

AULA 4

Introdução à câmera,
iluminação e textura.

PLAY



CÂMERA

A câmera tem um papel importante na apresentação do jogo: ela representa o ponto de vista do qual o jogador consegue perceber o mundo do jogo e existem diversos tipos de perspectiva que podem ser adotadas. Uma das decisões que você deve tomar como designer é definir o que melhor se encaixa para o seu jogo.



CÂMERA

Câmera fixa: É a mais clássica e a primeira utilizada em jogos com elementos gráficos. Ela consiste simplesmente em uma tela de visualização que contém todos os elementos da cena e o jogador consegue visualizar tudo que ele precisa para interagir com essa porção específica do jogo. Ainda é utilizada até hoje, principalmente em jogos mais casuais, devido à facilidade de visualização (o jogador não precisa se preocupar em movimentação de câmera para jogar).



CÂMERA

Um outro exemplo de uso desse tipo de câmera são os jogos de terror mais antigos, como Alone in the Dark e Resident Evil 1, nos quais a câmera é usada para maximizar a experiência de terror do jogador, escolhendo ângulos que aumentassem o suspense e tensão da cena. No Plants vs Zombies, a câmera é fixa. Toda a ação ocorre na tela visualizada pelo usuário.



CÂMERA

Câmera com Scrolling ou rolagem:

Essa câmera é bastante utilizada até hoje em jogos de plataforma em 2D ou jogos do gênero Shoot'em Up. Normalmente, quando o movimento do personagem se aproxima dos limites ou bordas da tela, o cenário começa a se deslocar, permitindo que o jogador explore um cenário muito mais amplo do que a janela de visualização.



CÂMERA

Em jogos 2D, a câmera focaliza uma cena a partir de um ângulo específico. Tradicionalmente, nós temos a visão lateral, que chamamos de **side view**. Então podemos dizer que um jogo como Sonic ou Mario tem uma câmera Side-Scrolling, porque a visão é lateral e a câmera vai acompanhando o movimento do personagem



CÂMERA

Outra perspectiva bastante usada é a visualização de cima, como se estivéssemos observando o jogo de uma visão aérea. Essa câmera é chamada de top-down. O jogo de Shoot'em Up M.A.C.E. tem uma câmera Top-Down-Scrolling, porque a vista é de cima do cenário.



CÂMERA

Já em jogos 3D, existem duas perspectivas tradicionais que podem ser adotadas. **Câmeras em primeira pessoa** (First Person) são colocadas em um ponto do cenário onde o jogador está e representam uma visualização do mundo do jogo a partir dos olhos do personagem.



CÂMERA

Esse tipo de perspectiva proporciona um grande senso de imersão para o jogador, porque ele se sente na pele do personagem (ou ele mesmo é o personagem), e ela restringe a visualização apenas ao que o personagem consegue ver. Essa câmera é muito utilizada em jogos de ação que requerem mecânicas de precisão (como tiro), porque facilita a mira, ou jogos de aventura focados na exploração do mundo do jogo e interação com o cenário e objetos.



CÂMERA

É importante atentar para o mapeamento do controle da câmera: em controles de videogames, utiliza-se um dos eixos analógicos para movimento do personagem e outro para movimento da câmera. Em esquemas de mouse/teclado, é comum controlar a câmera com a movimentação do próprio mouse, enquanto a movimentação do personagem é feita através das teclas direcionais ou as teclas WASD.



CÂMERA

A segunda perspectiva bastante utilizada em jogos 3D é a **câmera em terceira pessoa** (Third Person). Nela a visualização é similar a uma câmera de cinema, que nos mostra uma cena do mundo do jogo, focada geralmente no personagem principal. A distância de visualização corresponde ao quanto longe do personagem a câmera estará localizada: dessa forma, uma câmera mais distante consegue exibir mais detalhes do mundo do jogo, porém em menor escala, enquanto uma câmera mais próxima exibe uma área menor, de forma mais detalhada.



CÂMERA

É uma boa prática permitir que o jogador possa escolher a distância (zoom) em qualquer momento, para que ele possa usar a câmera como instrumento de exploração do mundo do jogo. Em jogos de terceira pessoa, a visão do mundo do jogo e das ações dos personagens podem proporcionar experiências fantásticas para os jogadores!



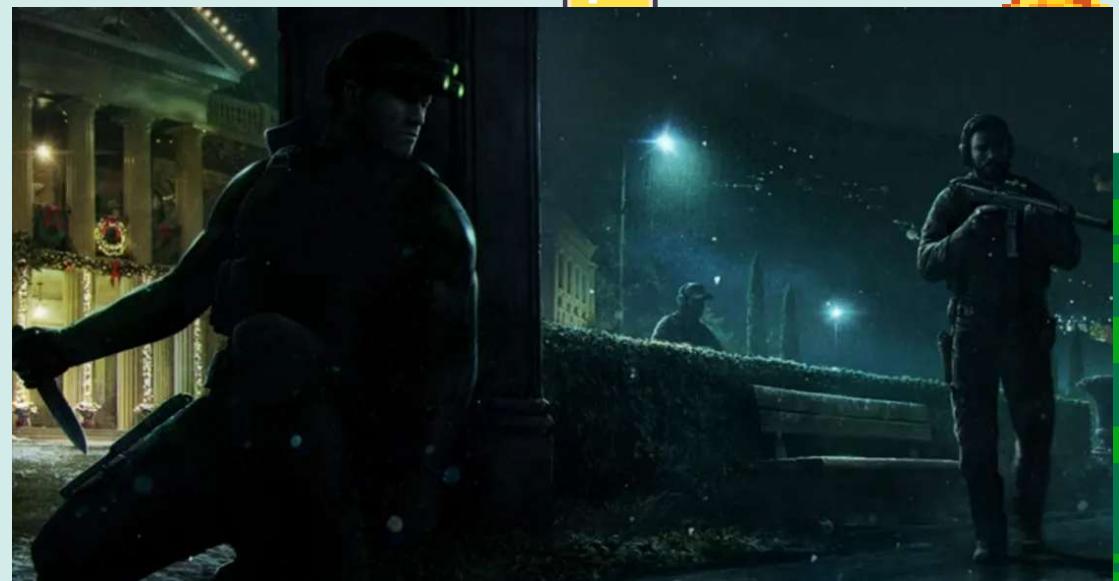
CÂMERA

Alguns jogos se aproveitam de projeções especiais para dar uma visão diferenciada ao jogo: perspectivas como a isométrica (conhecida como **pseudo 3D ou 2.5D**) utilizam técnicas de projeção que, através de artefatos 2D, dão uma ilusão de 3D para o jogador. Essas perspectivas são muito utilizadas em jogos de estratégia e em jogos de RPG que possuem grupos de personagens, porque facilita a gestão de todos os elementos.



ILUMINAÇÃO NOS JOGOS DIGITAIS

A luz pode ser entendida como parte de um conjunto de elementos desenvolvidos para facilitar a imersão do jogador naquela experiência.



ILUMINAÇÃO NOS JOGOS DIGITAIS

Neste sentido, pode ser um elemento chave no caso de gêneros ação–aventura como survival-horror, First Person Shooter (FPS) e stealth, em que a atmosfera normalmente é escura e de saturação e brilho baixo, com o intuito de criar determinada atmosfera ou ambiência.



ILUMINAÇÃO NOS JOGOS DIGITAIS

Com o evoluir da tecnologia, em especial da capacidade de processamento e armazenamento e da memórias das placas gráficas (placas de vídeo) dos computadores e consoles de videogame, não apenas os sons e gráficos se desenvolveram, como também novas possibilidades de exploração narrativa— o que gerou uma demanda por maior e melhor representação gráfica de cenários, objetos de cena e personagens.



ILUMINAÇÃO NOS JOGOS DIGITAIS

Os jogos 3D, principalmente os voltados para computadores e consoles, ampliou-se a gama de possibilidades técnicas de iluminação virtual. Tornou-se assim possível, simular e melhor controlar a luz por meio de ferramentas especialmente desenvolvidas para esse fim e disponibilizadas nos softwares de modelagem e animação tridimensionais.



ILUMINAÇÃO NOS JOGOS DIGITAIS

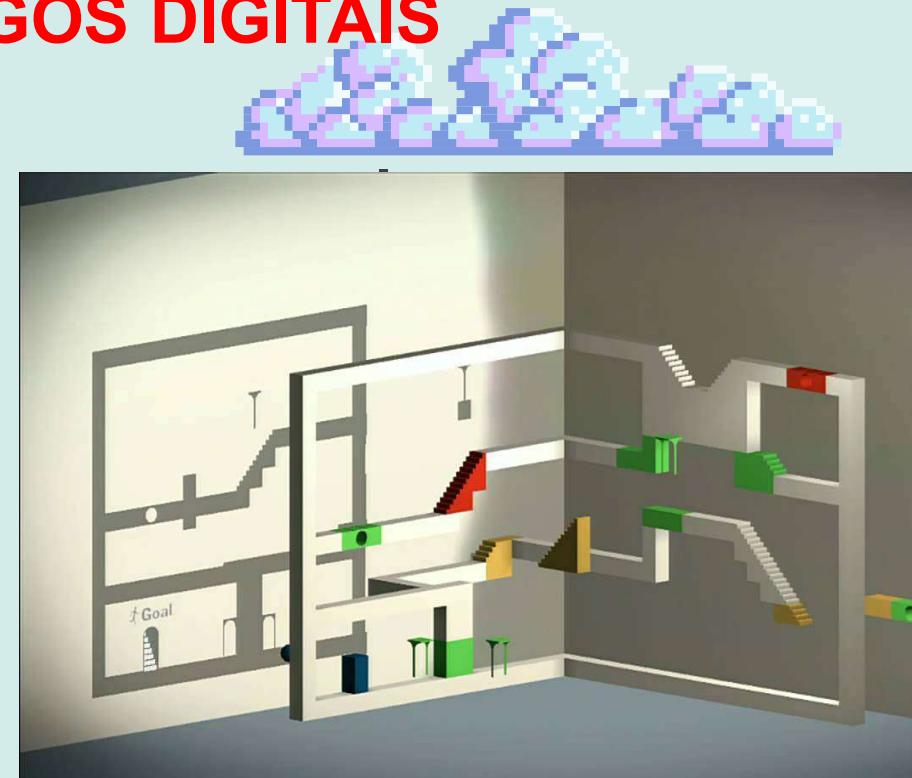
Os jogos dos gêneros ação–aventura stealth, tem elementos de mecânica de jogo à relação luz - sombra. Na série de jogos "Thief" (Square Enix 1998-2004) a percepção dos NPCs inimigos é diretamente afetada e reduzida enquanto o jogador permanecer nas partes escuras dos cenários, como nas sombras produzidas por prédios ou objetos.



Review Thief - 2014

ILUMINAÇÃO NOS JOGOS DIGITAIS

Produzido em 2008 pela JAPAN Studio, Echochrome foi um jogo que inovou ao trazer o uso de perspectiva como ferramenta para a progressão de fases. Em sua sequência, Echochrome 2, produzido em 2010, o estúdio buscou novamente uma proposta de inovação ao gameplay, incorporando luz e sombra como elemento básico para a progressão do jogo



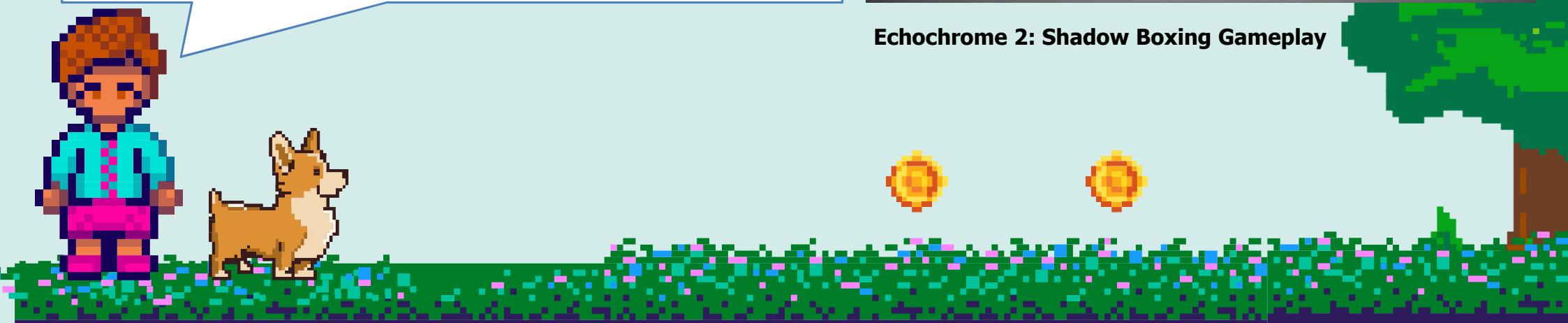
Echochrome 2: Shadow Boxing Gameplay

ILUMINAÇÃO NOS JOGOS DIGITAIS

Neste jogo luz e sombra funcionam em simbiose com a mecânica do jogo, de forma que os papéis que elas representam dentro do jogo não podem ser substituídos por qualquer outro tipo de elemento, estabelecendo dessa forma uma relação de indissociabilidade entre forma e função do jogo.



