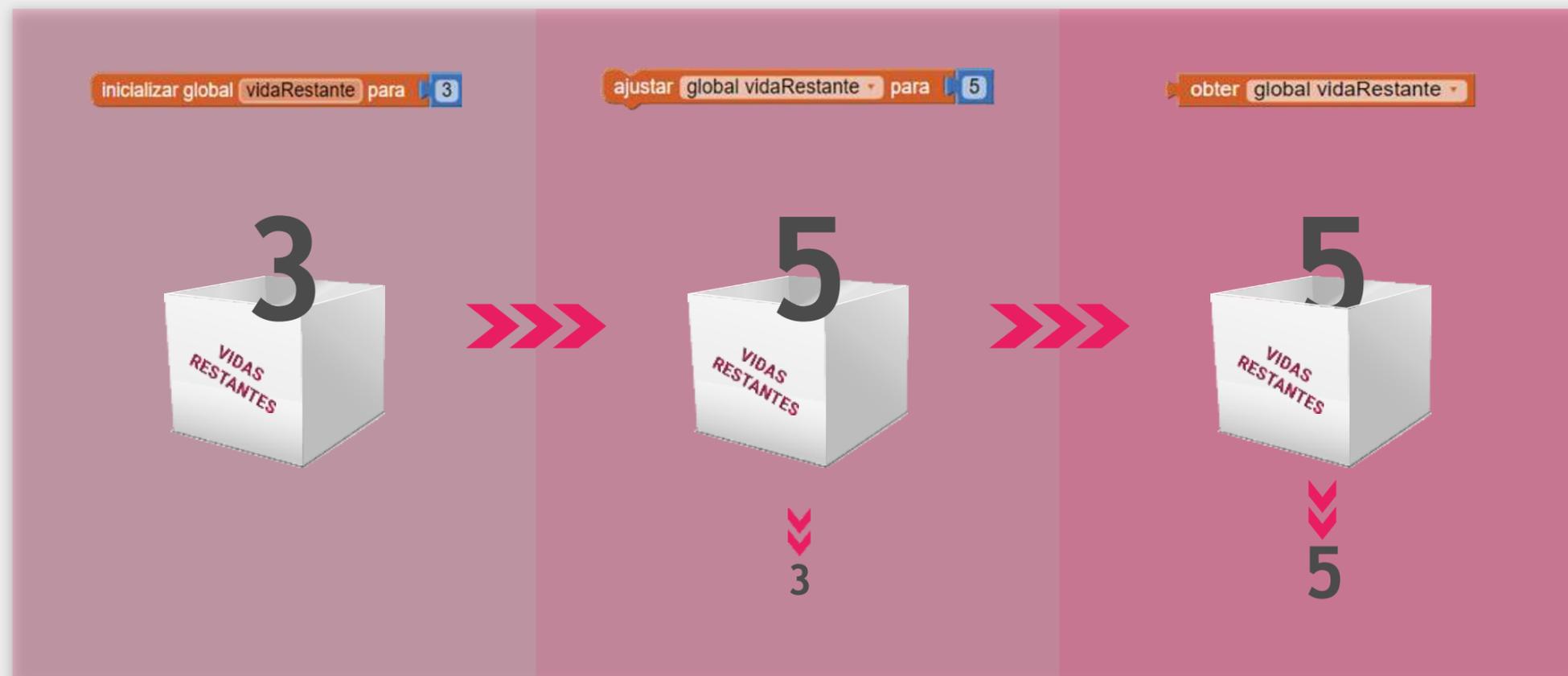
Precisamos criar um bloco que armazene o quanto de vida o mosquito possui. Esses blocos que armazenam valores são chamados de **variáveis**.

Vamos utilizar uma variável global que será chamada de “**vidaRestante**”. Usando o bloco “**inicializar global nome para**” e atribuir o valor “3”.

The image shows a software interface with two main panels: 'Blocos' (Blocks) on the left and 'Visualizador' (Stage) on the right. The 'Blocos' panel is organized into categories: Internos (Internal), Screen 1, Pintura 1 (Paint 1), and Qualquer componente (Any component). Under 'Internos', there are sub-categories: Controle (Control), Lógica (Logic), Matemática (Mathematics), Texto (Text), Listas (Lists), Cores (Colors), Variáveis (Variables), and Procedimentos (Procedures). The 'Variáveis' category is highlighted in green. Under 'Pintura 1', there are 'Mosquito' (Mosquito), 'BotãoIniciar' (Start Button), and 'Temporizador1' (Timer1). The 'Visualizador' panel shows a script with the following blocks: 1. 'inicializar global nome para' (Initialize global variable named 'nome' to 'para'). 2. 'obter' (Get). 3. 'ajustar para' (Adjust to 'para'). 4. 'inicializar local nome para dentro de' (Initialize local variable named 'nome' to 'para' within 'dentro de'). 5. Another 'inicializar local nome para dentro de' block.

# VIDAS DO MOSQUITO

**Variável** é o nome dado ao local onde você pode armazenar informações e são utilizadas para lembrar de coisas como: a pontuação, o nome de um jogador ou até a velocidade do personagem.



# VIDAS DO MOSQUITO

O bloco de **iniciar variável global** não precisa ser inserido dentro de outro bloco de função. Ele é executado automaticamente toda a vez que o aplicativo for aberto (inicializado).

Sendo assim, basta arrastarmos ele para a área do Visualizador e colocar o valor **Matemático** “3” junto a ele.

The screenshot displays the Scratch IDE interface. On the left, the 'Blocos' (Blocks) palette is visible, with the 'Matemática' (Math) category selected. On the right, the 'Visualizador' (Stage) area shows a script. The script starts with an 'iniciar global' block for the variable 'vidaRestante' set to the value 3. Below this, there is a procedure block named 'MoverMosquito' which contains two 'ajustar' blocks for the 'Mosquito' object, one for the X coordinate and one for the Y coordinate. The 'ajustar' blocks are set to 'inteiro al' (integer to).

# VIDAS DO MOSQUITO

Agora, toda vez que clicarmos no **Mosquito** devemos diminuir a variável “**vidaRestante**” em “1”. Para isto, adicionamos o bloco quando **mosquito.Tocou** pertencente aos blocos do componente **Mosquito**.

Esse bloco será executado toda vez que a imagem do **Mosquito** for tocada.

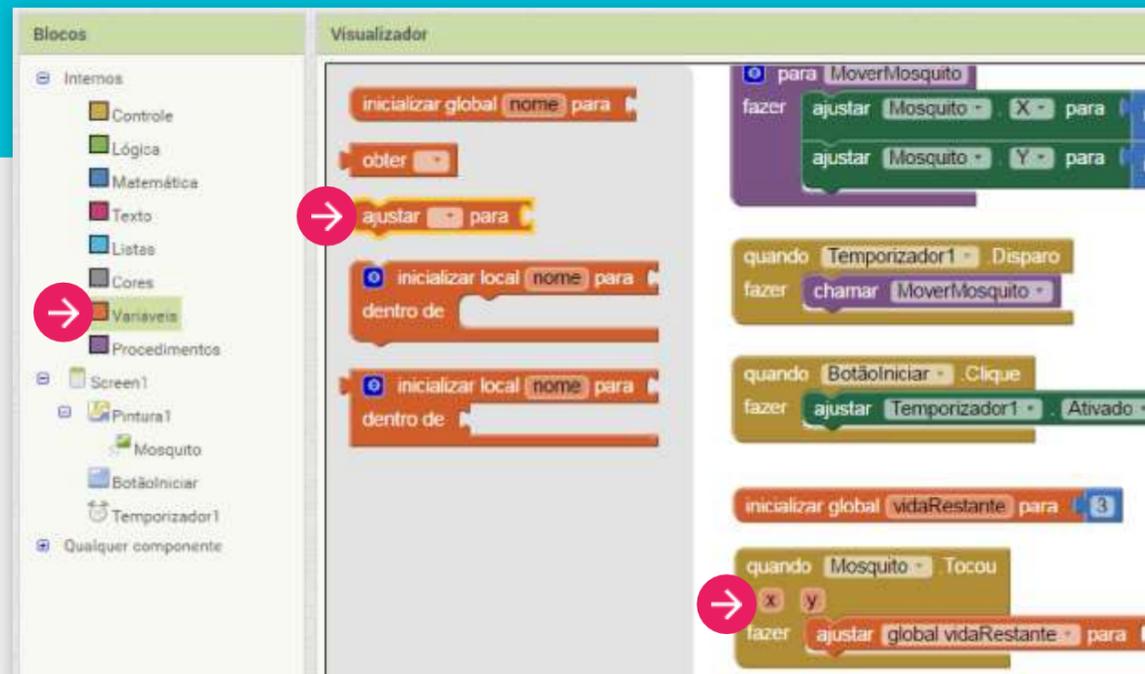
The image shows the Scratch IDE interface. On the left, the 'Blocos' (Blocks) panel is visible, showing a tree view of components. The 'Mosquito' component is selected. On the right, the 'Visualizador' (Inspector) panel shows the script area. There are three event blocks: 'quando Mosquito .ToqueParaBaixo', 'quando Mosquito .ToqueParaCima', and 'quando Mosquito .Tocou'. The 'quando Mosquito .Tocou' block is highlighted with a red arrow, and its 'fazer' field is empty. Below it is a 'chamar Mosquito .Quicar' block.

# VIDAS DO MOSQUITO

Vamos usar o bloco “ajustar ... para” que ficamos blocos de “Variáveis”.

Coloque o bloco dentro do procedimento “quando Mosquito.tocou”.

Escolha a variável “global vidaRestante” na lista 



The screenshot shows the Scratch IDE interface. On the left, the 'Blocos' (Blocks) panel is open, with the 'Variáveis' (Variables) category selected. In the center, the 'Visualizador' (Inspector) window shows the script for the 'Mosquito' character. The script includes an 'ajustar ... para' block highlighted with a red arrow. Below it, there is a 'quando Mosquito.tocou' event block with an 'ajustar global vidaRestante' block highlighted with a red arrow. The 'ajustar global vidaRestante' block has a dropdown menu open, showing the variable 'global vidaRestante' selected.

# VIDAS DO MOSQUITO

O novo valor vai ser o valor atual da vidaRestante - 1.

Para isso, escolha o bloco Matemático de Subtração.



# VIDAS DO MOSQUITO

O primeiro valor da subtração é o valor da variável vidaRestante.

Selecione o bloco “**obter ...**” e escolha “global vidaRestante” na lista.

O segundo valor é apenas o bloco Matemático de número (mude de 0 para 1).

O procedimento vai ficar assim:

The image shows the Scratch interface. On the left, the 'Blocos' panel is open to the 'Variáveis' category. On the right, the 'Visualizador' panel shows a script with three blocks: 'inicializar global nome para', 'obter', and 'ajustar para'. A red arrow points to the 'obter' block.

```
quando Mosquito .Tocou
  fazer
    ajustar global vidaRestante para
    obter global vidaRestante - 1
```

# VIDAS DO MOSQUITO

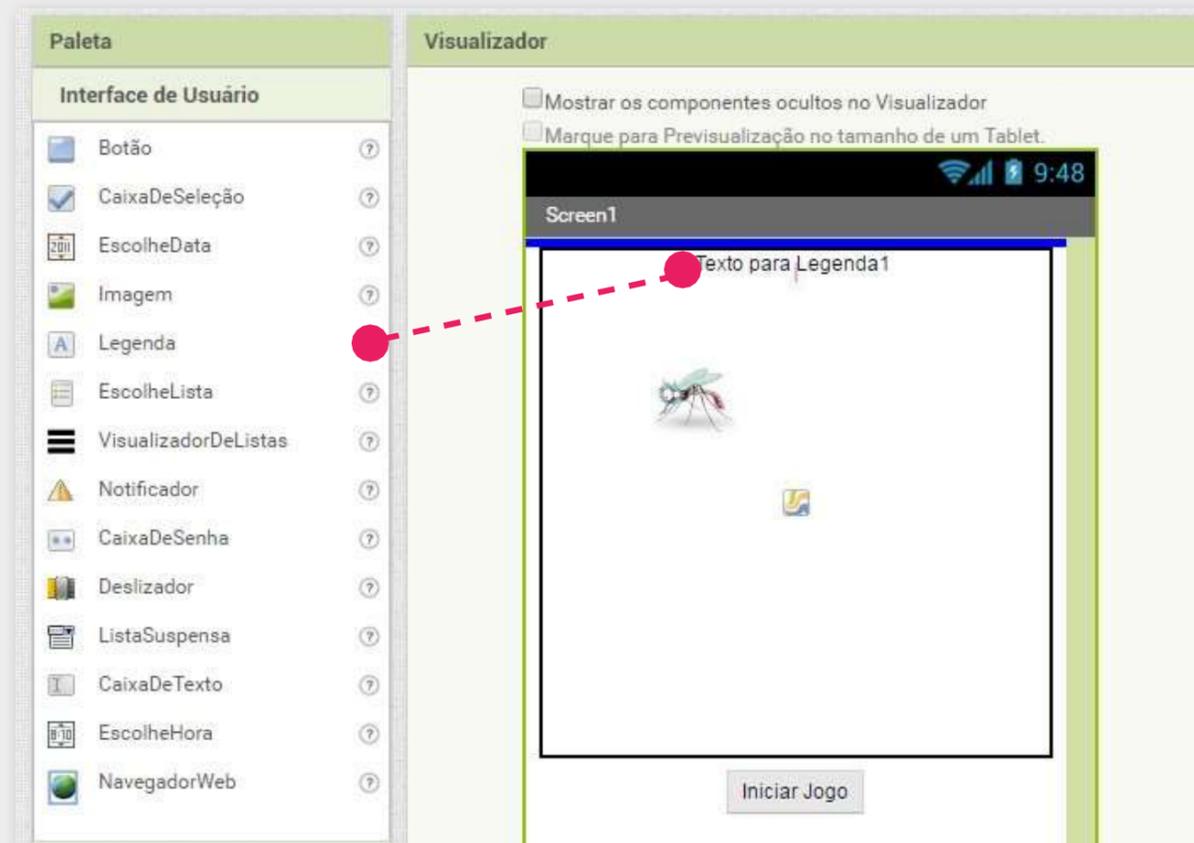
O nosso código de vida já está funcionando. Mas note que na tela do jogo não existe nada indicando para o jogador quanto é a vida do mosquito.