Echochrome 2: Shadow Boxing Gameplay



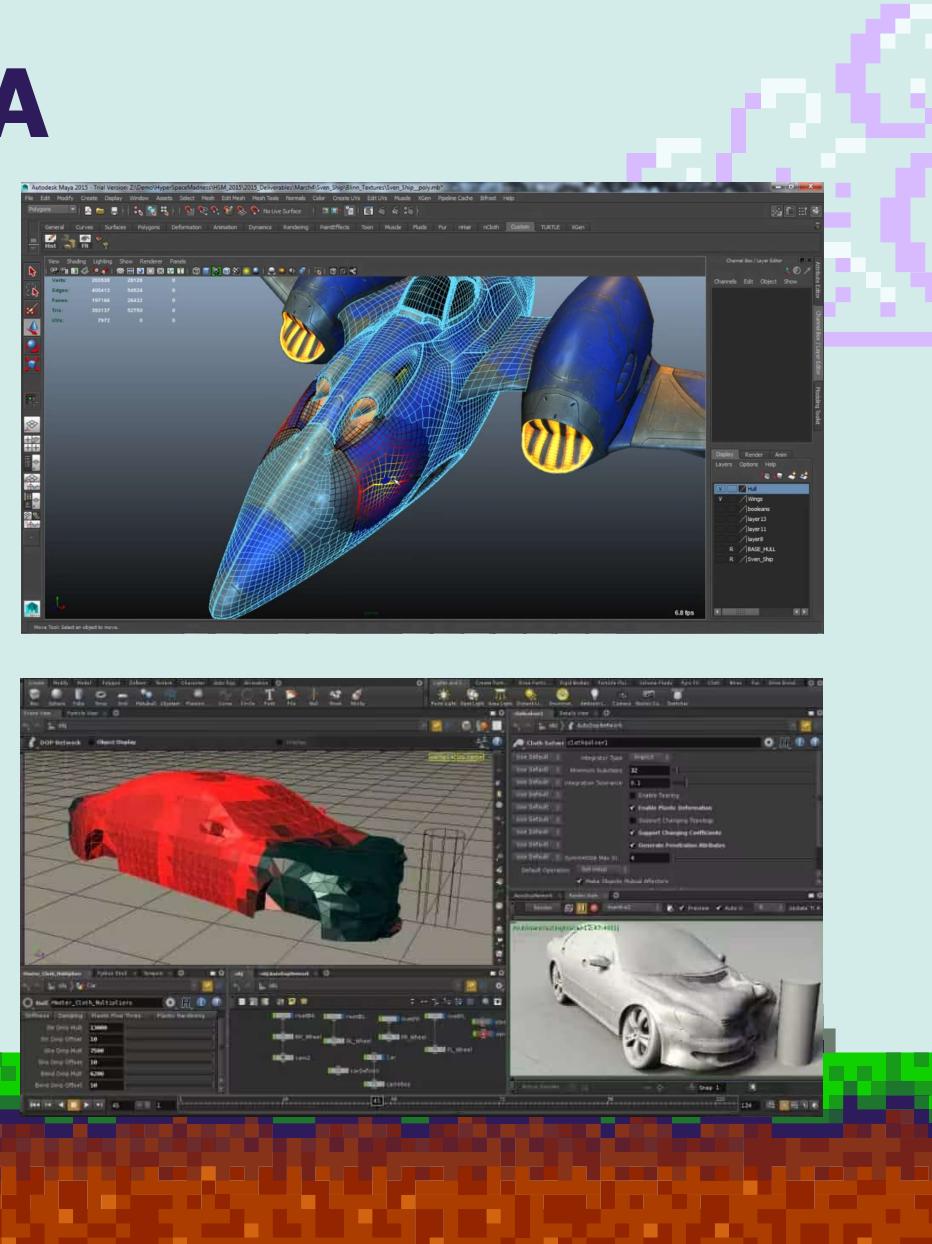


O que são?

Texturas são basicamente imagens, que servem de superfície para modelos. O que isto quer dizer? Basicamente, que elas são responsáveis por dar o visual esperado a determinados elementos que, de outra forma, talvez não fossem reconhecidos como o que se pretende mostrar.



TEXTURA





O que são?

Por si só, elas podem definir o grau de clareza e o quanto os visuais de um jogo são agradáveis. Texturas ruins podem resultar em gráficos que dão a impressão de estar borrados ou ultrapassados em tecnologia. Em casos extremos, podem até mesmo tornar difícil o entendimento do que exatamente é aquele item.

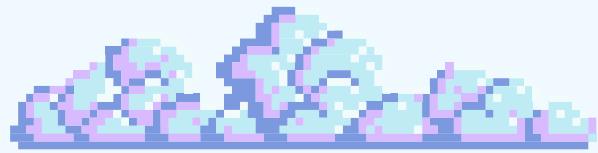


TEXTURA



Tom Holland / Spider-man 3D Model (Real-Time) -
Finished Projects - Blender Artists Community





TEXTURA

Onde são encontradas?

Em tudo. Cenários, personagens, itens... enfim, tudo o que se vê na tela possui texturas para dar a aparência de um determinado ser ou objeto ao que seria, de outra forma, apenas um modelo.



The Hunter: Call of the Wild

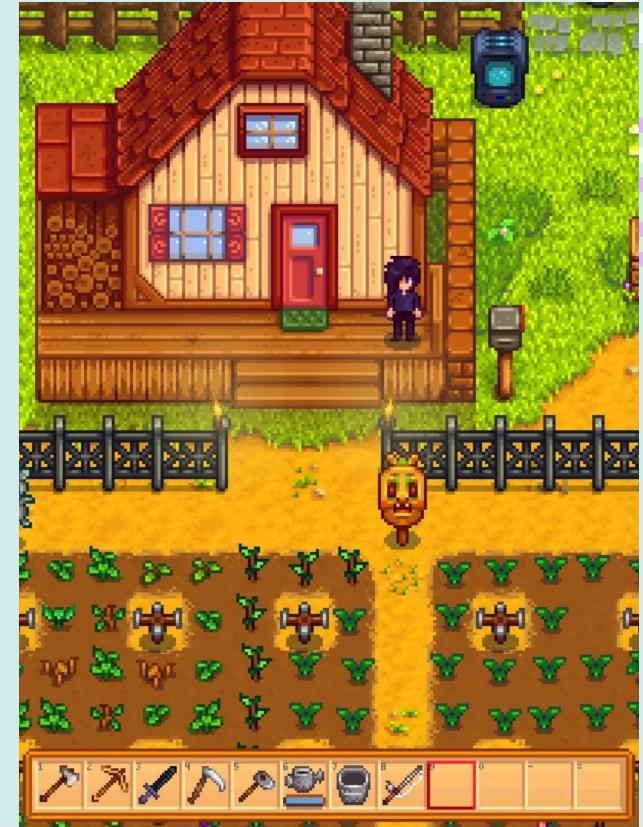


A evolução

Atualmente, as plataformas são capazes de gerar imagens com muitos detalhes, o que permite que as texturas sejam muito bem trabalhadas. Mas nem sempre foi assim. A quantidade de texturas e seu detalhamento são diretamente proporcionais à capacidade dos sistemas que rodam jogos, o que fez com que elas fossem um dos aspectos que mais evoluídos nos video games.



TEXTURA



Stardew Valley



O que é tileset?

Um tileset é um conjunto de texturas reunidas numa mesma imagem, uma composição. Estas texturas formam as peças gráficas que compõem um cenário de um videogame: pisos, paredes, escadas, telhados, etc. Para criar este conjunto, usamos uma grade composta de quadrados do mesmo tamanho. Neste caso, 16x16px. Cada quadrado desta grade é chamado de tile, a menor parte da composição.



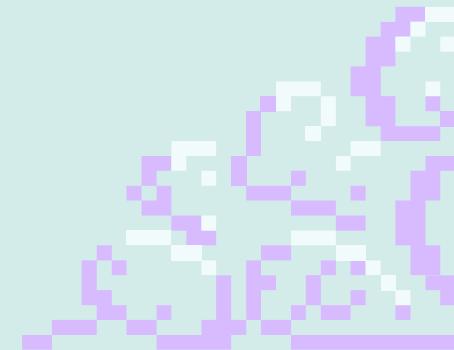
TEXTURA



Tiles



TEXTURA



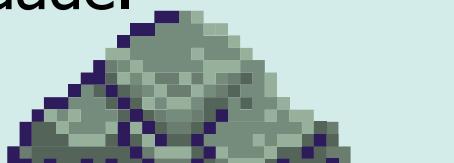
Criação de tilesets

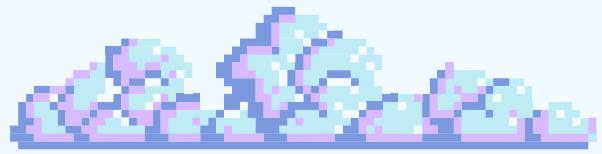
Pyxel Edit

Este software se destaca pela simplicidade e facilidade de uso, além das ferramentas para criar e animar tilesets. Além disso, o preço é acessível.

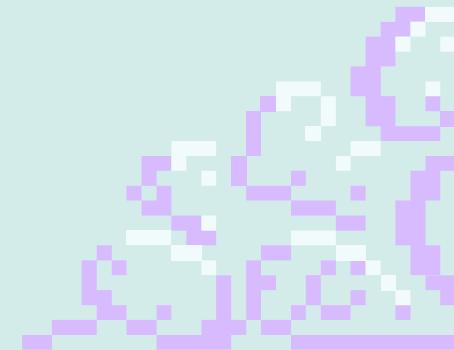
Aseprite

Um dos melhores programas do mercado, utilizado por diversos profissionais, seu design é excelente e está sempre recebendo atualizações. Embora seja mais caro do que o Pyxel Edit, vale o investimento, pois oferece mais recursos com mais simplicidade.





TEXTURA



Criação de tilesets

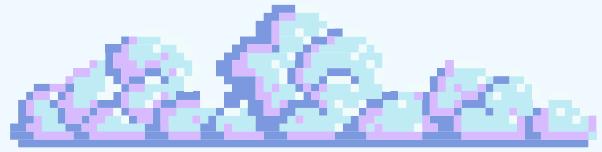
Photoshop

Se você já possui uma licença do Photoshop, pode criar os seus tilesets por ele, pois oferece diversas opções avançadas para a criação de diversos elementos de pixel art.

Recursos extras

Se você quer criar um tileset, mas não sabe por onde começar, recomenda-se o site [Itch.io](#). Nele você encontrará tilesets já prontos e com preços baixos. Além disso, é possível encontrar arquivos gratuitos.





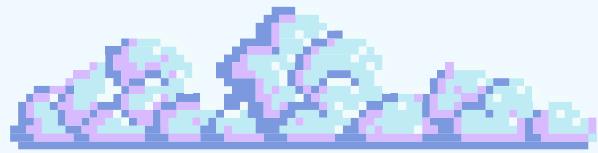
O que é um sprite?

Um sprite é uma imagem ou um objeto gráfico de duas dimensões em um jogo, que pode ser tanto personagem quanto objetos adicionados em um plano de fundo. Para os personagens, imagine que você queira criar os movimentos deles em seu jogo, como subida de mãos, corrida, pulo, entre outros. Nesse caso, tratamos que o sprite é o personagem e, para todos estes movimentos, devemos montar as variações de sprite.



TEXTURA





TEXTURA

Beleza não é tudo

Estas talvez sejam as funções principais das texturas. Afinal de contas, desde que ele era formado por poucos pixels nós podíamos perceber que Mario usava um macacão e boina. No entanto, só após a evolução das plataformas é que pudemos distinguir os botões amarelos segurando o macacão e o M na boina. Detalhes, mas que enriquecem a experiência.



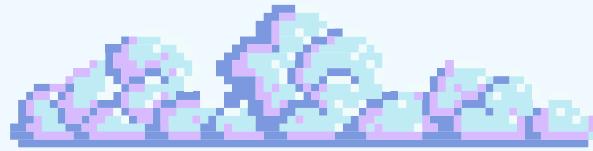


TEXTURA

Beleza não é tudo

Além disso, algumas vezes seria difícil reconhecer determinados inimigos se não houvessem texturas. Imagine a imagem abaixo sem elas, como reconhecer um Smoker no meio de outros tantos zumbis em Left 4 Dead? Talvez ao fazê-lo já fosse tarde demais. Neste sentido, as texturas têm a capacidade de tornar cada elemento único, destacando-o dos demais.





TEXTURA

Essencial

Como você pode ver, texturas são uma parte crucial dos games, sem as quais a maioria dos títulos se tornaria praticamente impossível de jogar – ou, caso contrário, não seriam nada divertidos e muito frustrantes visualmente.



Stray



SOBRE O GAMEMAKER



The image shows a screenshot of the GameMaker website. At the top, there is a navigation bar with the GameMaker logo, social media links (@ and GitHub), and a blue "BAIXAR" button. Below the navigation bar, a large banner features a cartoon character wearing a hat and vest, sitting on a rock. The banner text includes "A melhor engine de criação de jogos 2D" and "CRIE SEU JOGO". A subtitle below the banner states, "O GameMaker facilita a criação de jogos, independentemente do nível de habilidade. Tudo que você precisa é de uma ideia!" A blue "REGISTRAR" button is located at the bottom left of the banner. The background of the page has a pixelated, colorful theme with clouds and flowers.

A melhor engine de criação de jogos 2D

CRIE SEU JOGO

O GameMaker facilita a criação de jogos, independentemente do nível de habilidade. Tudo que você precisa é de uma ideia!

REGISTRAR

GameMaker™ Exposição Tutoriais Comunidade Blog

@ BAIXAR

DOWNLOAD GAMEMAKER GRÁTIS

DOWNLOAD GAMEMAKER FOR FREE



Windows



Mac

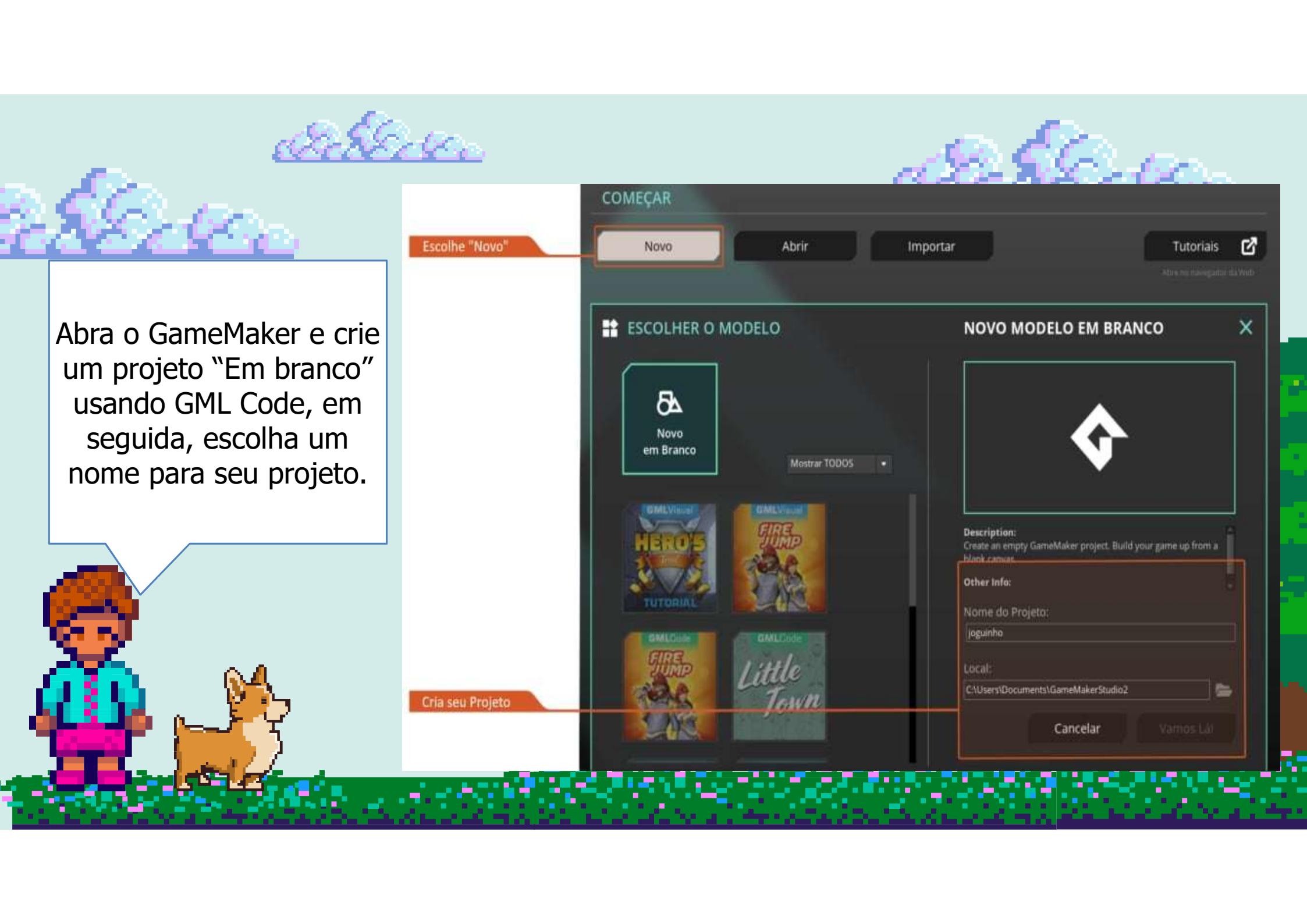


Ubuntu (Beta)

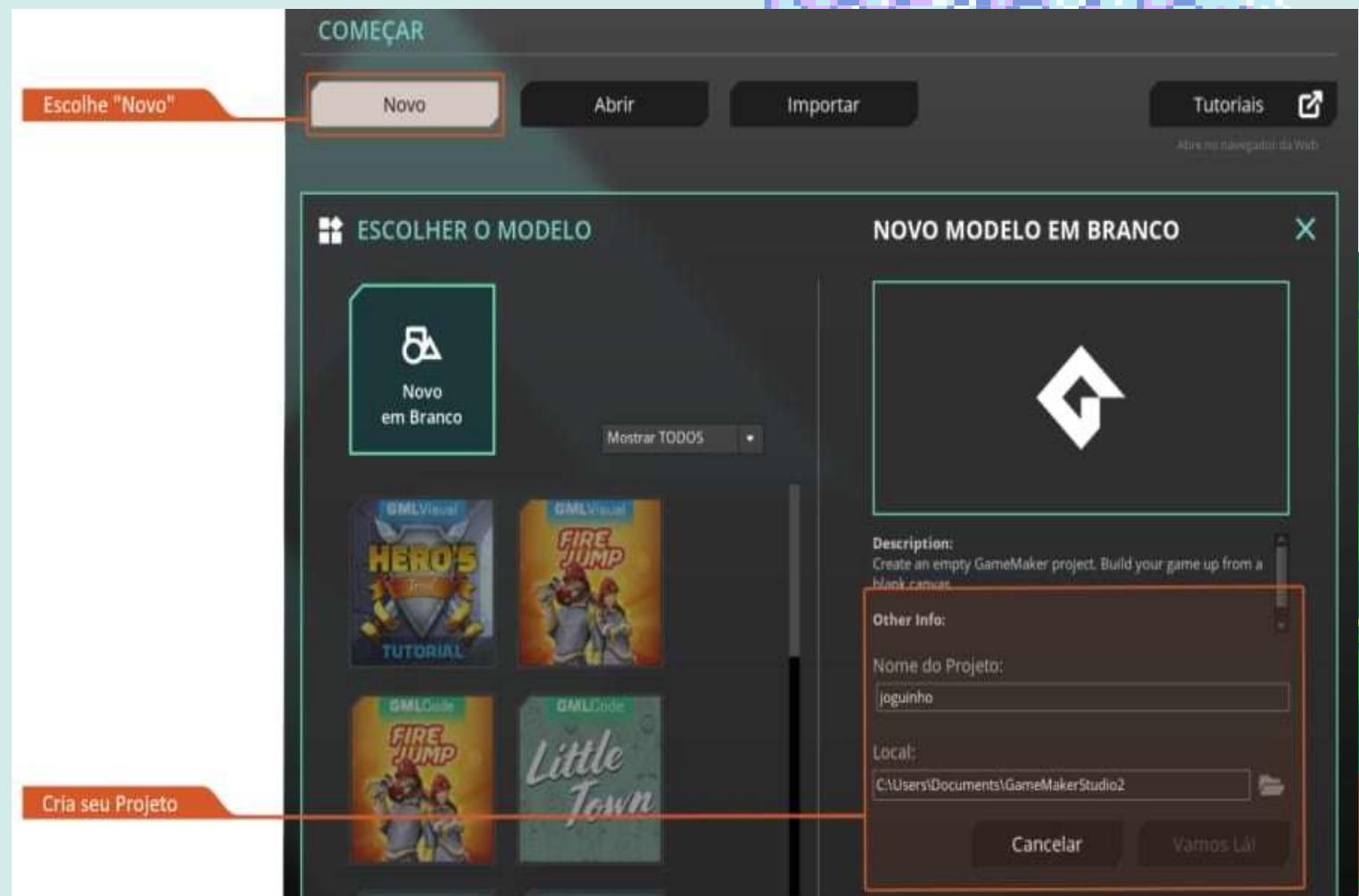
[DOWNLOAD](#)

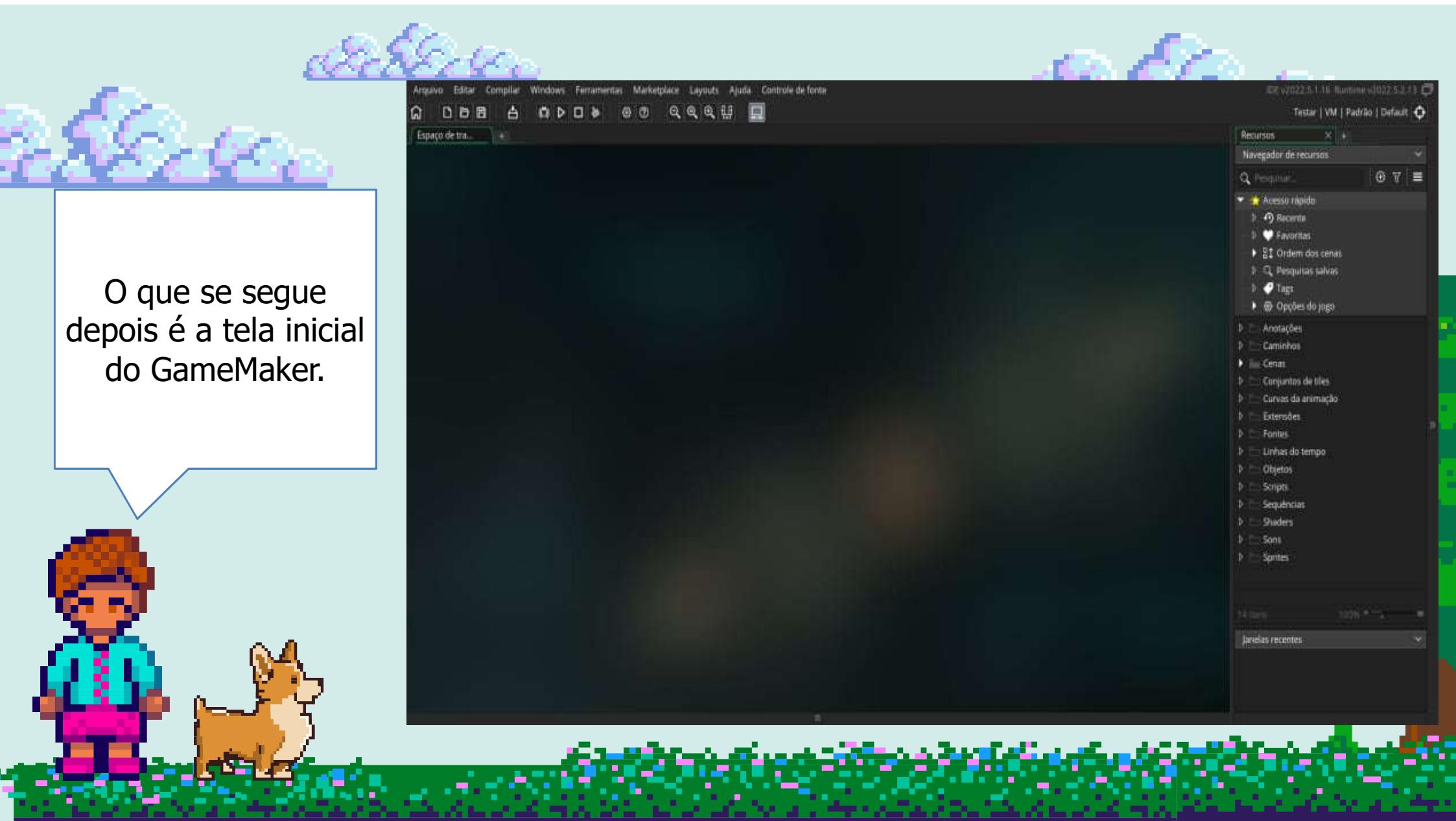
[DOWNLOAD](#)

[DOWNLOAD](#)

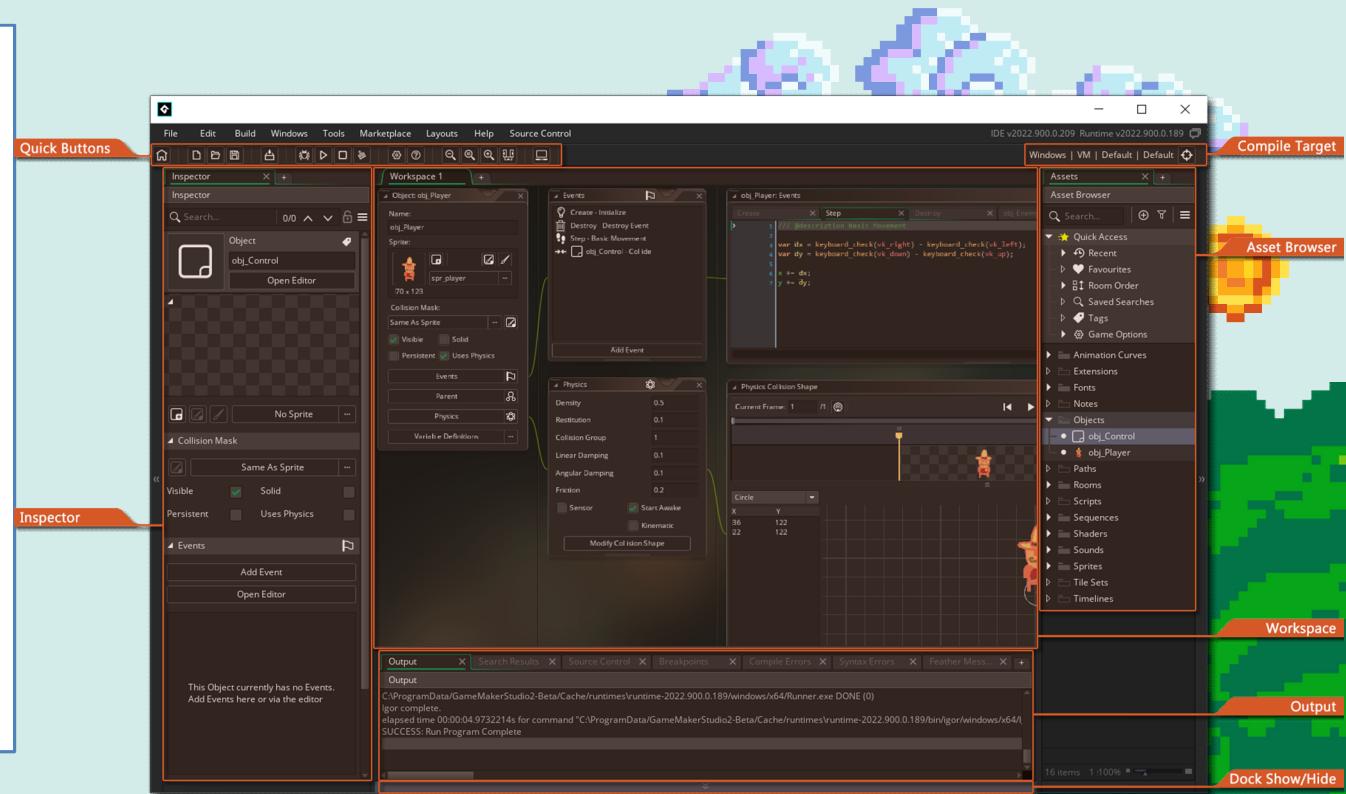


Abra o GameMaker e crie um projeto “Em branco” usando GML Code, em seguida, escolha um nome para seu projeto.





Após fazer o login e iniciar um novo projeto, o GameMaker o levará ao espaço de trabalho inicial com algumas janelas básicas acopladas à IDE. Em geral, o workspace é simplesmente uma área onde você pode organizar o diferente assets para seu jogo enquanto você está trabalhando.



BLANK GAME_3 - GameMaker

Arquivo Editar Compilar Windows Ferramentas Marketplace Layouts Ajuda Controle de fonte

O topo do GameMaker IDE é ocupado pela barra de menu. Os menus padrão disponíveis são: Menu Arquivo, edição, copilar, Windows, Ferramentas, Marketplace, Menu de Layouts, Ajuda e controle de fonte.



A janela do Inspector mostra todas as propriedades associadas com os elementos selecionados no IDE. Você pode selecionar um ou vários elementos em qualquer editor, e visualizar ou alterar suas várias propriedades.

The screenshot shows the Construct 3 IDE interface. On the left, there's a white box containing text about the Inspector window. Below this box is a pixel art character with orange hair and a teal jacket, standing next to a small brown dog. A blue line points from the character towards the Inspector window. The Inspector window itself has a dark background with orange borders around its title bar and tabs. It displays the properties for an object named "oPlayer". The "Events" tab is selected, showing options like "Create", "Begin Step", "Step", and "End Step". To the right of the Inspector is a large code editor window titled "Workspace 1 > oPlayer: Events > Step". The code is written in Construct's internal scripting language:

```
1 // @description Movement logic for player
2
3 // Move towards target
4 if (x < global.playerX && CTRL_G.gameIsActive) {
5   moveX = 0.2;
6 } else {
7   moveX = 0;
8 }
9
10 // Jump
11 if (iJump && collision(0, 1)) {
12   moveY = -jumpSpeed;
13
14   var _snd = audio_play_sound(sndJump, 1, 0);
15   audio_sound_pitch(_snd, random_range(0.8, 1.2));
16 }
17
18 // Parent event
19 event_inherited();
20
21 // Collect flag
22 var _flag = instance_place(x, y, oFlag);
23
24 if (_flag && !_flag.collected && CTRL_G.gameIsActive) {
25   with (_flag) {
26     x = other.x;
27     collected = true;
28     event_user(0);
29   }
30
31   CTRL_G.flags++;
32 }
```

The code editor has a status bar at the bottom showing "1/60 Col:43 Ch:43" and "INS".