Precisamos mostrar o valor da vida na tela.



# VIDAS DO MOSQUITO

Na tela de “Designer”, arraste o componente **Legenda** da Paleta para a Tela, preferencialmente acima da **Pintura1**.



# VIDAS DO MOSQUITO

Teste o seu aplicativo, veja se está aparecendo corretamente o número de vidas do mosquito! Você percebeu que ao tocar no mosquito o valor da vida dele não está sendo alterado? Precisamos arrumar isto!

The image shows a screenshot of a mobile application development IDE. On the left, the 'Componentes' (Components) panel displays a tree view of the application's UI elements. The root is 'Screen1', which contains several components: 'Legenda1' (highlighted with a red arrow), 'Pintura1', 'Mosquito', 'BotãoIniciar', and 'Temporizador1'. On the right, the 'Propriedades' (Properties) panel shows the configuration for the selected 'Legenda1' component. The 'Texto' (Text) property is set to 'Vidas: 3', and a red arrow points to this text field. Other visible properties include 'CorDeFundo' (None), 'FonteNegrito' (unchecked), 'Fonteltálico' (unchecked), 'TamanhoDaFonte' (14.0), 'FamíliaDaFonte' (padrão), 'TemMargens' (checked), 'Altura' (Automático...), 'Largura' (Automático...), and 'AlinhamentoDoTexto' (esquerda). At the bottom of the components panel, there are 'Renomear' (Rename) and 'Apagar' (Delete) buttons.

# VIDAS DO MOSQUITO

Quando o mosquito for tocado, o valor da variável “vidaRestante” é diminuído, mas também devemos atualizar o texto contido na **Legenda1**.

Vamos criar um procedimento chamado de “AtualizarVidaRestante” que será responsável por atualizar o texto da **Legenda1** baseado no valor que temos na variável “vidaRestante”.

The screenshot shows a programming environment with two main panels: 'Blocos' (Blocks) on the left and 'Visualizador' (Inspector) on the right. The 'Blocos' panel has a tree view under 'Internos' (Internal) with categories: Controle (Control), Lógica (Logic), Matemática (Mathematics), Texto (Text), Listas (Lists), Cores (Colors), Variáveis (Variables), and Procedimentos (Procedures). The 'Procedimentos' category is highlighted with a red arrow. The 'Visualizador' panel shows a script for a 'Mosquito' object. The script starts with a 'quando .Tocou' (when clicked) event block. Inside this event, there is a 'fazer' (do) block containing an 'ajustar global vidaRestante' (adjust global lifeRemaining) block. Below the event block, there is a 'para' (for) loop block with the label 'AtualizarVidaRestante'. Inside this loop, there is a 'fazer' (do) block containing a 'chamar MoverMosquito' (call moveMosquito) block. A red arrow points to the 'AtualizarVidaRestante' label in the loop block. Another red arrow points to the 'para' loop block. A third red arrow points to the 'fazer' block inside the loop.

# VIDAS DO MOSQUITO

Para alterar o valor do texto escrito no componente **Legenda1** utilizamos o bloco “ajustar **Legenda1.Texto**”.

The image shows the Scratch IDE interface. On the left is the 'Blocos' (Blocks) panel, which is organized into categories: Internos (Internal), Screen1, and Qualquer componente (Any component). Under 'Internos', the 'Texto' (Text) category is selected, showing various text-related blocks. A red arrow points to the 'Legenda1' component in the 'Internos' list. On the right is the 'Visualizador' (Inspector) panel, which displays a list of blocks for the selected component, 'Legenda1'. The block 'ajustar Legenda1 . Texto' is highlighted with a yellow border and a red arrow pointing to it. Other visible blocks include 'ajustar Legenda1 . TemMargens', 'ajustar Legenda1 . Altura', 'ajustar Legenda1 . PercentualDeAltura', 'Legenda1 . Texto', 'Legenda1 . CorDeTexto', 'ajustar Legenda1 . CorDeTexto', and 'Legenda1 . Visível'.

# STRINGS

Um string é uma sequência de caracteres. O texto da Legenda1 é considerado umaString.

STRING NOME



STRING SOBRENOME



JUNTAR



# VIDAS DO MOSQUITO

O texto da **Legenda1** sempre terá a palavra "Vida: " mais o valor da variável **vidaRestante**.

Por isso, precisamos juntar esses dois textos. Vamos utilizar o bloco de manipulação de texto **Juntar**.

The image shows the Scratch IDE interface. On the left, the 'Blocos' (Blocks) panel is visible, with the 'Texto' (Text) category selected. On the right, the 'Visualizador' (Viewer) panel shows a script for a mosquito character. The script consists of the following blocks:

- inicializar global vidaRestante para 3** (Initialize global variable 'vidaRestante' to 3)
- quando Mosquito . Tocou** (When Mosquito is clicked)
- fazer** (do) block containing:
  - ajustar global vidaRestante para obter global vida** (Adjust global variable 'vidaRestante' to get global variable 'vida')
- para AtualizarVidaRestante** (for) loop block containing:
  - fazer** (do) block containing:
    - ajustar Legenda1 . Texto para juntar** (Adjust 'Legenda1' text to 'juntar')

Red arrows highlight the 'juntar' block in the 'Blocos' panel and the 'ajustar Legenda1 . Texto para juntar' block in the script.

# VIDAS DO MOSQUITO

Encaixe no primeiro espaço o bloco de texto e escreva o valor “Vida:”.

No segundo espaço encaixe o valor da variável “vidaRestante” (bloco **obter** ...).



# VIDAS DO MOSQUITO

Não podemos esquecer de chamar esse procedimento para atualizar o texto na tela quando o mosquito for tocado.

The image shows a programming environment with two main panels: 'Blocos' (Blocks) on the left and 'Visualizador' (Viewer) on the right. The 'Blocos' panel is organized into categories: Internos (Internal), Screen 1, Pintura 1 (Paint 1), and Qualquer componente (Any component). Under 'Internos', there are sub-categories: Controle (Control), Lógica (Logic), Matemática (Mathematics), Texto (Text), Listas (Lists), Cores (Colors), Variáveis (Variables), and Procedimentos (Procedures). Under 'Pintura 1', there are 'Mosquito' and 'BotãoIniciar' (Start Button). Under 'Qualquer componente', there is 'Temporizador1' (Timer 1). The 'Visualizador' panel shows a sequence of code blocks: 1. A 'quando BotãoIniciar .Clique' (when Start Button clicked) block followed by 'fazer ajustar Temporizador1 .Ativado para verdadeiro' (do adjust Timer1 Active to true). 2. An 'inicializar global vidaRestante para 3' (initialize global remaining life to 3) block. 3. A 'quando Mosquito .Tocou' (when Mosquito touched) block with 'x' and 'y' coordinates, followed by 'fazer ajustar global vidaRestante para obter global vidaRestante - 1' (do adjust global remaining life to get global remaining life - 1), and then 'chamar AtualizarVidaRestante' (call UpdateRemainingLife). 4. A 'para AtualizarVidaRestante' (for UpdateRemainingLife) block followed by 'fazer ajustar Legenda1 .Texto para juntar " Vida: " obter global vidaRestante' (do adjust Legend1 Text to concatenate " Vida: " get global remaining life). A small mosquito icon is visible in the top right of the viewer area.

# VIDAS DO MOSQUITO

Teste o seu jogo e veja que agora as vidas estão diminuindo quando você toca no mosquito.

Porém, ainda está difícil saber quando você acertou o toque no mosquito ou não. Para ficar mais claro para o jogador, vamos vibrar o celular quando o mosquito for tocado.

## DICA

Não se preocupe se o número de vidas ficar negativo! Isso acontece porque ainda não criamos o código para terminar o jogo. Logo faremos isso!

# VIDAS DO MOSQUITO

Volte para a tela de “Designer”.

Para vibrar o celular, vamos inserir o componente **Som** da Paleta de “Mídia”.

O componente Som é invisível e vai ficar abaixo da tela, junto com o Temporizador.

## Componentes invisíveis



A imagem é uma captura de tela do ambiente de desenvolvimento de aplicativos. À esquerda, a 'Paleta' de componentes é exibida, com a categoria 'Mídia' selecionada. A lista de componentes inclui: CâmeraDeVideo, Câmera, Escolhelmaagem, Tocador, Som, Gravador, ReconhecedorDeVoz, TextoParaFalar, ReprodutorDeVideo e TradutorYandex. À direita, o 'Visualizador' mostra uma pré-visualização de uma tela de jogo. No topo da tela, há uma barra de status com o tempo 9:48. O conteúdo da tela inclui o texto 'Vidas: 3', um ícone de um mosquito no topo, um ícone de uma caixa de música no centro e um ícone de um alto-falante no canto inferior esquerdo rotulado 'Som1'. Abaixo da tela, há um botão 'Iniciar Jogo'. Na base do visualizador, há uma seção 'Componentes invisíveis' que mostra o ícone de um relógio rotulado 'Temporizador1'. No topo do visualizador, há duas opções de configuração: 'Mostrar os componentes ocultos no Visualizador' e 'Marque para Prévisualização no tamanho de um Tablet'.

# VIDAS DO MOSQUITO

Agora vamos voltar para a tela de “Blocos”

Temos que chamar o componente **Som1** para vibrar o telefone dentro do procedimento **mosquito.Tocou**.

O bloco que devemos utilizar é chamar **Som1.Vibrar**.

The image shows the Scratch IDE interface. On the left is the 'Blocos' (Blocks) panel, and on the right is the 'Visualizador' (Viewer) panel. In the 'Blocos' panel, the 'Som1' component is selected, indicated by a red arrow. In the 'Visualizador' panel, a script is visible with several blocks: a 'quando Som1 undefined' block, a 'mensagem' block, a 'fazer' block, a 'chamar Som1.Pausar' block, a 'chamar Som1.Tocar' block, a 'chamar Som1.Retomar' block, a 'chamar Som1.Parar' block, a 'chamar Som1.Vibrar milisegs' block (highlighted with a red arrow), a 'Som1.IntervaloMinimo' block, and an 'ajustar Som1.IntervaloMinimo para' block.