O núcleo de seu jogo será criado a partir de assets adicionado ao Navegador de Ativos localizado por padrão à direita da IDE. Aqui é onde você pode adicionar tudo que seu jogo requer para rodar, incluindo um jogo room, sprites, objects, paths e uma série de outras coisas. Um jogo básico no GameMaker exigirá um room para rodar (novos projetos serão sempre criados com um room asset já criado), e normalmente pelo menos um object e um sprite, embora você provavelmente usará muito mais!

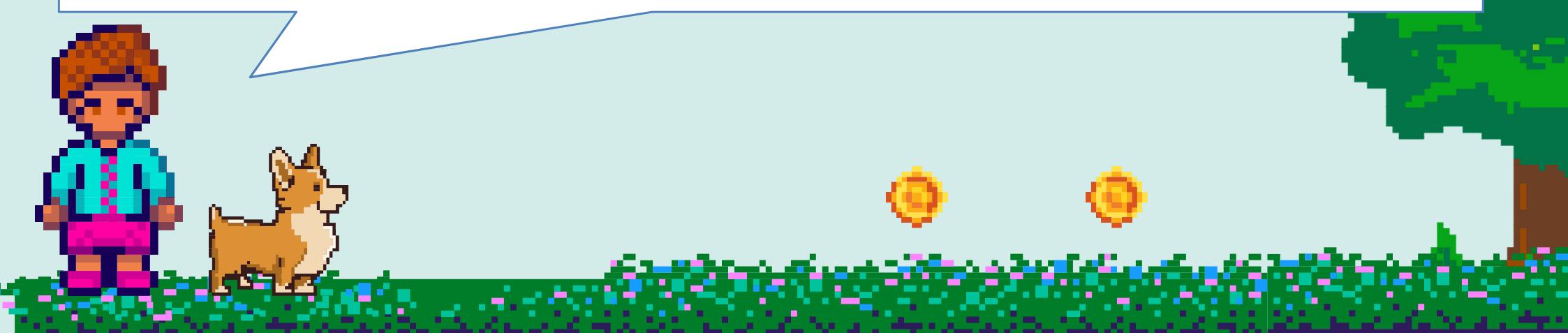


Compilação

Compilação seu jogo pode significar uma de duas coisas: compilá-lo para testes, ou compilá-lo para criar uma final pacote executável para uma plataforma alvo específica. Esta página tem como objetivo explicar detalhadamente ambas as opções.

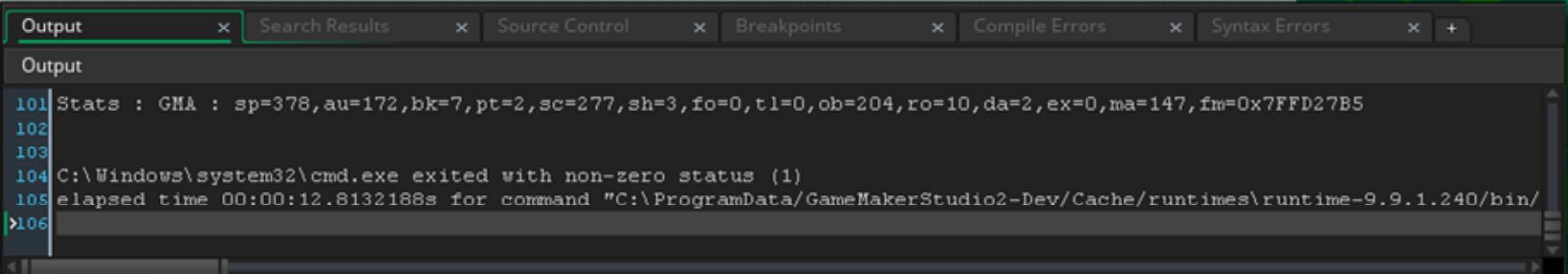
Depuração

Ao programar seu jogo, seja usando GML Code ou o GML Visual, pode ser muito fácil cometer erros - usar as variáveis erradas, passar os argumentos errados ou usar as funções erradas são apenas alguns dos erros mais comuns que todos nós cometemos - e estes erros nem sempre são detectados pelo verificador de sintaxe que está incluído no editor de roteiro/ação.



A Janela de Saída

Quando você abre um projeto no GameMaker pela primeira vez, você será apresentado com a Janela de Saída encaixada na parte inferior da tela. Esta janela acoplada contém várias abas que mostram as diferentes informações de saída para seu projeto, dependendo de certas circunstâncias.



```
Output
Output
101 Stats : GMA : sp=378,au=172,bk=7,pt=2,sc=277,sh=3,fo=0,tl=0,ob=204,ro=10,da=2,ex=0,ma=147,fm=0x7FFD27B5
102
103
104 C:\Windows\system32\cmd.exe exited with non-zero status (1)
105 elapsed time 00:00:12.8132188s for command "C:\ProgramData/GameMakerStudio2-Dev/Cache/runtimes\runtime-9.9.1.240/bin/
106
```

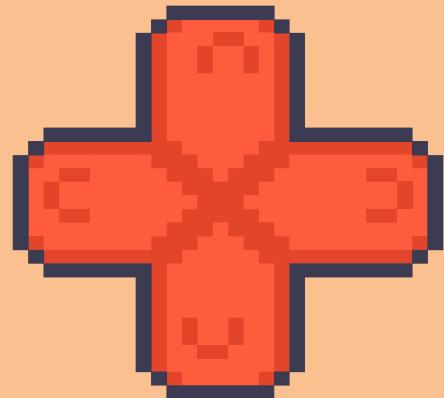


O Marketplace é o mercado online do GameMaker para compra e venda de pacotes asset. Um pacote asset pode ser praticamente qualquer coisa relacionada ao GameMaker, ou seja: sprites, scripts, shaders, ou até mesmo mecanismos ou estruturas completas de jogos. Para poder criar, comprar e vender pacotes, você deve primeiro estar registrado no Marketplace como um editor e ter se registrado através do GameMaker (isto será automático quando você entrar no programa) ou a partir da página principal do Marketplace. Você pode encontrar mais informações sobre como se tornar um editor nas páginas vinculadas na parte inferior desta página.

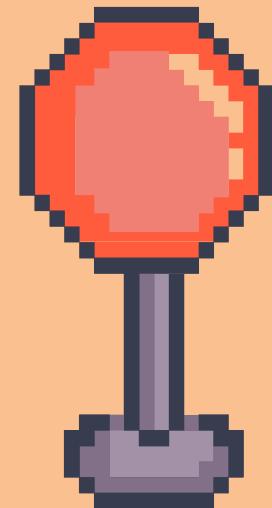
The screenshot shows the GameMaker Marketplace homepage. At the top, there's a navigation bar with the GameMaker logo and links for Mostruário, Tutoriais, Comunidade, and Blogue. Below the navigation, there are two sections of asset cards:

- Modelos e motores de jogos**:
 - Rolando 'sh...** Projetos US\$ 5,99 US\$ \$ 2,99 US\$ -50%
 - Bata neles...** Projetos US\$ 14,99 -28%
 - Texas Hold'e...** Projetos US\$34,99 US\$ \$24,99 US\$ -28%
 - Blackjack C...** Projetos US\$ 20,99 US\$ \$ 24,99 US\$ -16%
- Efeitos e cores legais**:
 - dynamic puddles** Efeitos US\$ 1,99 -100%
 - Baseado em** Objetos US\$ 5,99 GRATIS -100%
 - Cor Fácil...** Sombreadores LIVRE
 - Cor para** Outro LIVRE

EXERCÍCIO 3



MEDIUM
LEVEL



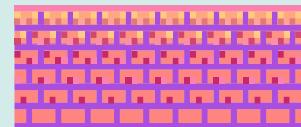
START



TUTORIAL GAME MAKER



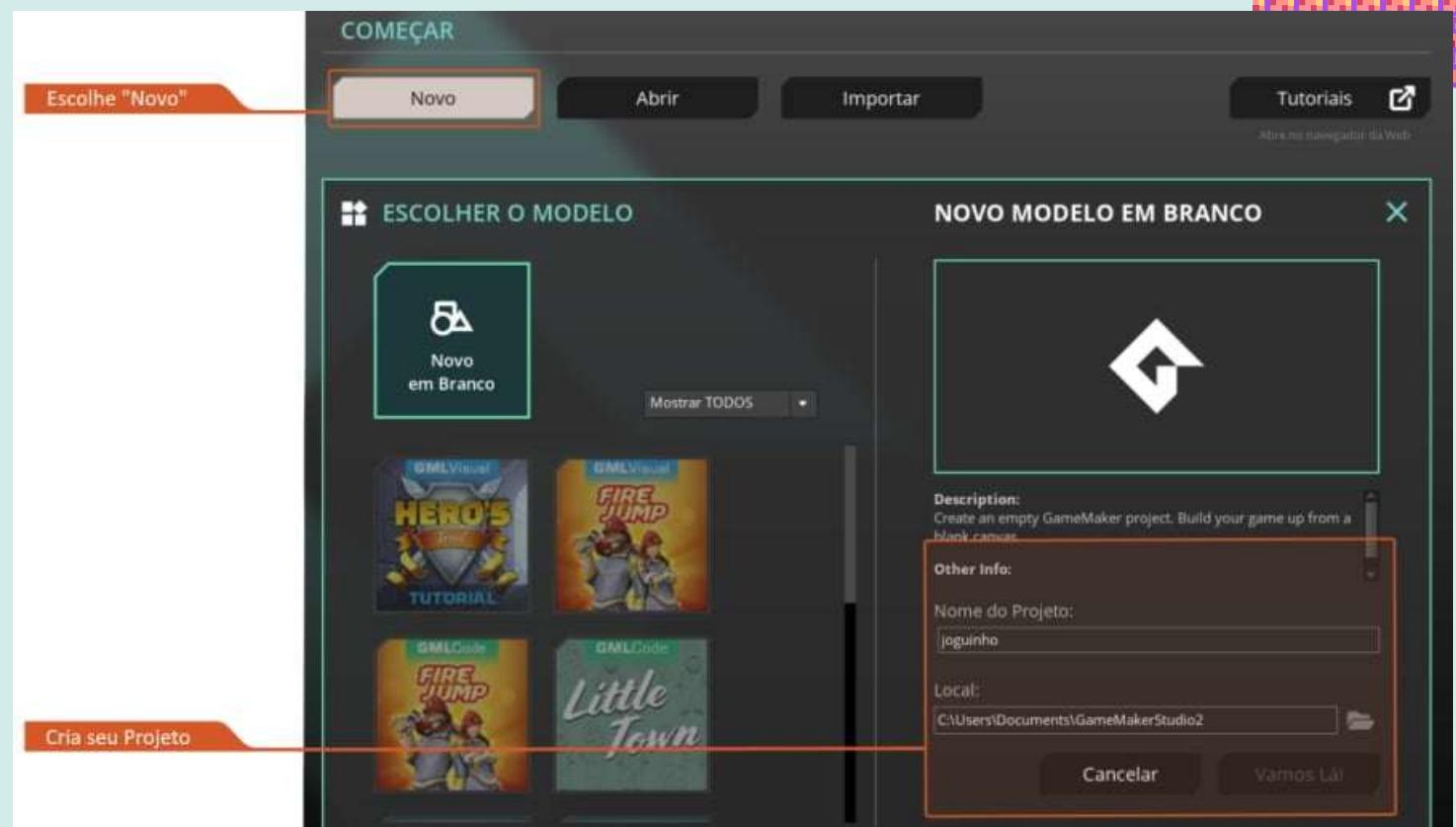
The image shows a promotional banner for GameMaker. On the left, there's a black rectangular area containing the GameMaker logo and the text "CRIE SEU PRIMEIRO JOGO COM GUSELECT". Below this, it says "Tutorial para Iniciante". A yellow arrow points from the bottom right of this text area towards a screenshot of a game level on the right. The screenshot depicts a 2D platformer scene with a blue sky, green trees, and a purple ground. A character is jumping over a gap between two platforms. A small orange enemy is visible nearby. The overall aesthetic is retro and colorful.



TUTORIAL GAME MAKER

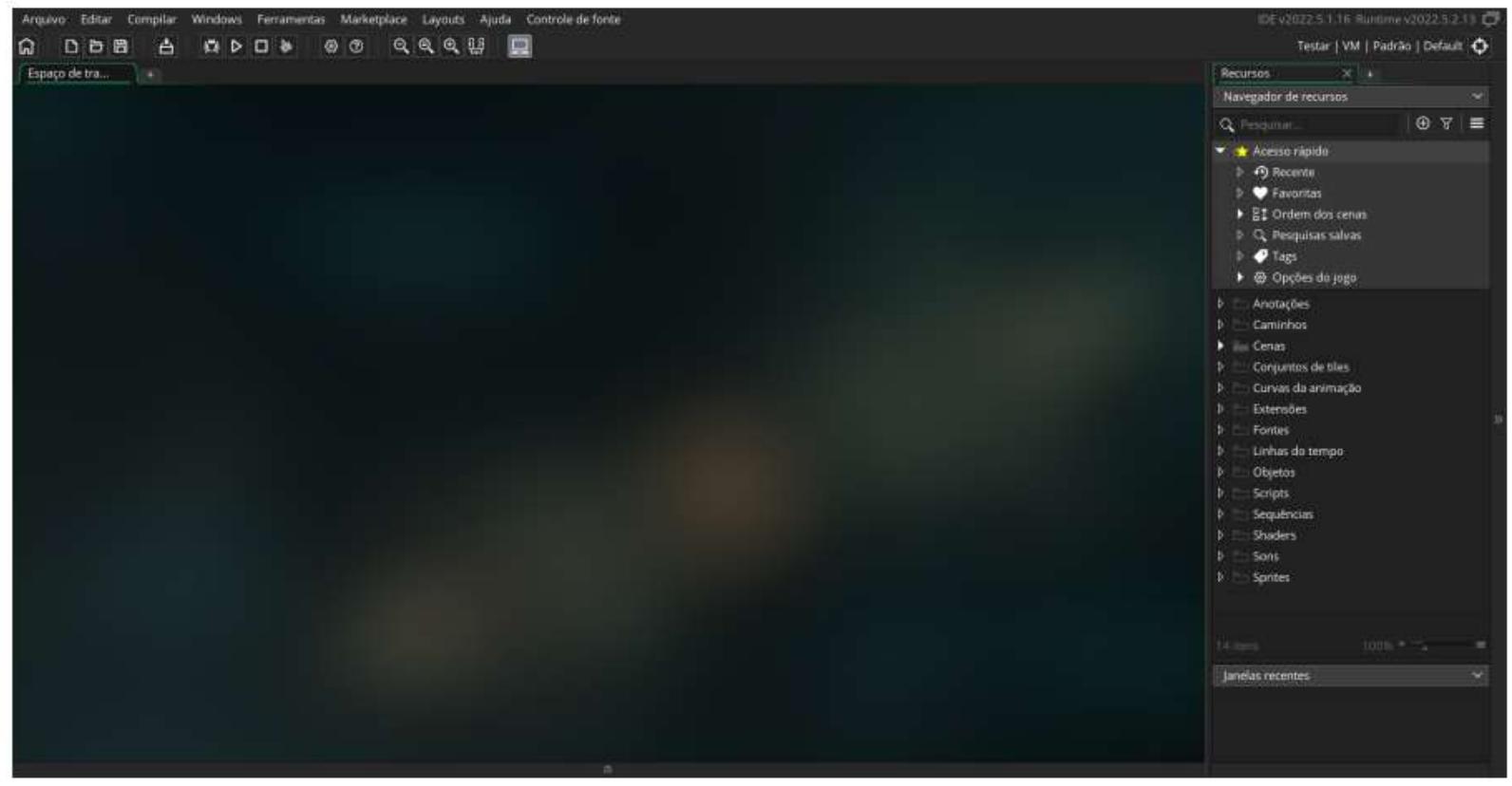


1 – Crie um novo projeto de jogo.