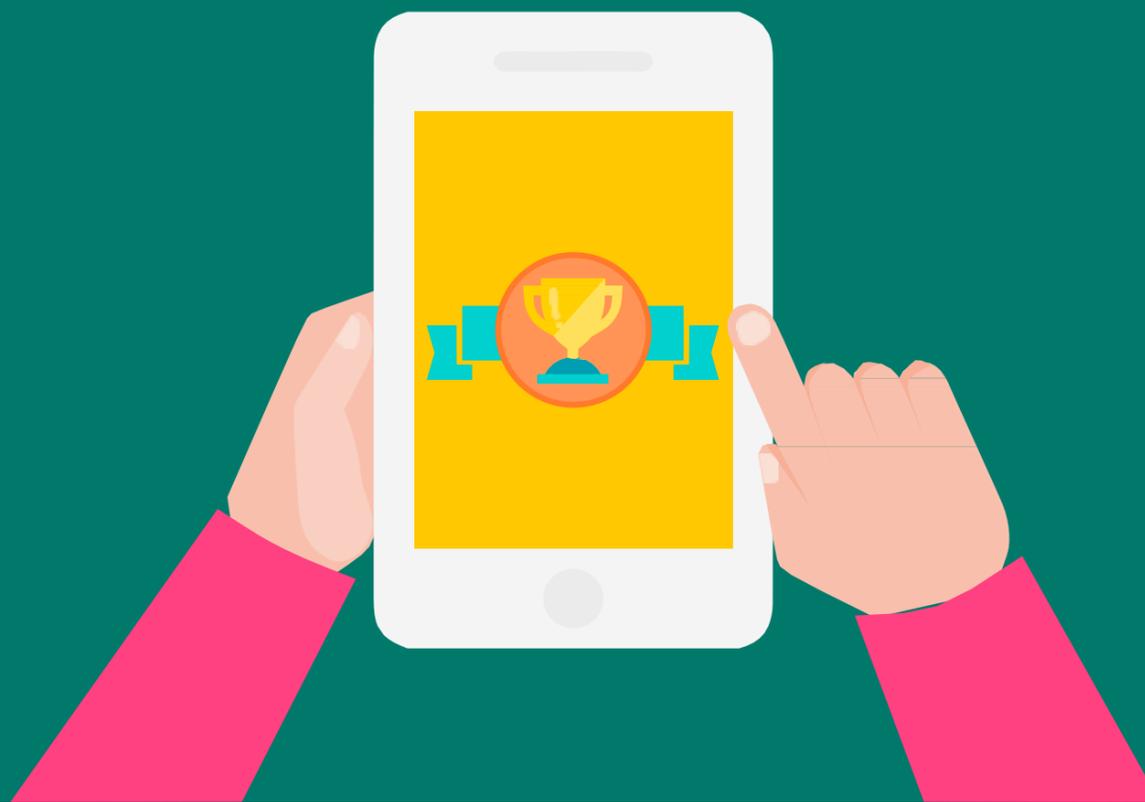
# VIDAS DO MOSQUITO

Vamos definir o tempo que desejamos que ele vibre como “100” milissegundos.



Agora você pode testar seu jogo novamente. Clique sobre o mosquito e veja se o seu celular está vibrando!

# FINALIZAR O JOGO



Como você deve ter percebido o nosso mosquito ainda não está morrendo, ele está ficando com a vida negativa. Para resolver este problema precisamos saber quando terminar o jogo, ou seja, quando o mosquito ficar com a vida igual a "0".

Para isto, depois de vibrar o telefone quando o Mosquito foi tocado, devemos verificar se a sua vida restante é maior ou igual a zero para decidir se o jogo continua ou não.

# TOMAR DECISÕES

**VIDA DO MOSQUITO = 0**



FALSO

CONTINUAR O JOGO



VERDADEIRO

TERMINAR O JOGO



Para fazer esta verificação vamos usar o Bloco de Controle “Se, então”.

# EXPRESSÕES BOOLEANAS

Computadores tomam suas decisões baseados em perguntas, cujas respostas, são duas: verdadeiro ou falso.



COMPARA SE DOIS  
VALORES SÃO IGUAIS

5=5

VERDADEIRO

5=2

FALSO

# FINALIZAR O JOGO

Para a comparação, utilizamos a operação Lógica “vidaRestante = 0”.

The image shows a Scratch-like programming environment with two main panes: 'Blocos' (Blocks) on the left and 'Visualizador' (Viewer) on the right. The 'Blocos' pane is divided into categories: Internos, Controle, Lógica, Matemática, Texto, Listas, Cores, Variáveis, and Procedimentos. The 'Lógica' category is selected, and the '=' block is highlighted with a red arrow. The 'Visualizador' pane shows a script for a game. The script starts with 'inicializar global vidaRestante para 3'. It then has a 'quando Mosquito .Tocou' event block. Inside this event, there is a 'fazer' loop block. The loop contains: 'ajustar global vidaRestante para obter global vidaRestante - 1', 'chamar AtualizarVidaRestante', and 'chamar Som1 .Vibrar milisegs 100'. After the loop, there is a 'se' block with the condition 'obter global vidaRestante = 0' and an 'então' block. The 'então' block contains a 'para AtualizarVidaRestante' block, which has a 'fazer' loop with 'ajustar Legenda1 . Texto para juntar " Vida: " obter global vidaRestante'.

# FINALIZAR O JOGO

Se o valor de “vidaRestante” for igual a zero devemos chamar um procedimento responsável por terminar o jogo.

Vamos criar um novo procedimento “FinalizarJogo” através dos Blocos de Procedimento.

The image shows the Scratch development environment. On the left is the 'Blocos' (Blocks) panel, and on the right is the 'Visualizador' (Stage) area. In the 'Blocos' panel, the 'Procedimentos' (Procedures) category is selected, indicated by a red arrow. In the 'Visualizador' area, a script for a mosquito object is shown. The script starts with a 'quando .Tocou' (when clicked) event. It then contains a 'para procedimento' (do procedure) block with a red arrow pointing to it, followed by 'chamar AtualizarVidaRestante', 'chamar FinalizarJogo', and 'chamar MoverMosquito'. The 'FinalizarJogo' procedure is shown as a 'para FinalizarJogo' block with a red arrow pointing to it, containing a 'fazer' (do) block with 'ajustar Legenda1 . Texto'.

# FINALIZAR O JOGO

## O QUE DEVE SER FEITO AO FINALIZAR O JOGO?

1. Precisamos fazer o mosquito parar de se movimentar.
2. Precisamos informar na tela que o mosquito foi eliminado.
3. Precisamos parar de diminuir o numero de vidas do mosquito e atualizar a tela ao clicar no mosquito.

# FINALIZAR O JOGO

1. Precisamos fazer o mosquito parar de se movimentar.
  - a) Para solucionar o primeiro problema, vamos desativar o Temporizador.
  - b) Adicione o bloco “ajustar Temporizador1.Ativado” informando o valor “falso”.



## FINALIZAR O JOGO

2. Precisamos informar na tela que o mosquito foi eliminado.

a) Vamos mudar o texto da Legenda1 para informar “Você eliminou o mosquito!”.

b) Use o bloco “ajustar Legenda1.Texto” e escreva o texto acima.



# FINALIZAR O JOGO

3. Precisamos parar de diminuir o numero de vidas do mosquito

a) Lembra que toda a lógica de verificar e diminuir vida estava no procedimento “quando Mosquito.tocou”?

b) Para parar de diminuir as vidas é só desativar o mosquito, assim ele não vai mais fazer ação nenhuma quando for tocado.



```
para FinalizarJogo
fazer
  ajustar Temporizador1 . Ativado para falso
  ajustar Legenda1 . Texto para "Você eliminou o mosquito!"
  ajustar Mosquito . Ativado para falso
```



```
quando Mosquito . Tocou
  x y
  fazer
    ajustar global vidaRestante para obter global vidaRestante - 1
    chamar AtualizarVidaRestante
    chamar Som1 . Vibrar
      milisegs 100
    se
      obter global vidaRestante = 0
    então
```

# FINALIZAR O JOGO

Agora que nosso procedimento para **FinalizarJogo** está pronto só precisamos chamá-lo quando a **vidaRestante** for igual a zero, ou seja, dentro do bloco “então”.

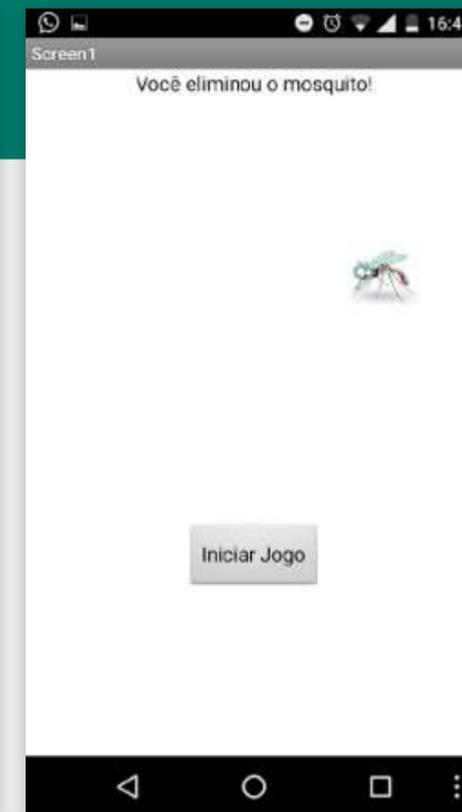
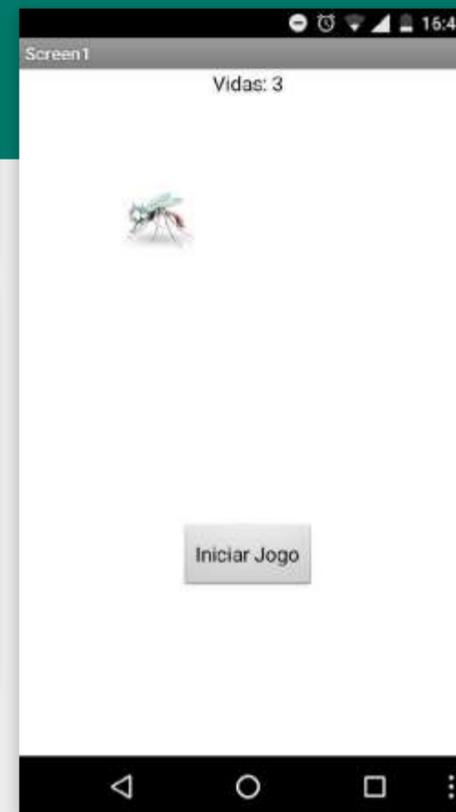
The image shows a programming environment with two main panels: 'Blocos' (Blocks) on the left and 'Visualizador' (Visualizer) on the right. The 'Blocos' panel shows a tree view with 'Internos' (Internal) blocks, including 'Controle', 'Lógica', 'Matemática', 'Texto', 'Listas', 'Cores', 'Variáveis', and 'Procedimentos'. The 'Visualizador' panel shows a script for a 'Mosquito' object. The script starts with 'inicializar global vidaRestante para 3'. It then has a 'quando Mosquito .Tocou' event block. Inside this event, there is a 'fazer' loop with 'ajustar global vidaRestante para obter global vidaRestante', 'chamar AtualizarVidaRestante', and 'chamar Som1 .Vibrar milisegs 100'. A 'se' block follows, with the condition 'obter global vidaRestante = 0'. Inside the 'se' block, there is an 'então' block with 'chamar FinalizarJogo'. Two red arrows point to the 'chamar FinalizarJogo' block in the 'Blocos' panel and the 'então' block in the 'Visualizador' panel.

# FINALIZAR O JOGO

## AGORA VAMOS TESTAR!

Veja no seu celular se o jogo está finalizando corretamente.

Você verificou que ao finalizar o jogo e depois recomeçar clicando no botão Iniciar Jogo o texto da **Legenda1** ainda está mostrando que o mosquito foi eliminado mesmo ele estando vivo? O mosquito não está mais perdendo vidas? Temos bastante coisas para consertar!



# REINICIAR O JOGO



Como no procedimento FinalizarJogo estamos trocando o texto da Legenda1, devemos trocá-lo novamente ao Iniciar o Jogo. Como o nosso jogo é iniciado com o clique do “BotãoIniciar” vamos alterar o seu conteúdo e fazer com que ele:

1. Inicie o valor da variável “vidaRestante” para “3” novamente.
2. Atualize o texto da “Legenda1” para o texto “Vida Restante: 3”.
3. E ativar o mosquito para ele perder vida ao ser tocado.

# REINICIANDO O JOGO

1. Inicie o valor da variável “vidaRestante” para “3” novamente.

a) Para alterar o valor da variável “vidaRestante” vamos utilizar o bloco “ajustar vidaRestante para” e informar o valor “3”.

Visualizador

The image shows a visual programming interface with a light green header labeled "Visualizador". The workspace contains several code blocks:

- On the left side, there are three blocks: "inicializar global nome para", "obter", and "star para".
- Below these is a "dentro de" block with a blue "e" icon, containing "inicializar local nome para".
- On the right side, there are three event-driven blocks:
  - "quando Temporizador1 .Disparo" followed by "fazer chamar MoverMosquito".
  - "quando BotãoIniciar .Clique" followed by "fazer" containing two stacked blocks: "ajustar Temporizador1 .Ativado para verdadeiro" and "ajustar global vidaRestante para 3".
  - "inicializar global vidaRestante para 3".

# REINICIANDO O JOGO

2. Atualize o texto da “Legenda1” para o texto “Vida Restante: 3”  
Para isso só precisamos chamar o procedimento “AtualizarVidaRestante”.

The screenshot displays a programming environment with two main panels: 'Blocos' (Blocks) on the left and 'Visualizador' (Visualizer) on the right. The 'Blocos' panel shows a category tree with 'Procedimentos' (Procedures) highlighted. The 'Visualizador' panel shows a script for the 'BotãoIniciar' (Start Button) click event. The script includes the following blocks:

- quando BotãoIniciar .Clique
- fazer
  - ajustar Temporizador1 .Ativado para verdadeiro
  - ajustar global vidaRestante para 3
  - chamar AtualizarVidaRestante

Below this script, there is a separate block: inicializar global vidaRestante para 3.

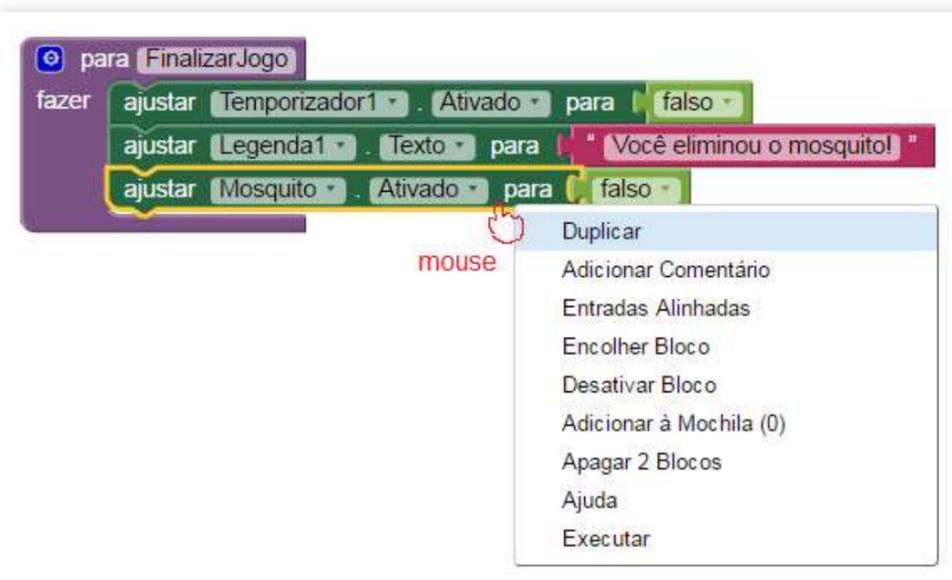
# REINICIANDO O JOGO

3. Reativar o mosquito para ele perder vida ao ser tocado

a) Para ativar o mosquito, podemos usar o bloco “ajustar Mosquito.Ativado” com o valor “Verdadeiro”.

b) A maneira mais fácil de fazer isso é clicar com o botão direito no bloco “ajustar Mosquito.Ativado” que já está dentro do procedimento “FinalizarJogo”, e escolher “Duplicar”.

c) Depois disso é só mudar o valor “falso” para “verdadeiro”. E arrastar o novo bloco para o procedimento “quando BotãoIniciar.Clique”





**ADICIONANDO  
TEMPO AO JOGO**

Agora o jogo deve estar funcionando perfeitamente!

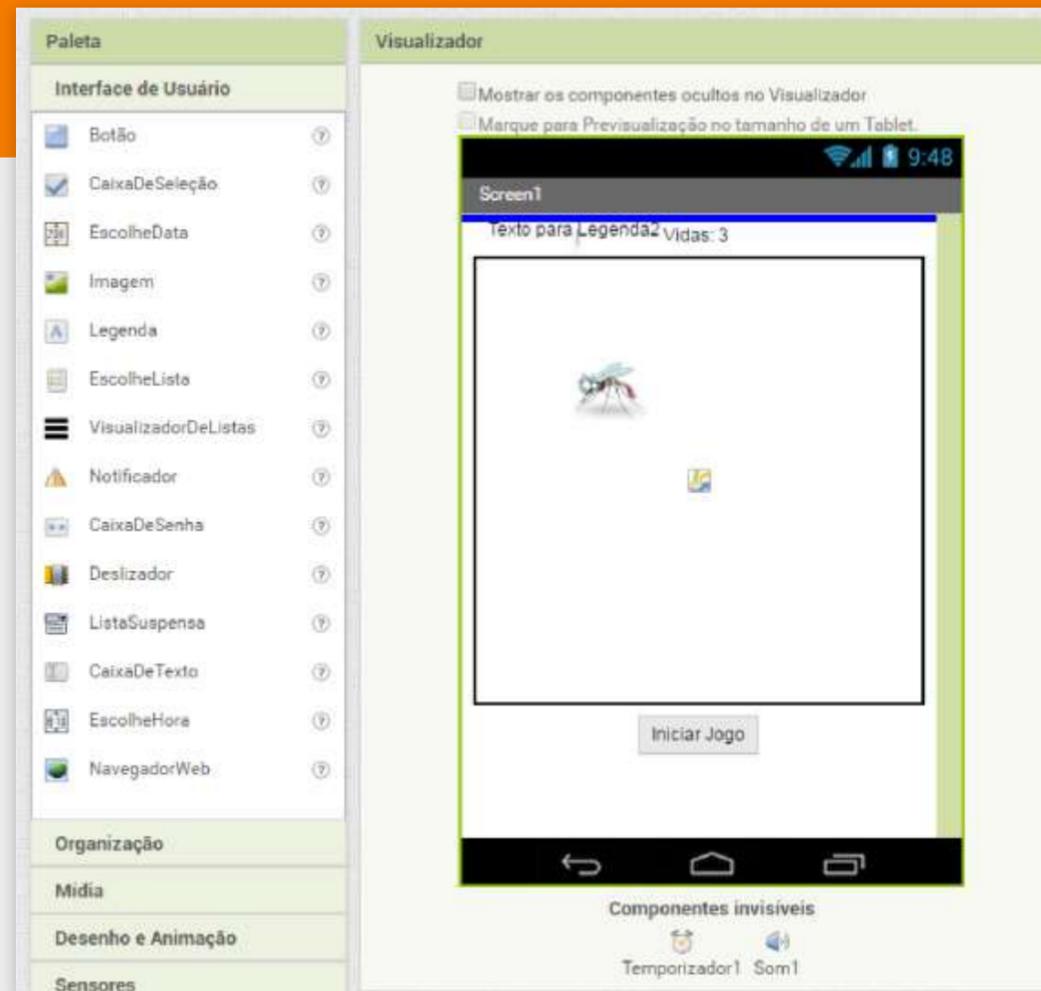
Você pode notar que o jogador ainda não possui nenhuma maneira de ser derrotado.

Para isto ocorrer, vamos fornecer um tempo máximo para o jogador eliminar o mosquito: caso o tempo acabe antes do mosquito perder todas as vidas, o jogador perde o jogo.

# ADICIONANDO TEMPO

Volte para a tela de “Designer”.

Vamos adicionar um novo componente Legenda para mostrar o tempo restante, preferencialmente acima da **Legenda1** (da vida).



# ADICIONANDO TEMPO

Vamos alterar o texto da “Legenda2” para mostrar quanto tempo o jogador ainda tem para eliminar o mosquito.

Para fazer isto é necessário seleccionar o componente “Legenda2” e no campo Texto digitar “Tempo: 10”

The image shows a software development interface with two main panels: 'Componentes' and 'Propriedades'. The 'Componentes' panel displays a hierarchical tree view of the application's components. Under 'Screen1', the following components are listed: 'Legenda2' (highlighted in green), 'Legenda1', 'Pintura1', 'Mosquito', 'BotãoIniciar', 'Temporizador1', and 'Som1'. At the bottom of this panel are 'Renomear' and 'Apagar' buttons. The 'Propriedades' panel shows the configuration for the selected 'Legenda2' component. It includes several properties: 'CorDeFundo' (set to 'Nenhum'), 'FonteNegrito' (unchecked), 'FonteTálico' (unchecked), 'TamanhoDaFonte' (set to 14.0), 'FamíliaDaFonte' (set to 'padrão'), 'HTMLFormat' (unchecked), 'TemMargens' (checked), 'Altura' (set to 'Automático...'), 'Largura' (set to 'Automático...'), and 'Texto' (set to 'Tempo: 10').

# ADICIONANDO TEMPO

Para contar o tempo vamos utilizar um novo Temporizador.

Selecione a paleta Sensores e arraste o componente Temporizador até a tela.

Configure o **Temporizador2** para realizar disparos a cada 1 segundo (1000 milissegundos), vamos deixá-lo desativado por enquanto.

The screenshot displays the LEGO Mindstorms software interface with four main panels:

- Paleta (Palette):** Lists various components under categories like 'Sensores' (Sensors), 'Social', 'Armazenamento' (Storage), 'Conectividade' (Connectivity), 'LEGO MINDSTORMS', and 'Experimental'. The 'Temporizador' (Timer) component is highlighted in the 'Sensores' category.
- Visualizador (Viewer):** Shows a mobile app preview with a mosquito icon and a timer component labeled 'Temporizador2' on the screen. A black arrow points from the 'Temporizador' in the palette to this component on the screen.
- Componentes (Components):** Lists the components on the screen, including 'Temporizador2', which is highlighted in green. A red arrow points from this component to the 'Propriedades' panel.
- Propriedades (Properties):** Shows the configuration for 'Temporizador2', including 'Disparos Contínuos' (Continuous Shots) checked, 'Ativado' (Activated) unchecked, and 'Intervalo' (Interval) set to 1000.

# ADICIONANDO TEMPO

Voltamos a tela de “Blocos” para adaptar o nosso jogo.

```
quando BotãoIniciar .Clique
fazer
  ajustar Temporizador1 . Ativado para verdadeiro
  ajustar Temporizador2 . Ativado para verdadeiro
  ajustar global vidaRestante para 3
  chamar AtualizarVidaRestante
  ajustar Mosquito . Ativado para verdadeiro
```

Primeiramente, vamos ativar o nosso **Temporizador2** quando o jogo for iniciado, incluindo o bloco **ajustar Temporizador2.Ativado** com valor Verdadeiro dentro do procedimento quando **BotãoIniciar.Clique**.

## DICA

Você pode Duplicar o bloco “ajustar Temporizador1.Ativado” como já ensinamos anteriormente. Lembre-se de mudar o componente para “Temporizador2”.

## ADICIONANDO TEMPO

Também precisamos lembrar de desativar o [Temporizador2](#) quando o jogo for finalizado.



Inclua o mesmo bloco com valor Falso no procedimento [FinalizarJogo](#).

# ADICIONANDO TEMPO

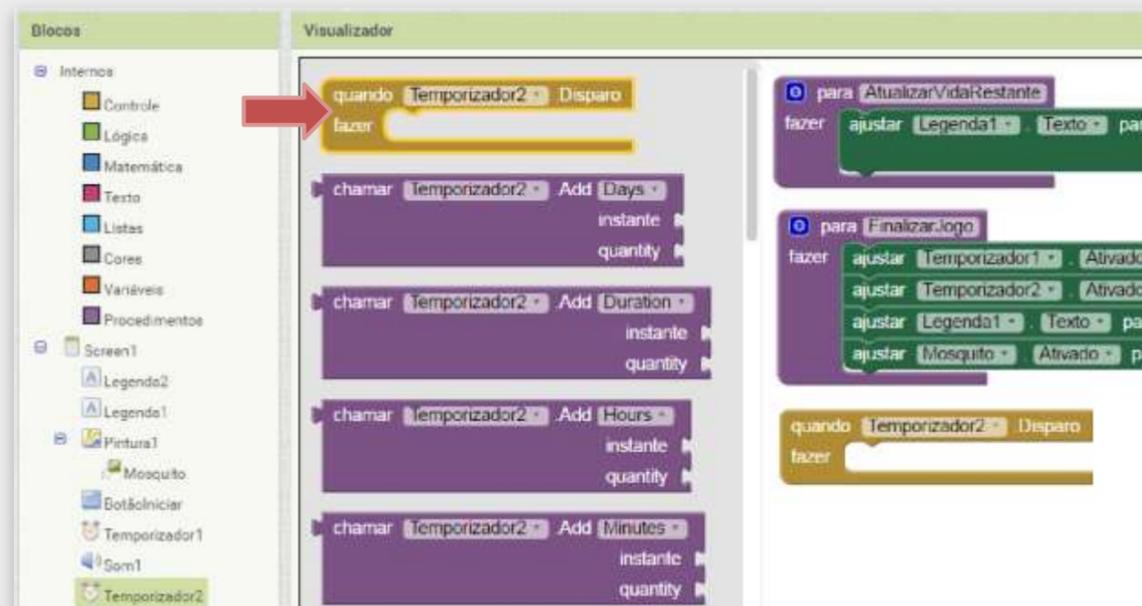
Agora precisamos criar uma variável global, responsável por armazenar o tempo restante de jogo. Vamos iniciá-la com o valor "10".