TUTORIAL GAME MAKER

O que se segue depois é a tela inicial do GameMaker.



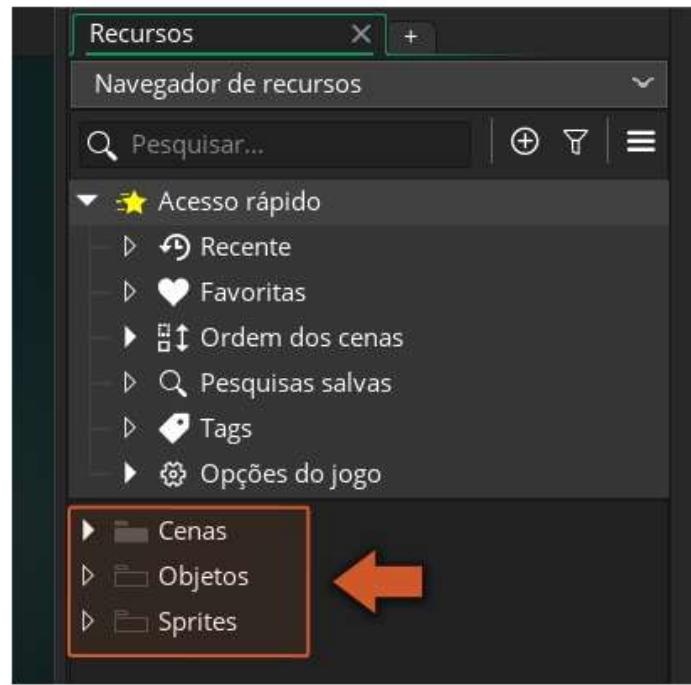
2 – Tela Inicial GameMaker

TUTORIAL GAME MAKER

3 – Apague as outras pastas, deixe apenas: Cenas, objetos e Sprites.

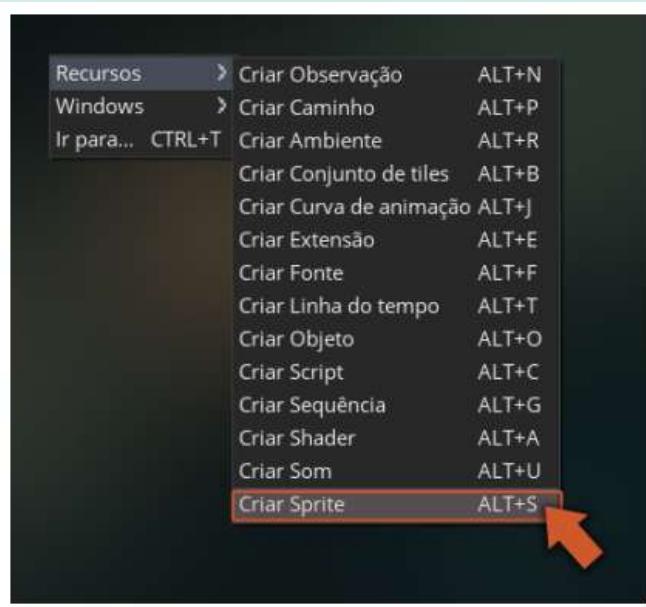
Ao lado direito da tela, perceba o “Navegador de Recursos”, onde ficam os recursos do jogo: sprites, sons, objetos e etc...

Neste projeto você só irá utilizar três tipos de recursos: Sprites, Objetos e Cenas.

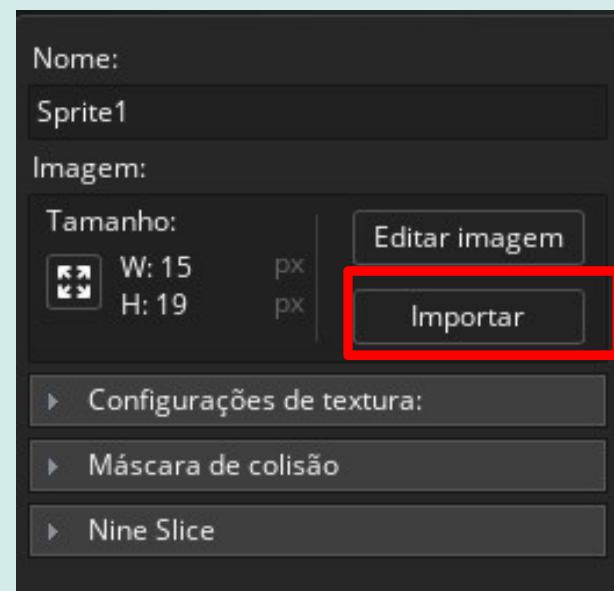


TUTORIAL GAME MAKER

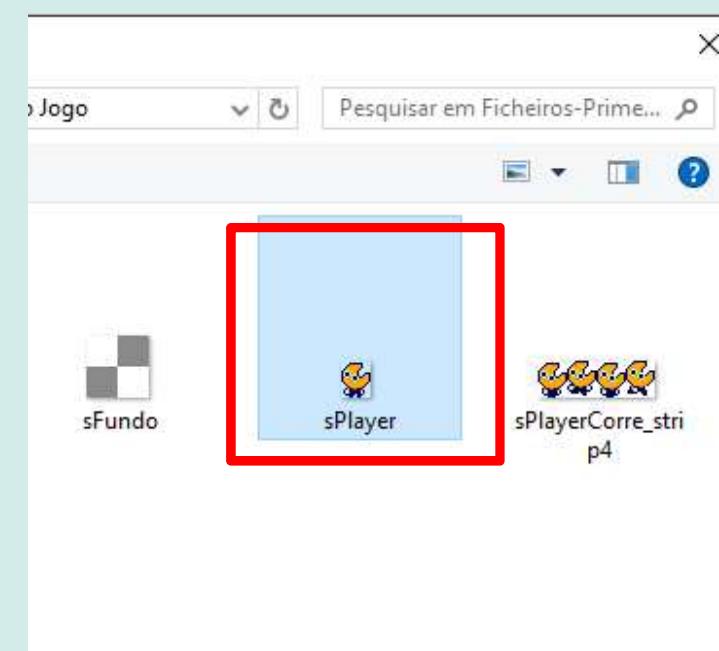
4- Crie Sprite



5- Importe Sprite

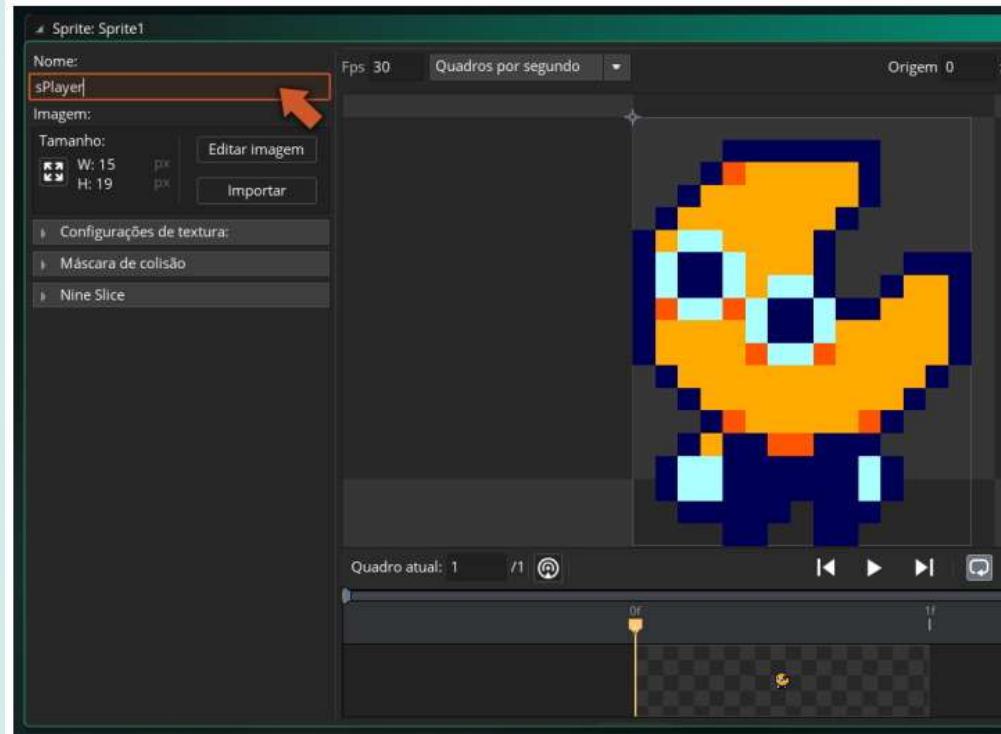


6 - Escolha-o.

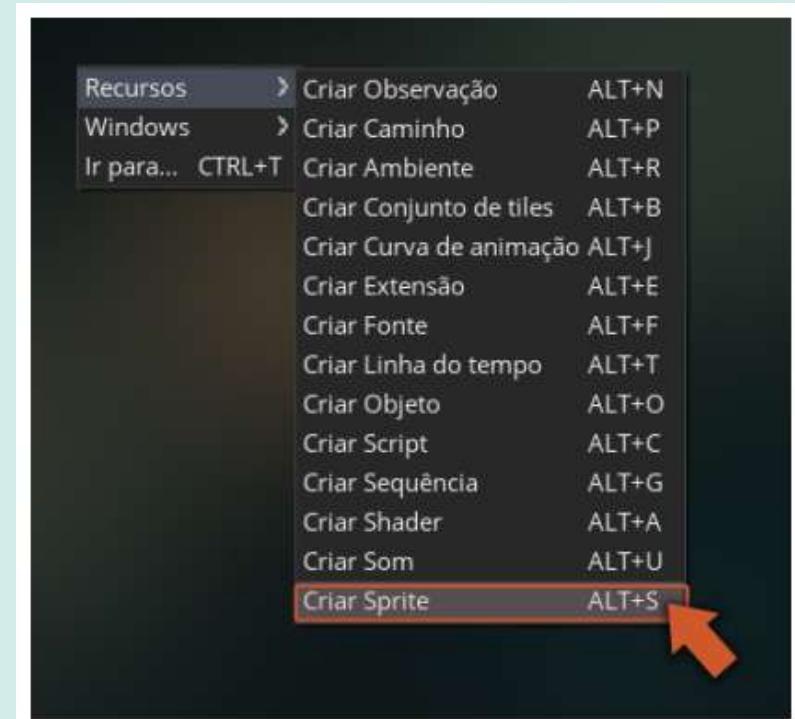


TUTORIAL GAME MAKER

7-Dê nome de sPlayer

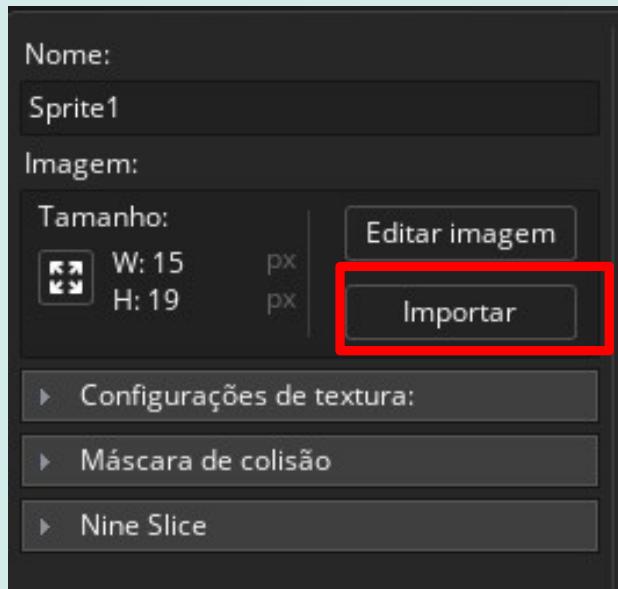


8- Crie outro Sprite



TUTORIAL GAME MAKER

9 - Importe Sprite



10 - Escolha-o



TUTORIAL GAME MAKER

O que é Strip?