The screenshot shows a programming environment with a 'Blocos' (Blocks) panel on the left and a 'Visualizador' (Visualizer) panel on the right. The 'Blocos' panel has a 'Variáveis' (Variables) category selected. The 'Visualizador' panel shows a script for a 'BotãoIniciar' (Start Button) click event. The script includes the following blocks: 'ajustar Temporizador1 . Ativado para verdadeiro', 'ajustar Temporizador2 . Ativado para verdadeiro', 'ajustar global vidaRestante para 3', 'ajustar global tempoRestante para 10', 'chamar AtualizarVidaRestante', and 'ajustar Mosquito . Ativado para verdadeiro'. Two red arrows point from the 'ajustar global tempoRestante para 10' block in the script to the 'ajustar para' block in the 'Blocos' panel, indicating the block being used to create the global variable.

Também devemos ajustar o seu valor para "10" no procedimento de iniciar o jogo (BotãoIniciar.Clique).

# ADICIONANDO TEMPO

Precisamos atualizar o tempo restante a cada segundo, ou seja, toda vez que o “Temporizador2” disparar.



Para isso, vamos adicionar no Visualizador o bloco “quando Temporizador2.Disparo”.

## ADICIONANDO TEMPO

Dentro do `Temporizador2.Disparo` vamos diminuir o valor da variável “tempoRestante” em “1”.



Usamos o bloco “ajustar global tempoRestante”, os blocos Matemáticos de Subtração e número inteiro “1” e o bloco “obter globalRestante”.

## ADICIONANDO TEMPO

O valor do `tempoRestante` está sendo diminuído, porém o jogador ainda não está percebendo isto. Precisamos atualizar o texto da `Legenda2` que é mostrado na tela.

Para isto, vamos criar um procedimento “AtualizarTempoRestante”, similar ao procedimento “AtualizarVidaRestante”, responsável por trocar o texto da `Legenda2`.

Lembre-se que você pode duplicar o procedimento “AtualizarVidaRestante”, e apenas mudar os parâmetros, para ficar conforme o bloco abaixo.



## ADICIONANDO TEMPO

Temos que chamar o procedimento para atualizar a tela logo após diminuir o valor “tempoRestante”, dentro do bloco “Temporizador2.Disparo”.



## ADICIONANDO TEMPO

Também precisamos chamar o procedimento “AtualizarTempoRestante” dentro do bloco “quando BotãoIniciar.Clique”, para mostrar o tempo correto quando o jogo iniciar.

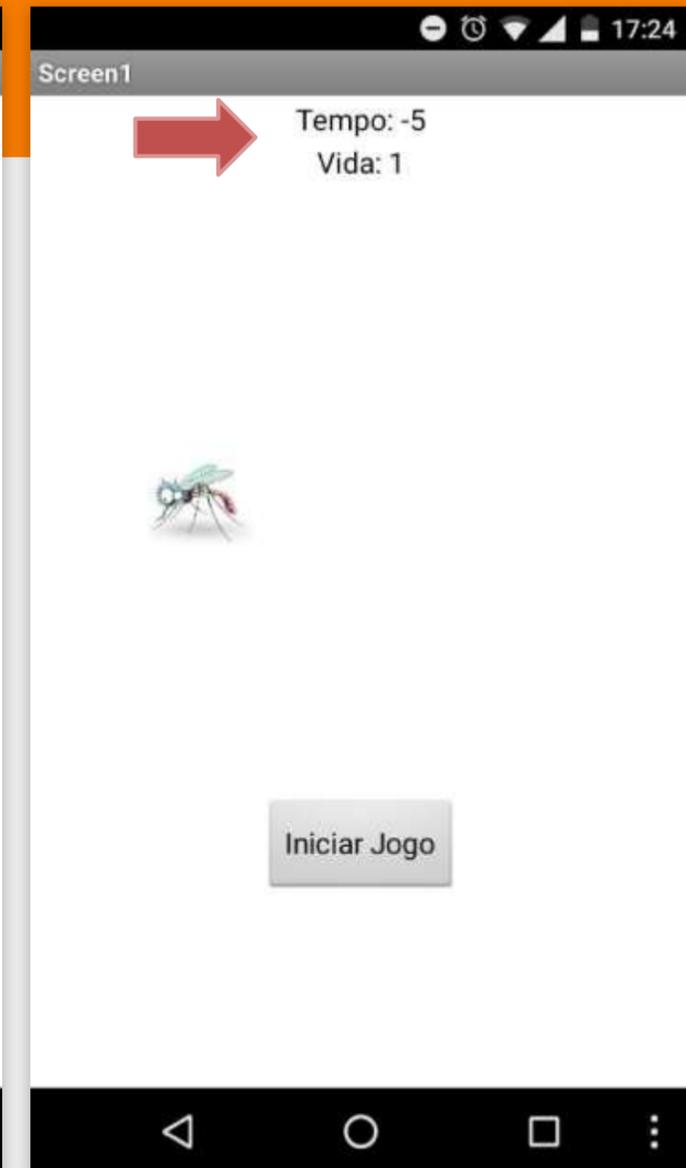
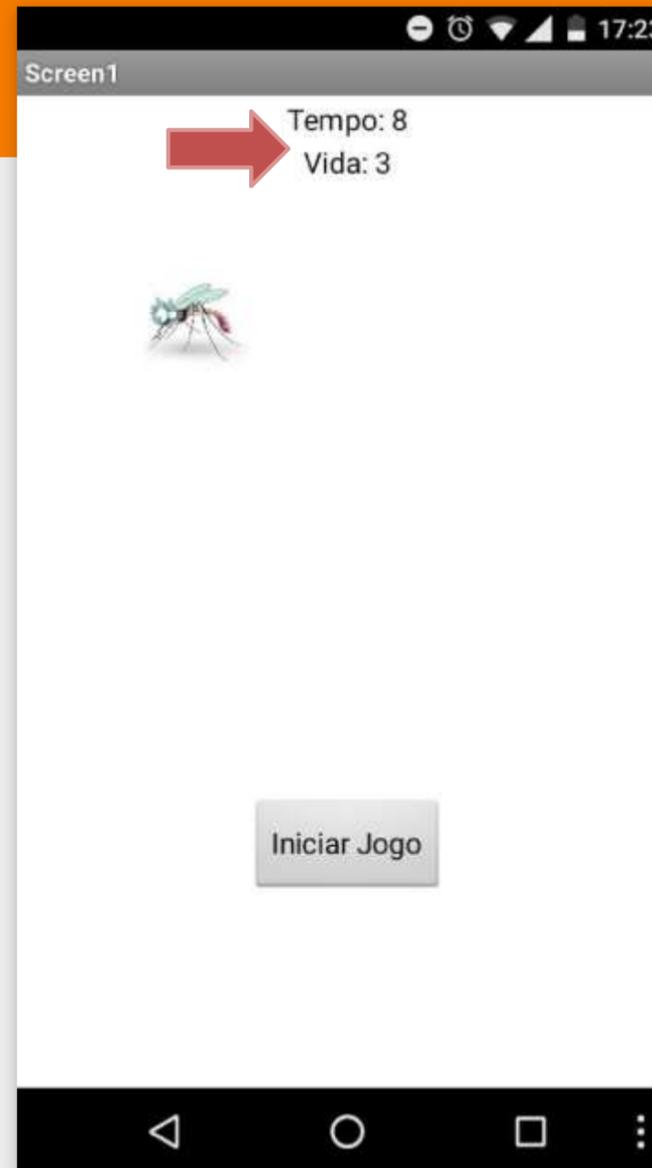


# ADICIONANDO TEMPO

## VAMOS TESTAR!

Verifique se o tempo está diminuindo na tela e se está se comportando corretamente.

Você deve ter percebido que o jogo ainda não está terminando quando o valor do “tempoRestante” chega a “0” (zero).



## ADICIONANDO TEMPO

Sendo assim, vamos começar alterando o procedimento “Temporizador2.Disparo” adicionando o bloco de controle “Se, então” e utilizar a operação lógica “tempoRestante = 0” para terminar o jogo.



# ADICIONANDO TEMPO

Devemos finalizar o jogo caso o tempo tenha esgotado, então vamos chamar o procedimento **FinalizarJogo**.

Teste o se o jogo está finalizando corretamente quando o tempo acaba! O jogador deve ser derrotado.

Você verificou que apesar do tempo ter se esgotado ainda apareceu que o jogador conseguiu eliminar o mosquito?



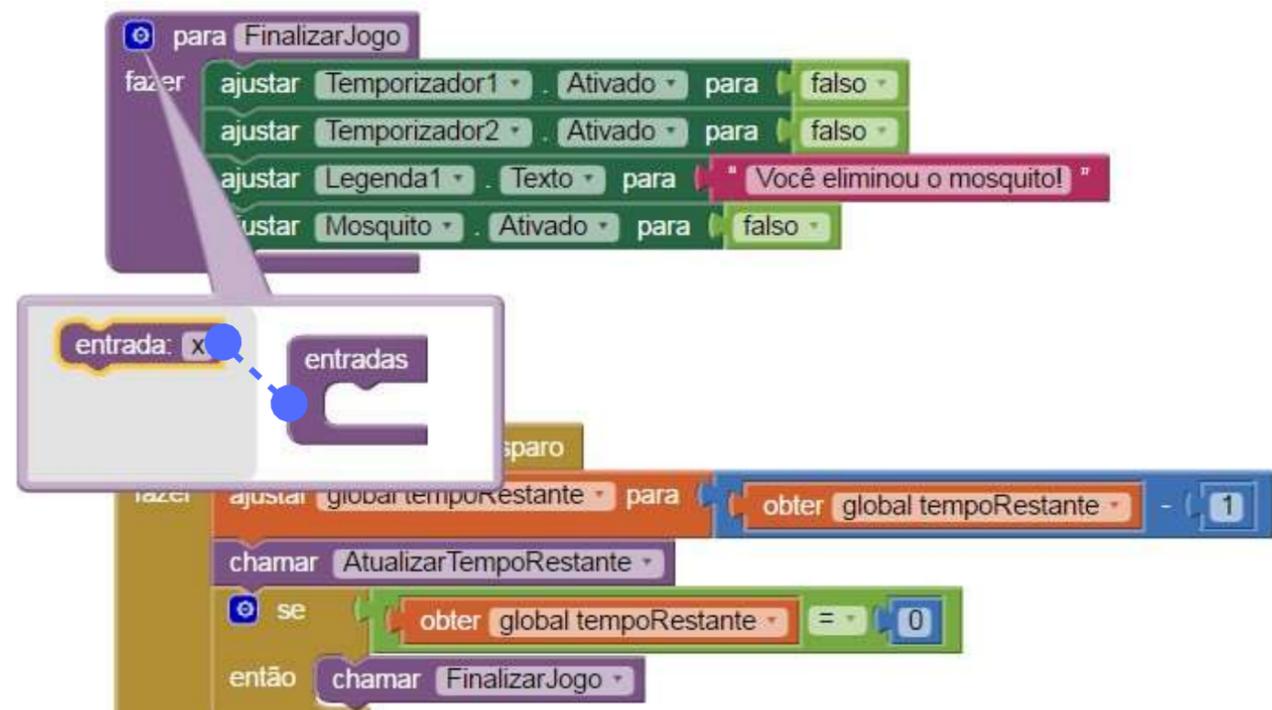
```
quando Temporizador2 .Disparo
fazer
  ajustar global tempoRestante para obter global tempoRestante - 1
  chamar AtualizarTempoRestante
  se obter global tempoRestante = 0
  então chamar FinalizarJogo
```

The image shows a Scratch code block for a timer event. The event is 'quando Temporizador2 .Disparo'. The code block contains the following actions: 'fazer' (do) containing 'ajustar global tempoRestante para obter global tempoRestante - 1' (adjust global tempoRestante to obtain global tempoRestante - 1), 'chamar AtualizarTempoRestante' (call AtualizarTempoRestante), and an 'if' block ('se obter global tempoRestante = 0') with the action 'então chamar FinalizarJogo' (then call FinalizarJogo). A red arrow points to the 'FinalizarJogo' block.

# ADICIONANDO TEMPO

Temos que modificar o procedimento “FinalizarJogo” para receber a informação se o jogador ganhou ou perdeu, e mudar o texto da [Legenda1](#) corretamente.

Para isto, clique no ícone azul no canto superior do procedimento “FinalizarJogo” e arraste o bloco “entrada” para encaixar no bloco “entradas” e troque o texto de “X” para “resultadoFinal”.



# ADICIONANDO TEMPO

DEVE FICAR ASSIM:



# ADICIONANDO TEMPO

Agora precisamos verificar o valor da variável local “resultadoFinal”. Caso esse resultado seja Verdadeiro o jogador foi o vencedor, caso contrário (senão) ele perdeu.

Primeiro adicione o bloco “se então” ao “FinalizarJogo”, e coloque uma bloco Lógico de comparação (=).

Visualizador

The image shows a visual programming environment with a sidebar on the left labeled "Visualizador" containing blocks for "verdadeiro", "falso", "não", "=", and "e". The main workspace contains two "para" (loop) blocks. The first block is "para AtualizarVidaRestante" with a "fazer" (do) block containing "ajustar Legenda1 . Texto para" and "juntar ' Vida: ' obter global vidaRestante". The second block is "para FinalizarJogo resultadoFinal" with a "fazer" block containing a "se então" (if-then) block. The "se então" block has an "=" comparison block as its condition and an "ajustar Temporizador1 . Ativado para falso" block as its action. Red arrows point to the "=" block and the "então" block.

# ADICIONANDO TEMPO

Adicione na comparação a variável **resultadoFinal** e o valor Verdadeiro.

A variável local é obtida da mesma maneira que as variáveis globais, mas ela só pode ser acessada dentro do procedimento **FinalizarJogo**.

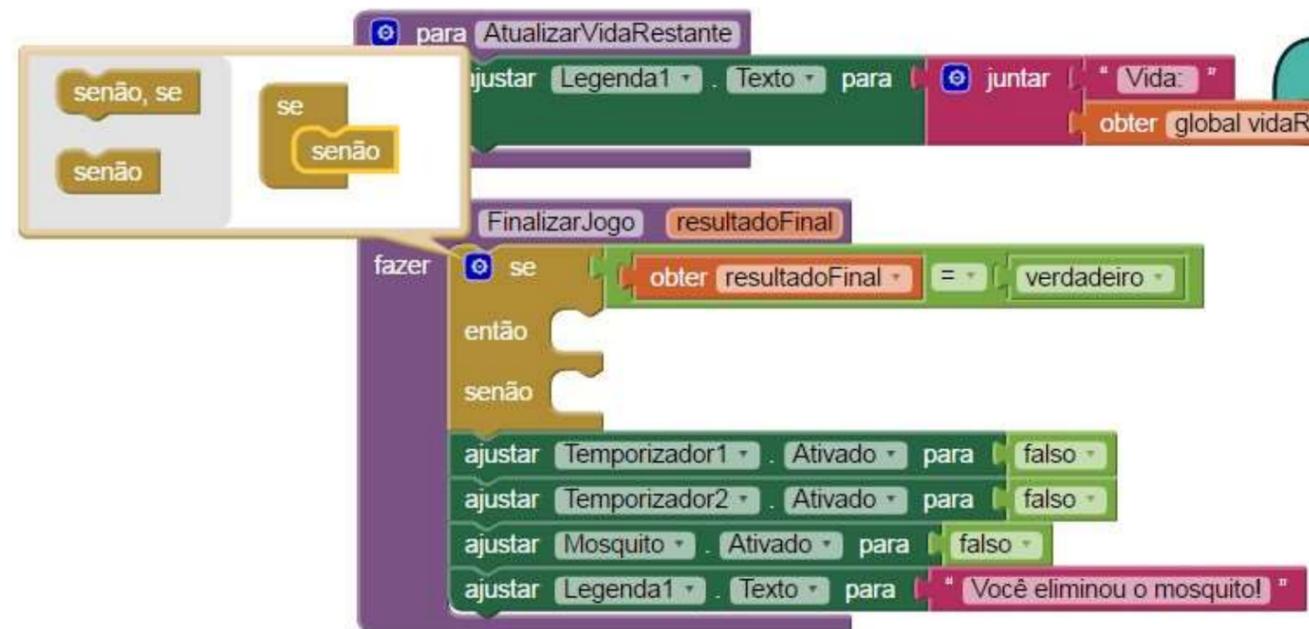


```
para FinalizarJogo resultadoFinal
fazer
  se obter resultadoFinal = verdadeiro
  então
    ajustar Temporizador1 Ativado para falso
    ajustar Temporizador2 Ativado para falso
    ajustar Mosquito Ativado para falso
    ajustar Legenda1 Texto para "Você eliminou o mosquito!"
```

The image shows a Scratch code block for a procedure named 'FinalizarJogo' with a local variable 'resultadoFinal'. The code is structured as follows: a 'para' block containing a 'fazer' block. Inside the 'fazer' block, there is a 'se' block with a red arrow pointing to the 'obter resultadoFinal' block. The 'se' block is followed by an 'então' block containing four 'ajustar' blocks: 'ajustar Temporizador1 Ativado para falso', 'ajustar Temporizador2 Ativado para falso', 'ajustar Mosquito Ativado para falso', and 'ajustar Legenda1 Texto para "Você eliminou o mosquito!"'.

## ADICIONANDO TEMPO

Para mudar o bloco “Se então” para “Se então senão”, clique no ícone azul do bloco e arraste o bloco “senão” para dentro do “se”.



## ADICIONANDO TEMPO

Quando o resultadoFinal for verdadeiro, vamos atualizar a **Legenda1** com o texto de vitória, senão vamos atualizar com o texto de derrota (“Você perdeu!”).



# ADICIONANDO TEMPO

Agora, sempre que chamamos o procedimento **FinalizarJogo** temos que passar o valor de **resultadoFinal** que queremos.

Visualizador

```
quando Mosquito .Tocou
  fazer
    ajustar global vidaRestante para obter global vidaRestante - 1
    chamar AtualizarVidaRestante
    chamar Som1 .Vibrar milisegs 100
  se
    obter global vidaRestante = 0
  então
    chamar FinalizarJogo resultadoFinal verdadeiro
```

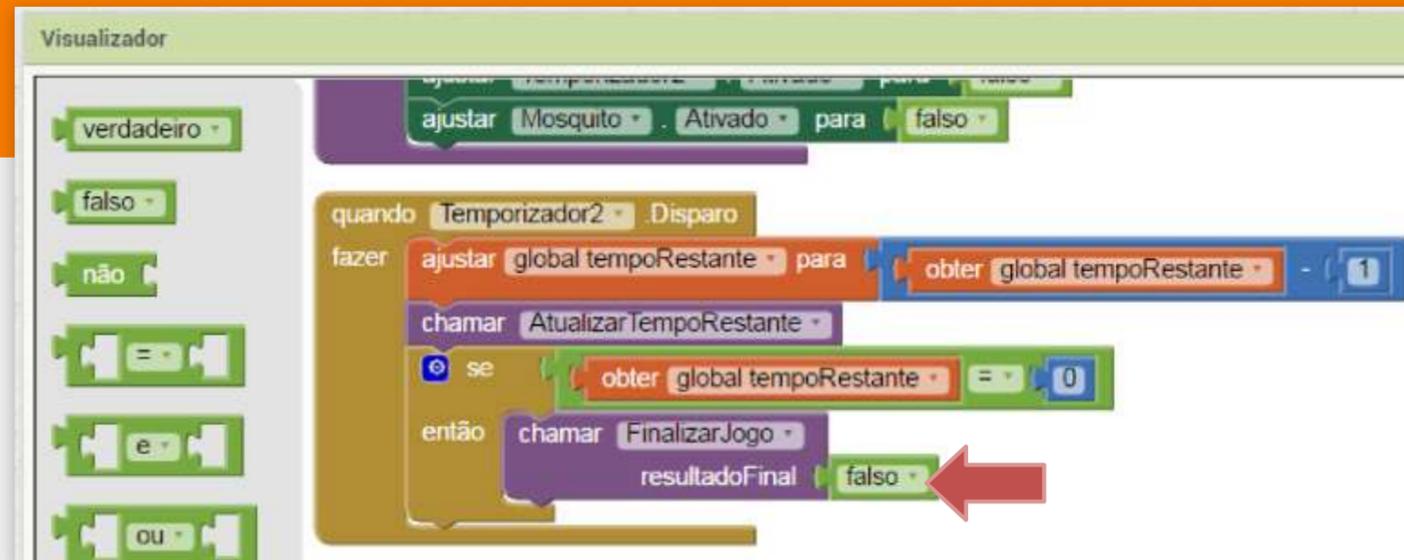
Vamos atualizar o bloco “Mosquito.Tocou” passando o valor “Verdadeiro” como parâmetro, pois neste procedimento o jogador foi vencedor.

# ADICIONANDO TEMPO

E também atualizar a chamada do procedimento **FinalizarJogo** no bloco “quando Temporizador2.Disparo” passando o valor Falso como parâmetro, pois neste procedimento o jogador perdeu.

Agora verifique se o seu jogo está funcionando perfeitamente com o tempo para matar o mosquito!

Assim conseguimos finalizar a funcionalidade de Adicionar Tempo para matar o mosquito.





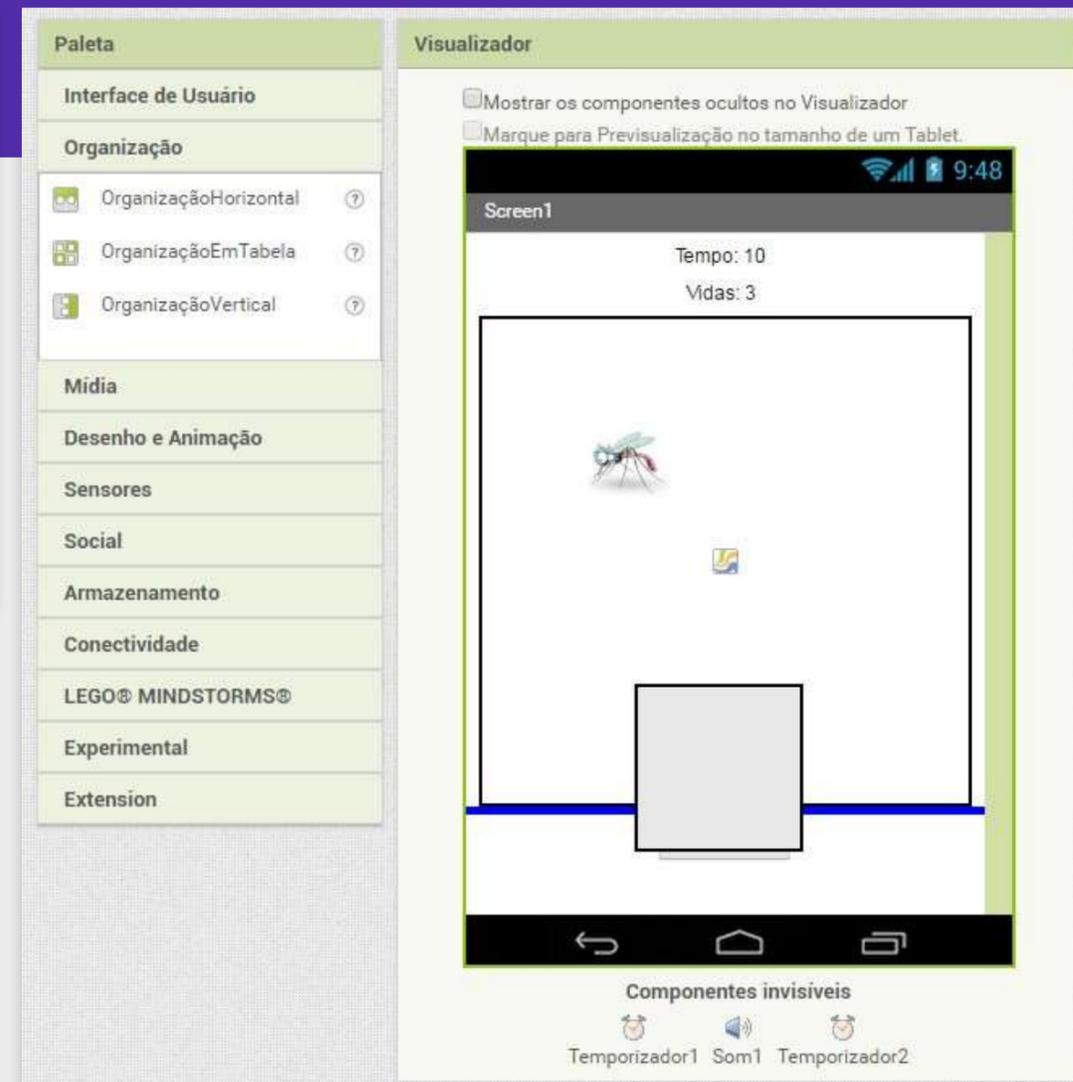
**ADICIONAR  
DIFICULDADE**

Para deixar o jogo mais divertido, vamos deixar o jogador escolher a dificuldade do jogo.