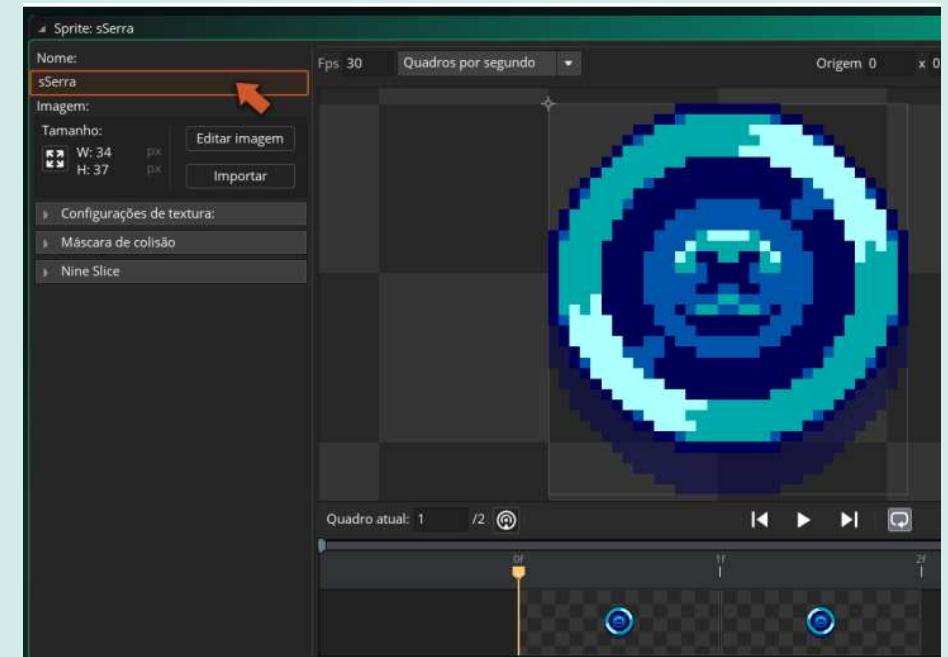
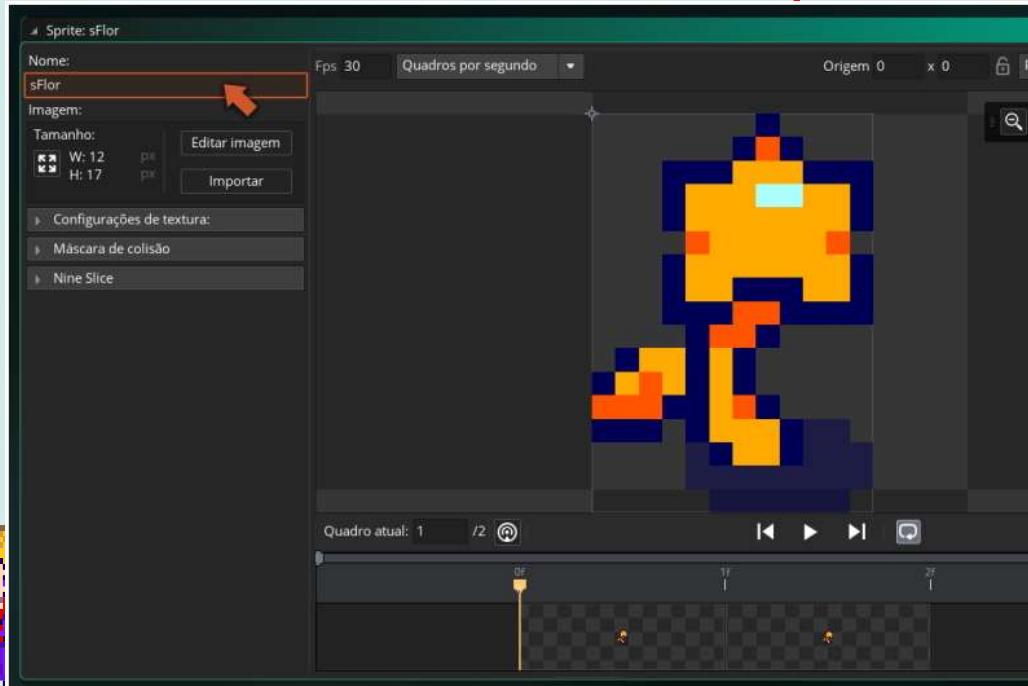
A imagem da faixa acima contém 5 quadros. Seu nome de arquivo é **PlayerSprite_strip5.png**, indicando que possui 5 frames, que o GameMaker usa para dividi-lo em subimagens separadas. O nome de um arquivo de imagem de tira deve terminar com " _stripN ", onde " N " é o número de quadros da animação.



TUTORIAL GAME MAKER

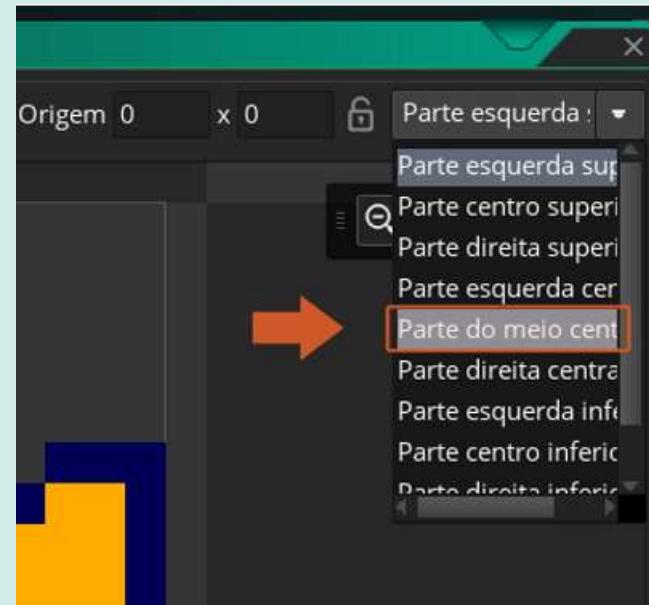


11 – Dê o nome de sFlor e faça os mesmo passo a passo para a Sprite sSerra



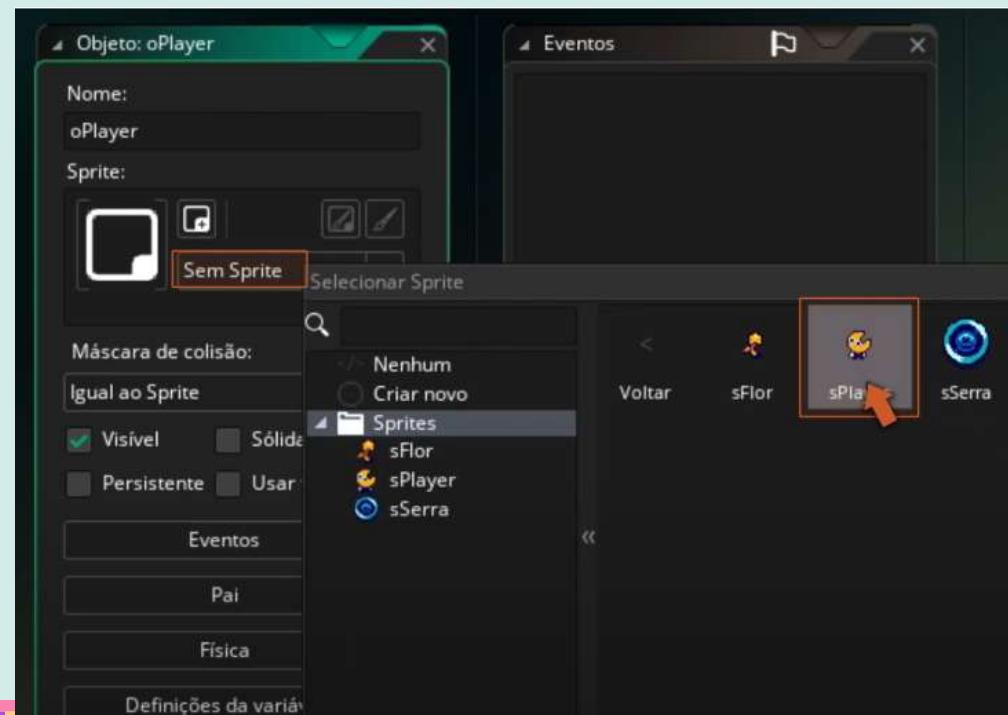
TUTORIAL GAME MAKER

12 – Ajuste o ponte origem para todos os Sprites.



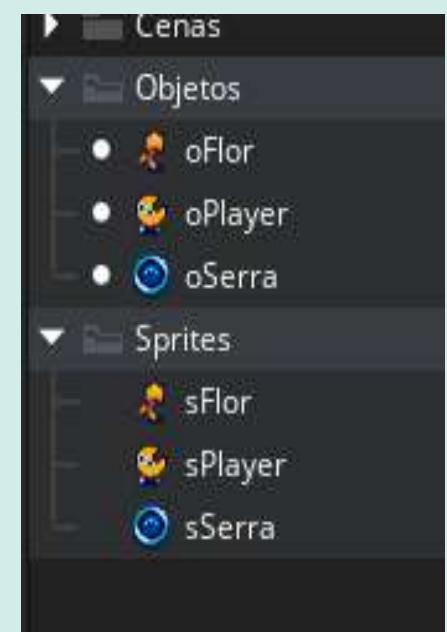
TUTORIAL GAME MAKER

13 – Agora iremos criar um Objeto para cada Sprite do jogo.



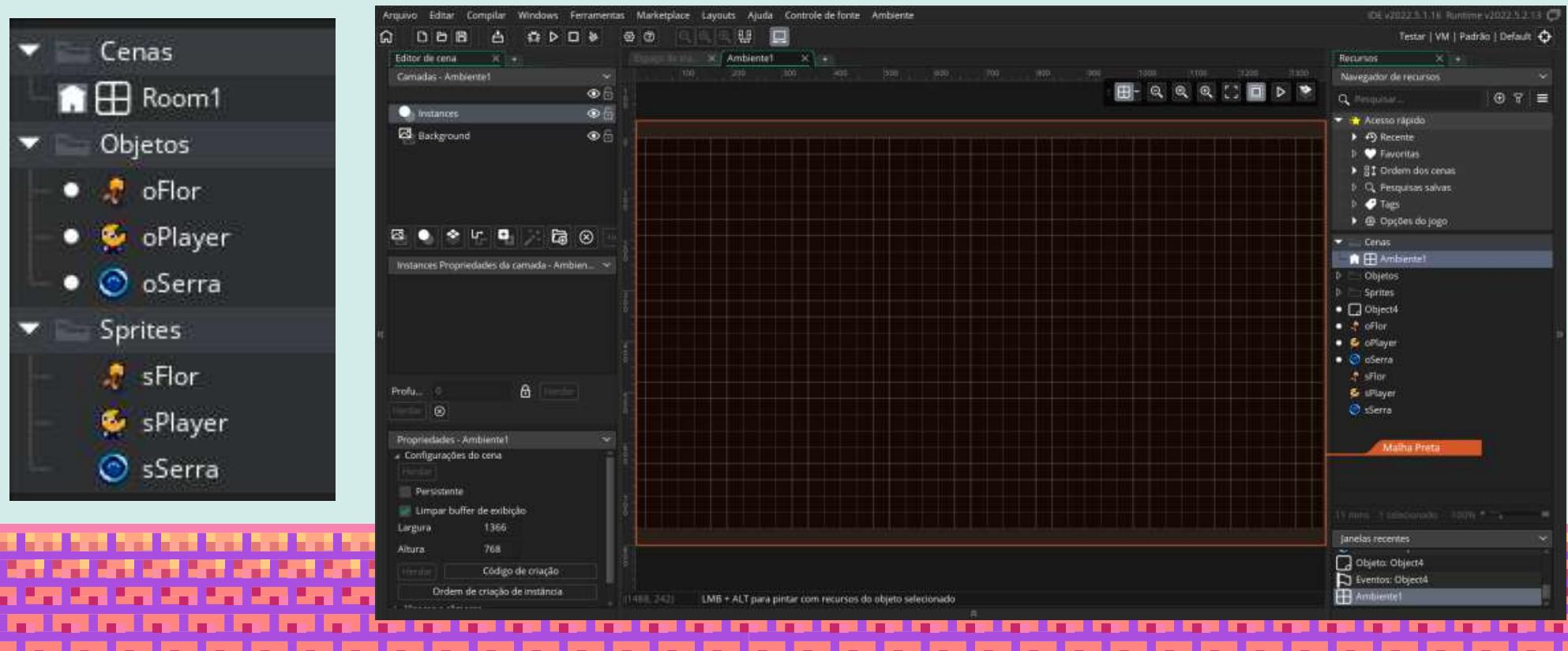
TUTORIAL GAME MAKER

14 – Faça o mesmo para a “oFlor” e a “oSerra”, nunca esquecendo de colocar nos objetos, seus respectivos Sprites.



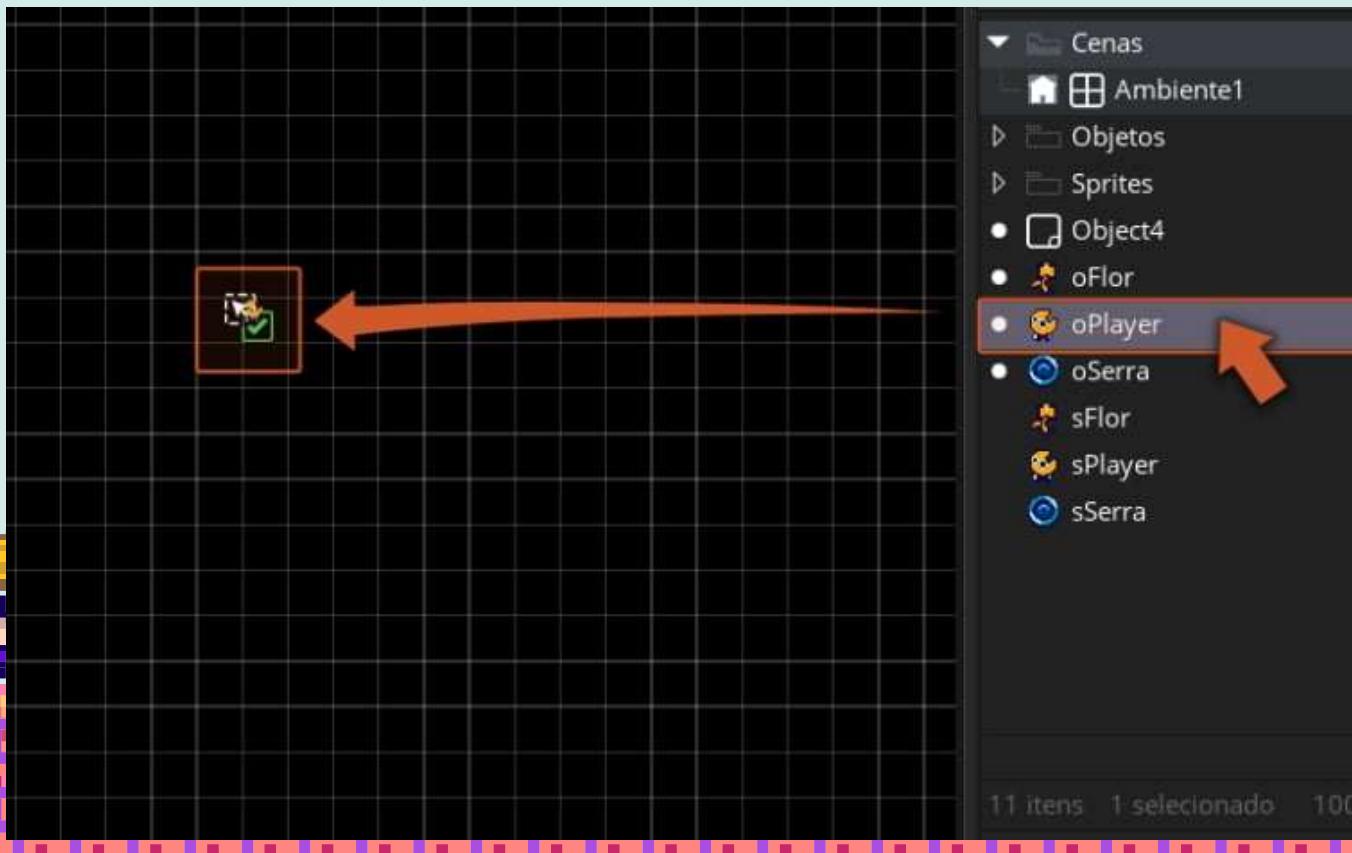
TUTORIAL GAME MAKER

15 – CRIAR E EDITAR O NÍVEL - Navegador de Recursos, vá em “Cenas” e clique no Ambiente 1 (Room 1).