A variação de dificuldade vai atuar sobre a variável tempoRestante e a velocidade em que o mosquito irá se movimentar.

# ADICIONAR DIFICULDADE

Para exibir na tela os níveis de dificuldade vamos adicionar o componente **ListaSuspensa** ao lado do **BotãoIniciar**. Para isto devemos ir ao “Designer” e selecionar a paleta **Organização** e arrastar o componente **OrganizaçãoHorizontal** para uma posição acima do **BotãoIniciar**.



# ADICIONAR DIFICULDADE

O componente **OrganizaçãoHorizontal1** acabou por colocar o **BotãoIniciar** para fora da tela, por isso temos que configurar a sua altura como "20", e arrastar o **BotãoIniciar** para dentro da **OrganizacaoHorizontal1**.

The image shows a screenshot of a mobile application development IDE with three main panels: Visualizador, Componentes, and Propriedades.

- Visualizador:** Displays a mobile app preview. At the top, it shows "Tempo: 10" and "Vidas: 3". Below this is a large white rectangle representing the game area, containing a mosquito icon and a small square icon. At the bottom, there is a button labeled "Iniciar Jogo". A red arrow points to this button.
- Componentes:** Lists the components of the app. A red arrow points to "OrganizaçãoHorizontal1".
- Propriedades:** Shows the properties for "OrganizaçãoHorizontal1". The "Altura" (Height) property is set to "20" under the "Preencher principal pontos" (Fill main points) option. A red arrow points to the "20" value in the input field. Other properties include "AlinhamentoHorizontal" (Esquerda), "AlinhamentoVertical" (Topo), "CorDeFundo" (Padrão), and "Visível" (checked).

At the bottom of the IDE, there are buttons for "Renomear" and "Apagar", and a "Mídia" section.

# ADICIONAR DIFICULDADE

Agora devemos clicar na paleta Interface de Usuário e arrastar o componente **ListaSuspensa** para dentro da **OrganizaçãoHorizontal1** recém adicionada, ao lado do **BotãoIniciar**.

The image shows a software development environment with two main panels: 'Paleta' (Palette) and 'Visualizador' (Previewer).

**Paleta (Palette):**

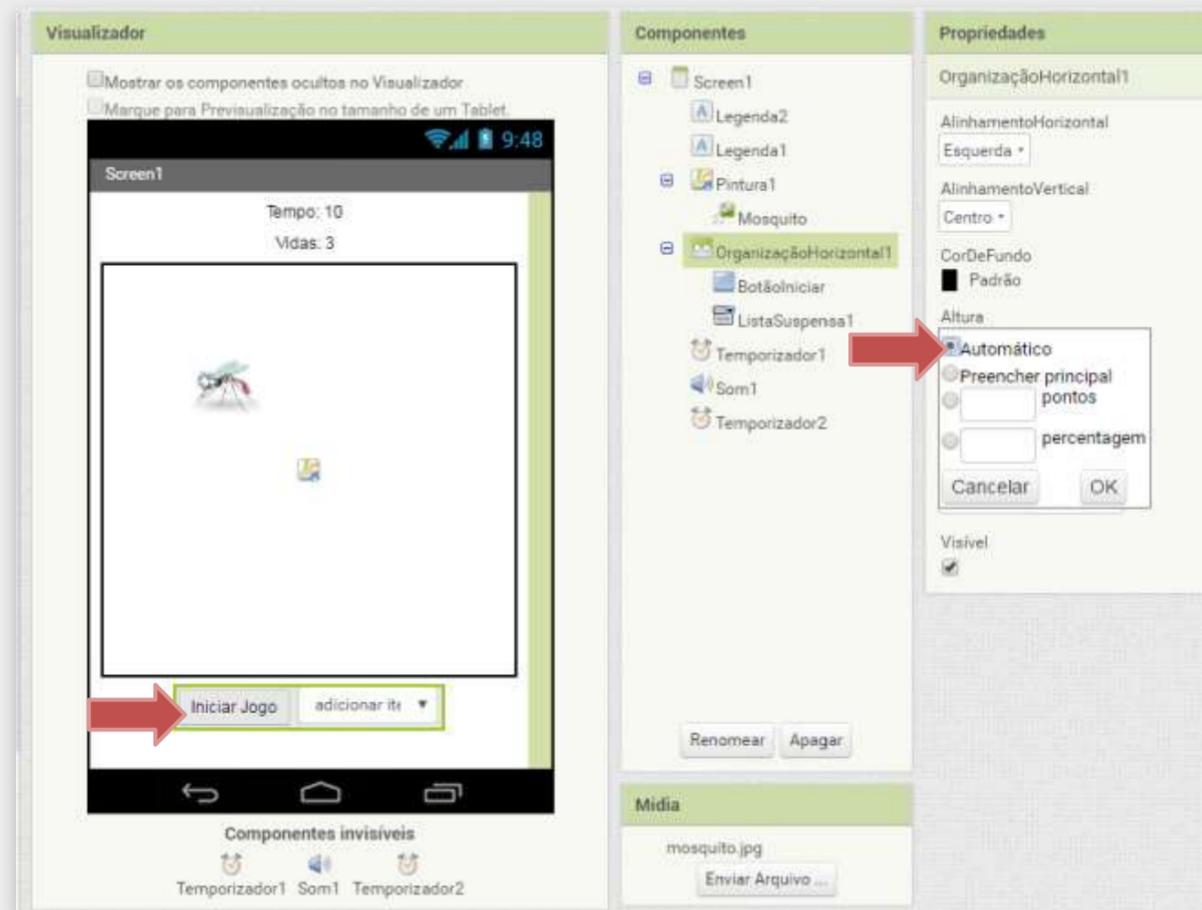
- Interface de Usuário (User Interface):**
  - Botão
  - CaixaDeSeleção
  - EscolheData
  - Imagem
  - Legenda
  - EscolheLista
  - VisualizadorDeListas
  - Notificador
  - CaixaDeSenha
  - Deslizador
  - ListaSuspensa
  - CaixaDeTexto
  - EscolheHora
  - NavegadorWeb
- Organização**
- Mídia**
- Desenho e Animação**
- Sensores**

**Visualizador (Previewer):**

- Mostrar os componentes ocultos no Visualizador
- Marque para Previsualização no tamanho de um Tablet
- Screen1
  - Tempo: 10
  - Vidas: 3
  - Visual area containing a fly icon and a small square icon.
  - Botão Iniciar
  - Menu 'adicionar ite' (add item)
- Componentes invisíveis (Invisible components):
  - Temporizador1
  - Som1
  - Temporizador2

# ADICIONAR DIFICULDADE

Após colocar a lista no lugar, podemos trocar a altura da **OrganizaçãoHorizontal1** para automática novamente, assim ela vai ficar no tamanho adequado para o botão e a lista.

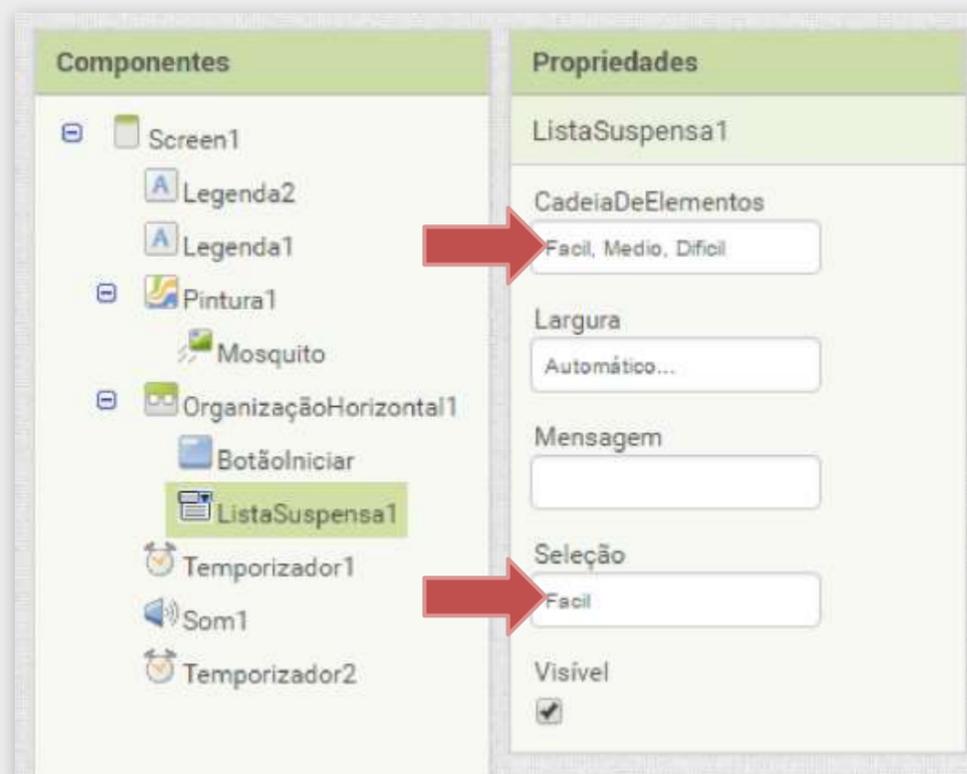


# ADICIONAR DIFICULDADE

Agora devemos configurar o componente **ListaSuspensa1** para apresentar os valores “Facil, Medio e Dificil”. Para isto preencha os campos da seguinte maneira:

## DICA

Para adicionar os elementos da lista coloque cada item separado por vírgula e sem espaço entre a “,” e o texto. Por exemplo: “Facil,Medio,Dificil”.



# ADICIONAR DIFICULDADE

Após adicionar a **ListaSuspensa1** no bloco “Designer” precisamos voltar ao “Blocos” para configurar a sua funcionalidade.

Primeiro precisamos inicializar uma variável global chamada de **nivelSelecioneado** com o valor inicial em “1”.

The screenshot shows the Scratch 'Visualizador' window with the following code blocks:

- inicializar global nome para
- obter
- ajustar para
- inicializar local nome para dentro de
- quando Temporizador1 .Disparo
- fazer chamar MoverMosquito
- inicializar global vidaRestante para 3
- inicializar global tempoRestante para 10
- inicializar global nivelSelecioneado para 1

Iremos definir que o **nivelSelecioneado** receberá os seguintes valores:

- 1 = Fácil
- 2 = Médio
- 3 = Difícil

# ADICIONAR DIFICULDADE

Utilizaremos o bloco “quando ListaSuspensa1.DepoisDeSelecionar” para definir qual o valor da variável nivelSelecionado. Este bloco possui uma variável local chamada de seleção que contém o valor escolhido (Fácil, Médio ou Difícil).

The image shows a Scratch script editor with two script areas. The left area, titled 'Visualizador', contains a sequence of blocks for a list object named 'ListaSuspensa1':

- A yellow 'quando ListaSuspensa1 . DepoisDeSelecionar' block with a 'seleção' block inside its 'fazer' loop.
- A purple 'chamar ListaSuspensa1 . MostrarListaSuspensa' block.
- A green 'ListaSuspensa1 . Elementos' block.
- A green 'ajustar ListaSuspensa1 . Elementos para' block.
- A green 'ajustar ListaSuspensa1 . CadeiaDeElementos para' block.
- A green 'ListaSuspensa1 . Altura' block.
- A green 'ajustar ListaSuspensa1 . Altura para' block.
- A green 'ajustar ListaSuspensa1 . PercentualDeAltura para' block.