TUTORIAL GAME MAKER

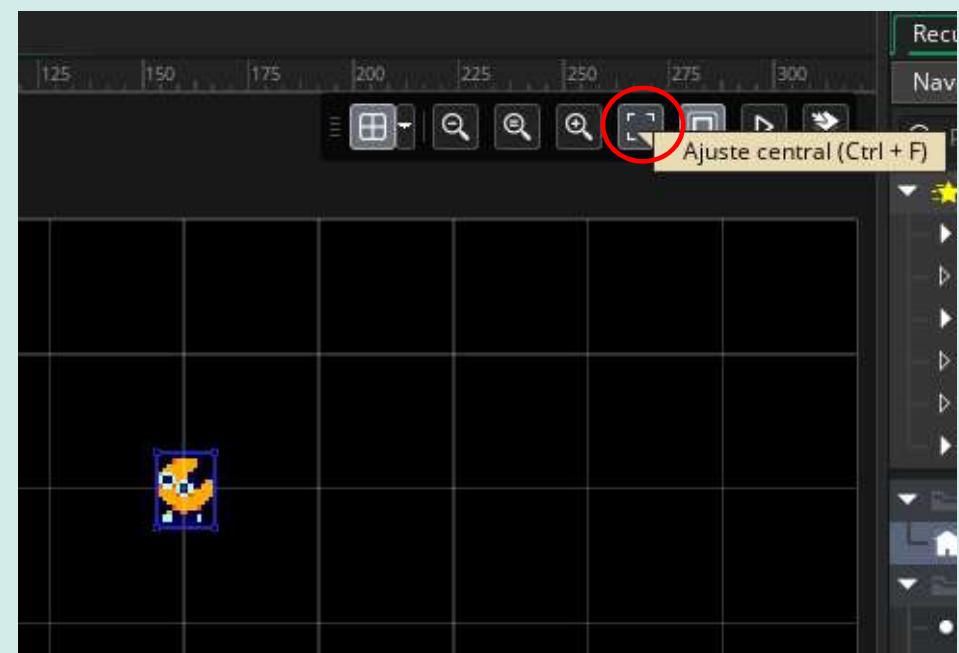
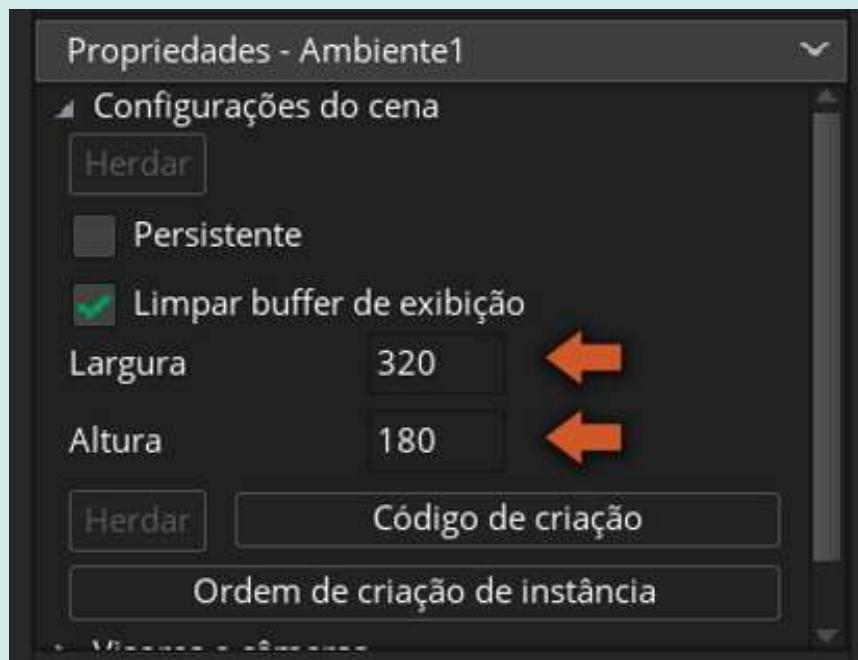
16 – Agora clique no “oPlayer” com o botão esquerdo do mouse e o arraste para a fase segurando “ALT”.



O “oPlayer” está bem pequeno. Isso é porque a nossa fase está muito grande. Você pode alterar isso no canto inferior esquerdo da tela, nas propriedades dessa cena.

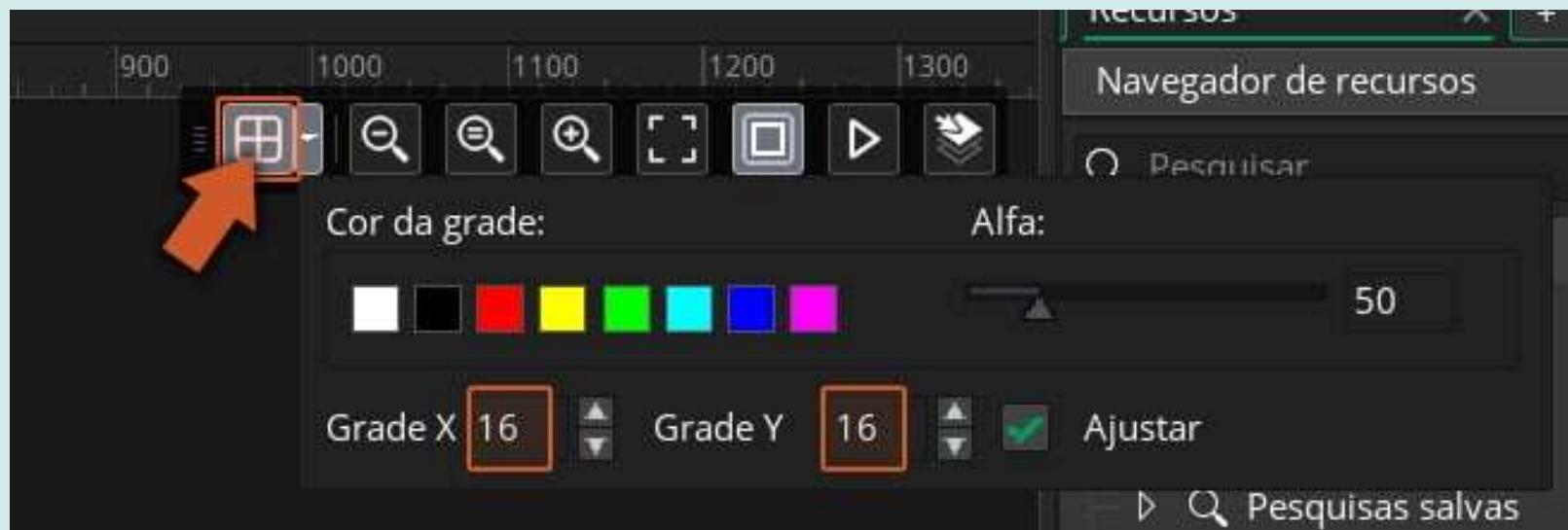
TUTORIAL GAME MAKER

17 – Na “largura” coloque 320 e na “altura” 180 e você pode alterar o zoom, clique em “Ajuste central” para ajustar a zoom da tela automaticamente.



TUTORIAL GAME MAKER

18 – Altere o tamanho da grade para 16 por 16 pixels.



TUTORIAL GAME MAKER

19 – Agora pode colocar outros objetos na cena. Segurando ALT e clicando com o botão esquerdo do mouse, coloque algumas serras e flores pela fase.



TUTORIAL GAME MAKER

20 – Mas ao apertar F5 para testar nosso jogo, ele está super pequeno e nosso personagem não consegue se mover...



TUTORIAL GAME MAKER

21 – CRIAR EVENTOS: Vá para a janela do “oPlayer”, clique em “Adicionar Evento” e “Criar”.



TUTORIAL GAME MAKER

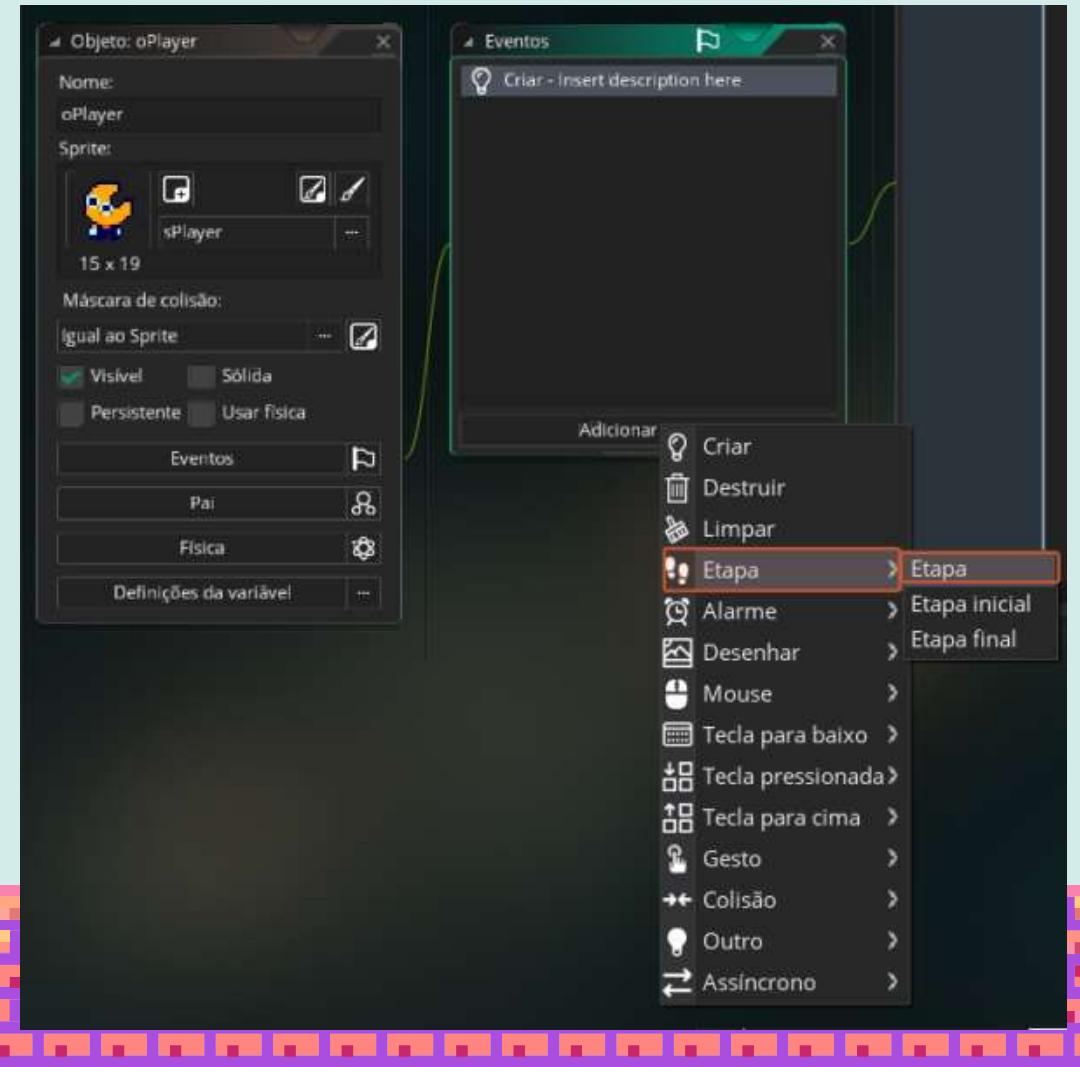
22 – Queremos que a janela do jogo seja 1280 por 720 pixels, então vou colocar esses valores aqui nessa função.

```
window_set_size(1280,720)
```

TUTORIAL GAME MAKER

23 – FAZER O PLAYER MOVER:

Agora faremos a movimentação do Player. Queremos que ao apertar a tecla de cima, ele ande 1 pixel para cima, ao apertar a de baixo, ele vai ande 1 pixel para baixo. Para a esquerda, 1 pixel para a esquerda, para a direita, 1 pixel para a direita.





TUTORIAL GAME MAKER



24 – Dentro de “Etapa”, escreva:

```
if keyboard_check(vk_up)
{
}
```



Este código significa: se (tradução de “if”) apertarmos a tecla de cima, a área que tá dentro do “{ }” acontece. Essa função vai checar se alguma tecla foi apertada, no caso “vk_up”.



TUTORIAL GAME MAKER



25 – Dentro de “Etapa”, escreva:

```
if keyboard_check(vk_up)
{
    y-=1
}                                1 píxel para cima

if keyboard_check(vk_down)
{
    y+=1
}                                1 píxel para baixo

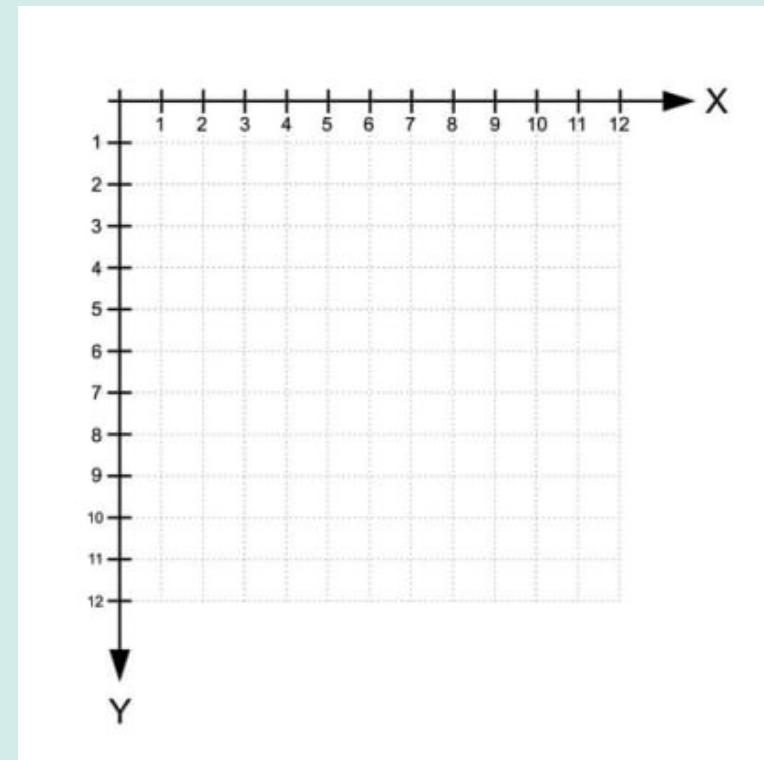
if keyboard_check(vk_left)
{
    x-=1
}                                1 píxel para a esquerda

if keyboard_check(vk_right)
{
    x+=1
}                                1 píxel para a direita
```

TUTORIAL GAME MAKER

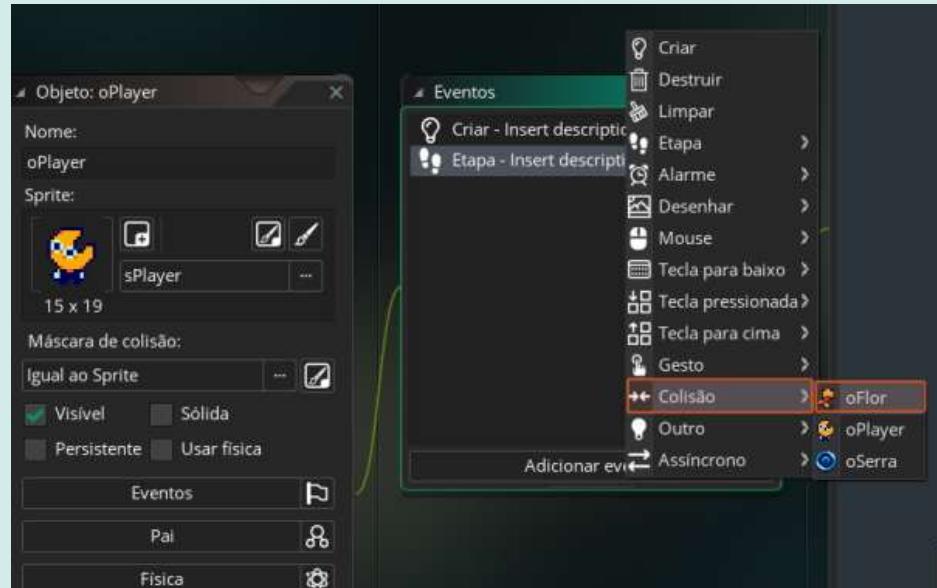
26 – Entenda a fase como um plano cartesiano, nosso personagem sempre se encontra em algum ponto desse plano e para fazer ele se movimentar, vamos alterar suas coordenadas durante o jogo.