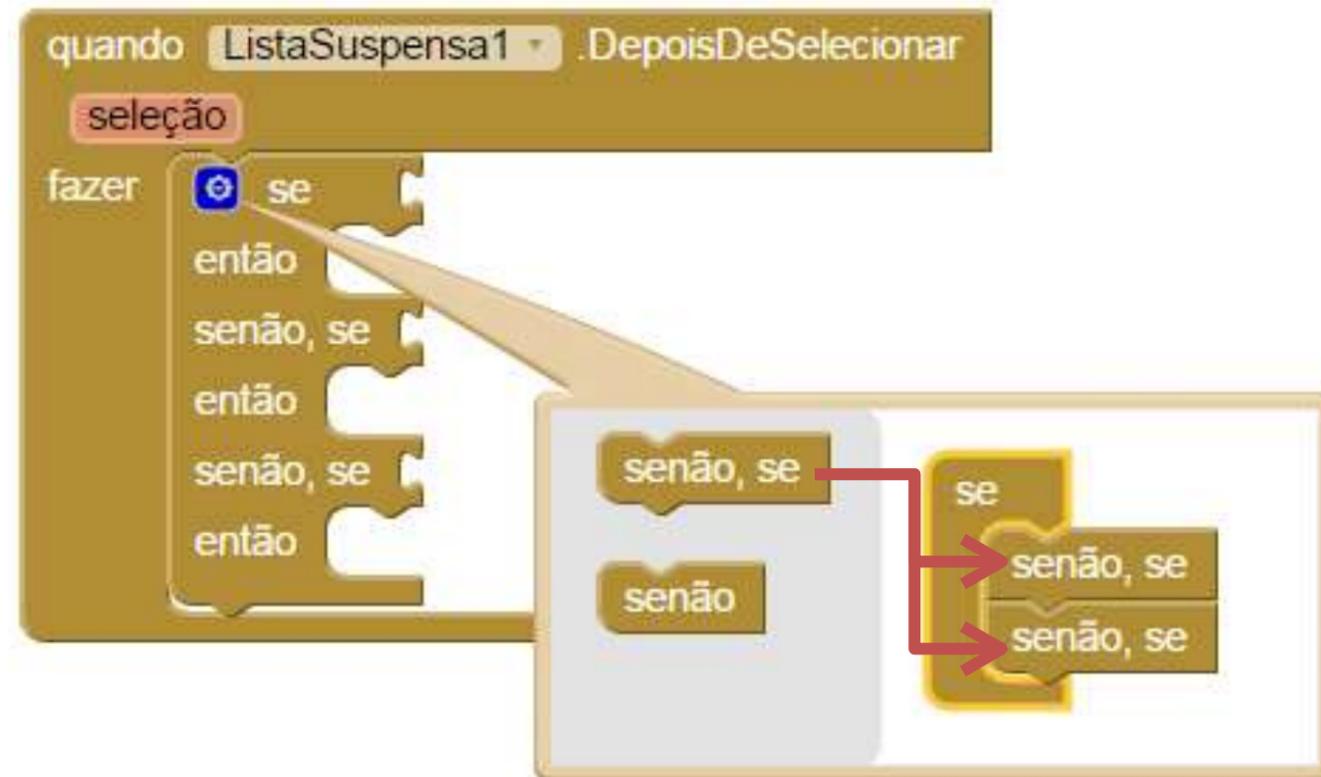
The right area contains a script for a timer object named 'Temporizador2':

- A yellow 'quando Temporizador2 . Disparo' block.
- A purple 'fazer' loop containing:
  - An orange 'ajustar global tempoRestante para' block.
  - A purple 'chamar AtualizarTempoRestante' block.
  - A green 'se' block with a blue eye icon, containing a green 'obter global tempoRestante' block.
  - A purple 'então' block containing a purple 'chamar FinalizarJogo' block with a 'resultadoFinal' block set to 'falso'.
- A purple 'para AtualizarTempoRestante' block.
- A purple 'fazer' loop containing a green 'ajustar Legenda2 . Texto para' block.
- A yellow 'quando ListaSuspensa1 . DepoisDeSelecionar' block with a 'seleção' block inside its 'fazer' loop. A red arrow points to this block from the text above.

## ADICIONAR DIFICULDADE

Iremos utilizar o bloco “Se então” e utilizar o ícone azul do canto superior para configurá-lo de acordo com a nossa lógica.

Precisamos adicionar duas vezes o bloco “senão, se” dentro do bloco “Se”.



# ADICIONAR DIFICULDADE

## NOSSA LÓGICA SERÁ

Se (seleção = “Facil”)  
Então nivelSelecionado = 1

Senão Se (seleção = “Medio”)  
Então nivelSelecionado = 2

Senão Se (seleção = “Difícil”)  
Então nivelSelecionado = 3

# ADICIONAR DIFICULDADE

Como o valor da seleção é um texto, devemos utilizar o bloco comparar textos para saber se o valor de seleção = "Facil" ou algum dos outros valores definidos.

Substitua o símbolo "<" por "=".

Adicione o bloco obter seleção e o texto "Facil".

The screenshot shows a block editor interface with a 'Blocos' (Blocks) panel on the left and a 'Visualizador' (Viewer) panel on the right. The 'Blocos' panel is expanded to show the 'Texto' (Text) category, which includes blocks like 'comprimento', 'é vazio?', 'comparar textos', 'apagar espaços', 'maiúsculas', and 'começa em texto'. The 'Visualizador' panel shows a script for a 'ListaSuspensa1' (Dropdown1) widget. The script starts with a 'quando ListaSuspensa1 . DepoisDeSelecionar' (when selected) event block. Inside the event block, there is a 'se' (if) block. The 'se' block has a 'comparar textos' (compare text) block as its condition. The 'comparar textos' block has 'obter seleção' (get selection) as its first input and '"Facil"' as its second input. The operator in the 'comparar textos' block is set to '='. A red arrow points to the '=' operator. Above the 'se' block, there is a 'juntar' (join) block with 'Tempo: ' and 'obter global tempoRestante' as inputs. A black arrow points from the 'comparar textos' block in the 'Blocos' panel to the 'comparar textos' block in the script. Another black arrow points from the 'comparar textos' block in the script to the 'obter seleção' block in the script.

# ADICIONAR DIFICULDADE

O procedimento inteiro para seleção do nível de dificuldade na [ListaSuspensa1](#) deve ficar assim:



## ADICIONAR DIFICULDADE

Agora que temos a variável `nivelSelecionado` configurada corretamente, devemos utilizar esta informação no procedimento de iniciar o jogo (`quando BotaIniciar.Clique`) para alterar os valores de `tempoRestante` e Intervalo do `Temporizador1` de acordo com a dificuldade selecionada.

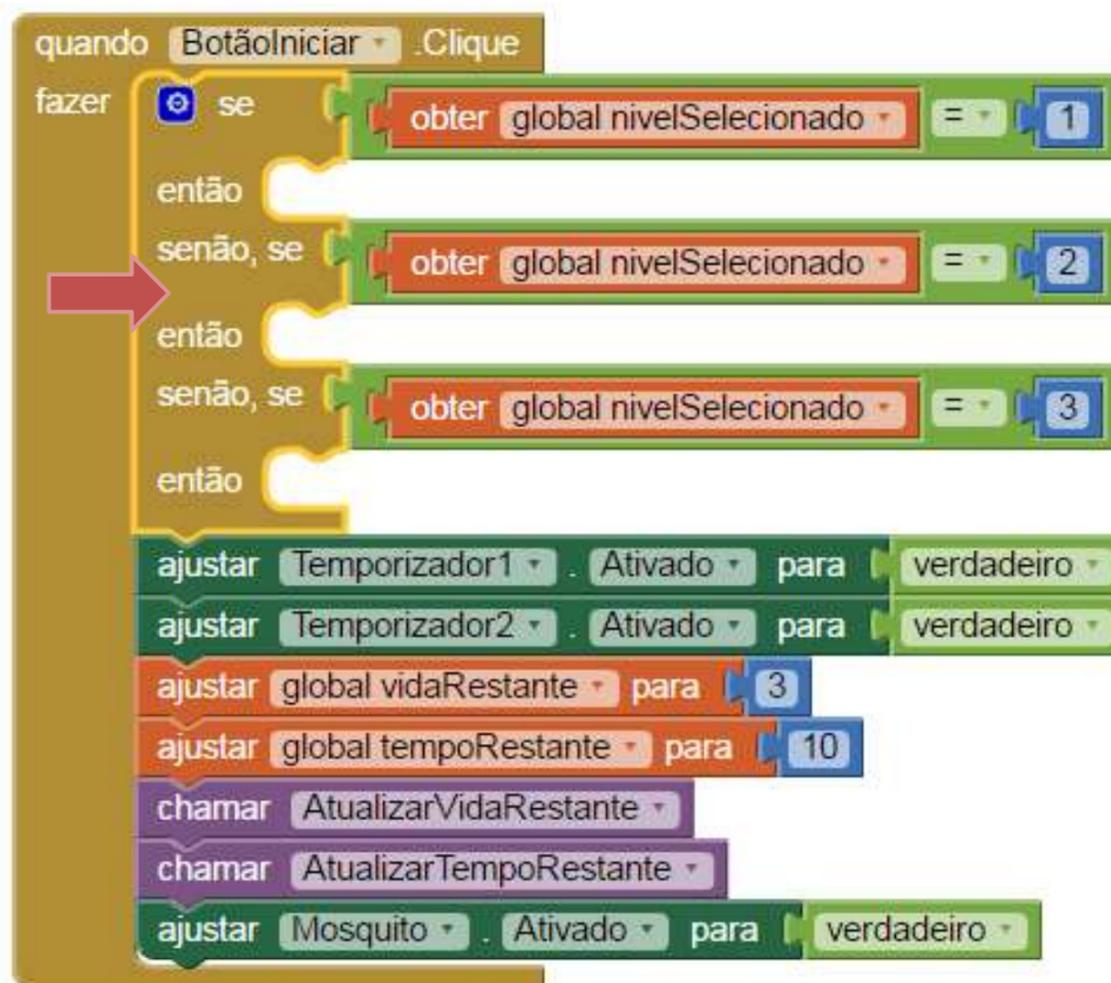
### DICA

Lembrar que o `Temporizador1` é o responsável por mandar o mosquito se mover.

# ADICIONAR DIFICULDADE

Primeiro devemos adicionar o mesmo bloco “Se, então” com dois blocos “Senão, Se”, igual realizamos no passo anterior.

Nesse caso, vamos usar operações Lógicas “=” para saber qual o valor do nivelSelecionado.



# ADICIONAR DIFICULDADE

Vamos definir as dificuldades de acordo com a seguinte regra:

## FÁCIL

tempoRestante = 20 s

Intervalo Temporizador1 = 500 ms

## MÉDIO

tempoRestante = 10 s

Intervalo Temporizador1 = 400 ms

## DIFÍCIL

tempoRestante = 8 s

Intervalo Temporizador1 = 350 ms

## DICA

Lembre que você pode duplicar os blocos para ser mais rápido.

```
quando BotãoIniciar .Clique
fazer
  se
    obter global nivelSelecioneado = 1
  então
    ajustar global tempoRestante para 20
    ajustar Temporizador1 . Intervalo para 500
  senão, se
    obter global nivelSelecioneado = 2
  então
    ajustar global tempoRestante para 10
    ajustar Temporizador1 . Intervalo para 400
  senão, se
    obter global nivelSelecioneado = 3
  então
    ajustar global tempoRestante para 8
    ajustar Temporizador1 . Intervalo para 350
  ajustar Temporizador1 . Ativado para verdadeiro
  ajustar Temporizador2 . Ativado para verdadeiro
  ajustar global vidaRestante para 3
  ajustar global tempoRestante para 10
  chamar AtualizarVidaRestante
  chamar AtualizarTempoRestante
  ajustar Mosquito . Ativado para verdadeiro
```

## ADICIONAR DIFICULDADE

A última coisa a fazer é excluir o bloco “ajustar global tempoRestante para 10” que já tínhamos.

```
quando BotãoIniciar .Clique
fazer
  se
    obter global nivelSelecioneado = 1
  então
    ajustar global tempoRestante para 20
    ajustar Temporizador1 . Intervalo para 500
  senão, se
    obter global nivelSelecioneado = 2
  então
    ajustar global tempoRestante para 10
    ajustar Temporizador1 . Intervalo para 400
  senão, se
    obter global nivelSelecioneado = 3
  então
    ajustar global tempoRestante para 8
    ajustar Temporizador1 . Intervalo para 350
  ajustar Temporizador1 . Ativado para verdadeiro
  ajustar Temporizador2 . Ativado para verdadeiro
  ajustar global vidaRestante para 3
  chamar AtualizarVidaRestante
  chamar AtualizarTempoRestante
  ajustar Mosquito . Ativado para verdadeiro
```

# HORA DE TESTAR!

Teste o seu aplicativo e verifique se os níveis de dificuldade estão realmente funcionando.

Agora o jogo deve estar funcionando perfeitamente!

Deste modo, finalizamos o desenvolvimento de todas as funcionalidades do jogo PegaMosquito.