Assim, ao apertar a tecla de cima, o jogador diminui 1 da variável Y. Ao apertar baixo, o jogador soma 1 a essa variável Y.



TUTORIAL GAME MAKER

27 – CRIAR COLISÕES: Adicione um novo evento ao “oPlayer” clique em “Colisão” e selecione o objeto “oFlor”. O código que estará aqui será reproduzido assim que o jogador encostar na flor.

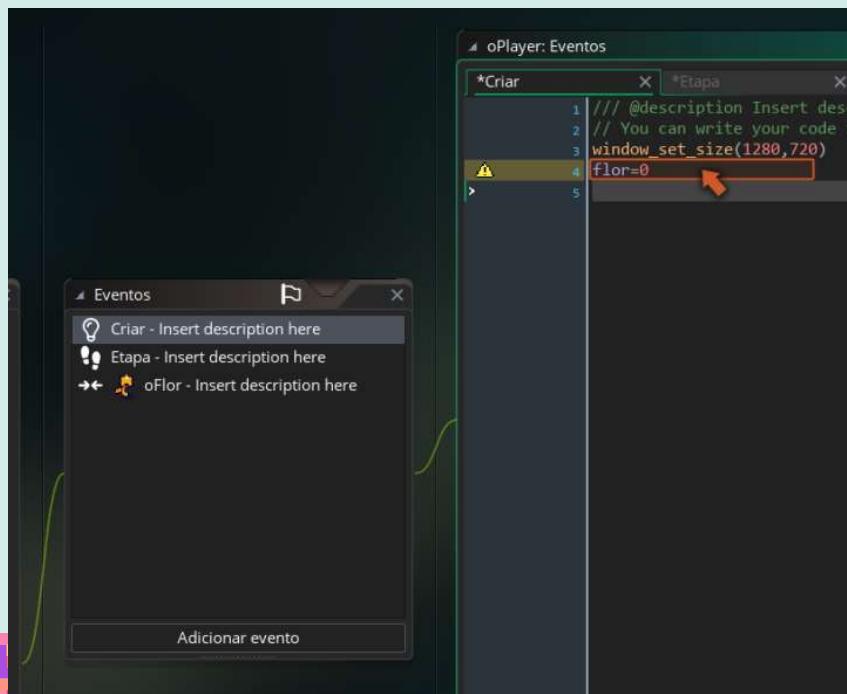


instance_destroy(other)

Essa função destrói algum objeto. Escrevendo dentro “other”, ela vai destruir outro objeto que, neste caso, vai ser “oFlor”, por ser quem está participando da colisão.

TUTORIAL GAME MAKER

28 – Queremos que ele consiga contar quantas flores ele tem e para isso criaremos uma variável. Vamos clicar no “Criar” do “oPlayer” e criar a variável flor.

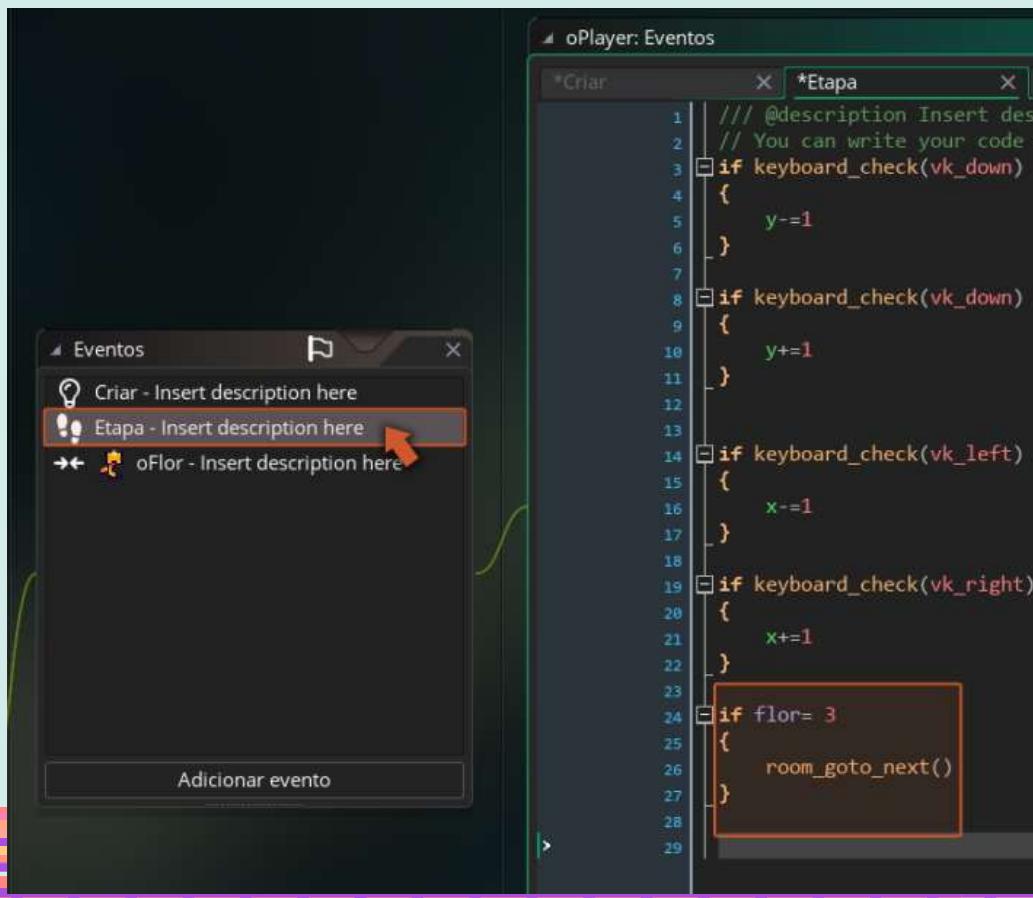


flor=0

Uma vez que foi criada, essa variável pode ser manipulada, podemos aumentar, diminuir e checar qual o estado dela.

TUTORIAL GAME MAKER

29 – Clique em “Etapa”, adicione o código



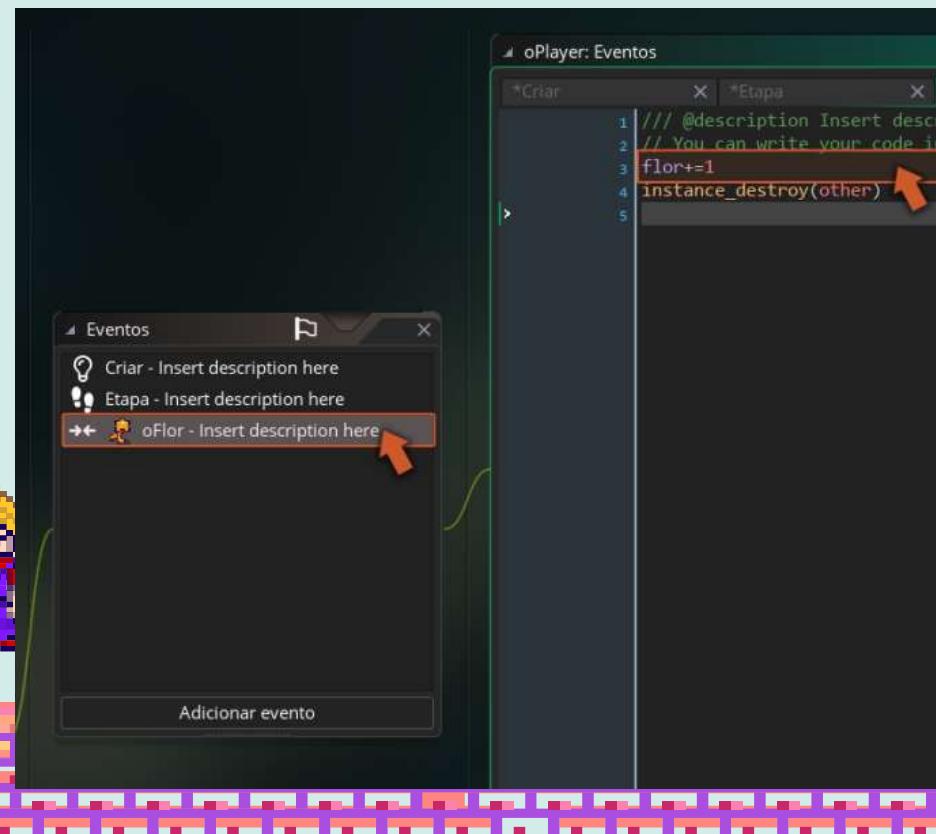
```
if flor= 3
{
    room_goto_next()
}
```

Esse é um código que permite que o jogador passe de fase, ele leva a gente do ambiente atual para o próximo ambiente da lista assim que o jogador coletar 3 flores.

Em inglês os ambientes são chamados Rooms e a função para passar para a próxima Room é “room_goto_next()”

TUTORIAL GAME MAKER

30 – E para fazer o jogador lembrar de cada flor que é colecionada, volte para a “Colisão” com “oFlor” adicione ao código:

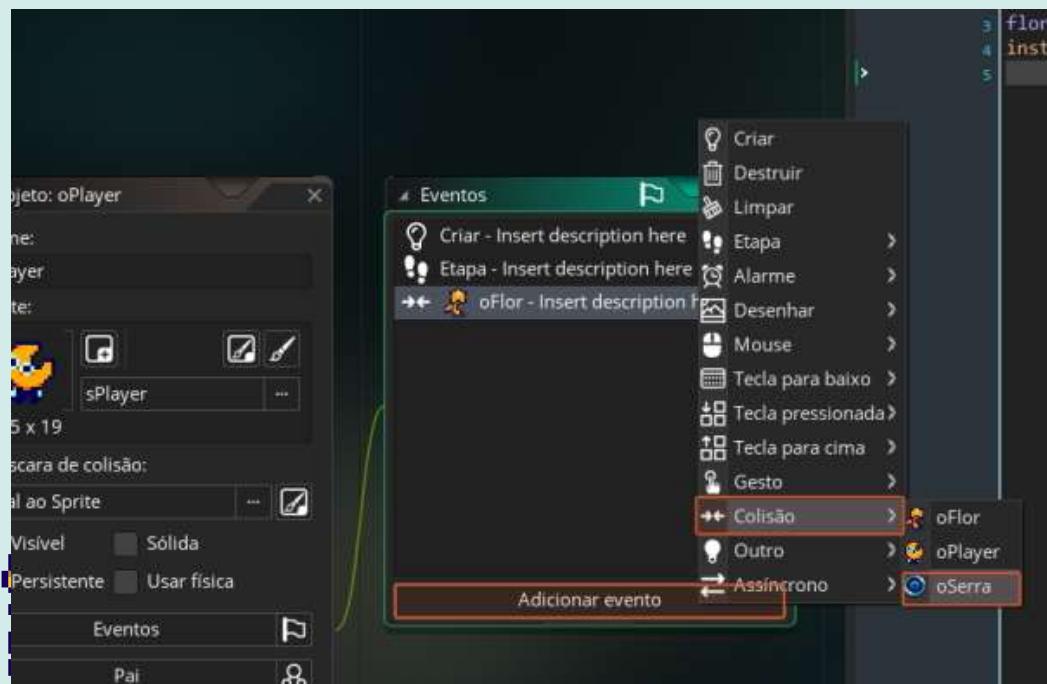


flor+=1

Com isto, toda vez que o personagem encostar numa flor, ele vai aumentar 1 na variável “oFlor” dele, e vai destruir a flor que encostou.

TUTORIAL GAME MAKER

31 – MORRER NA SERRA E REINICIAR O JOGO: No “oPlayer” adicione um evento de “Colisão” com a “oSerra”.

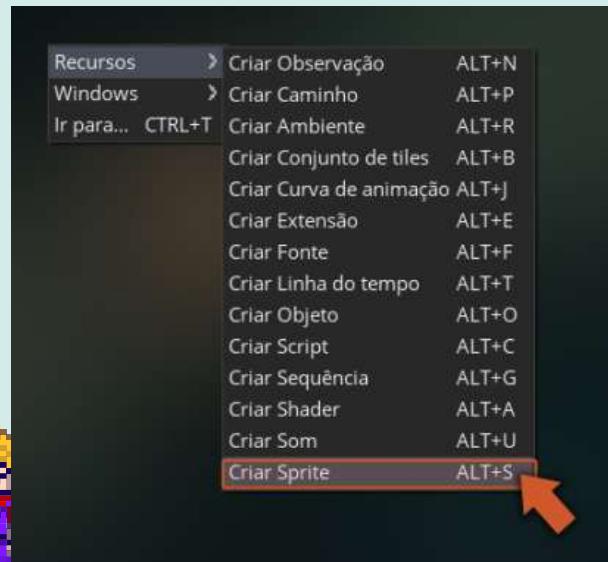


room_restart()

Aqui vamos fazer a cena reiniciar assim que o personagem toca na serra, dando a impressão de que ele morreu.

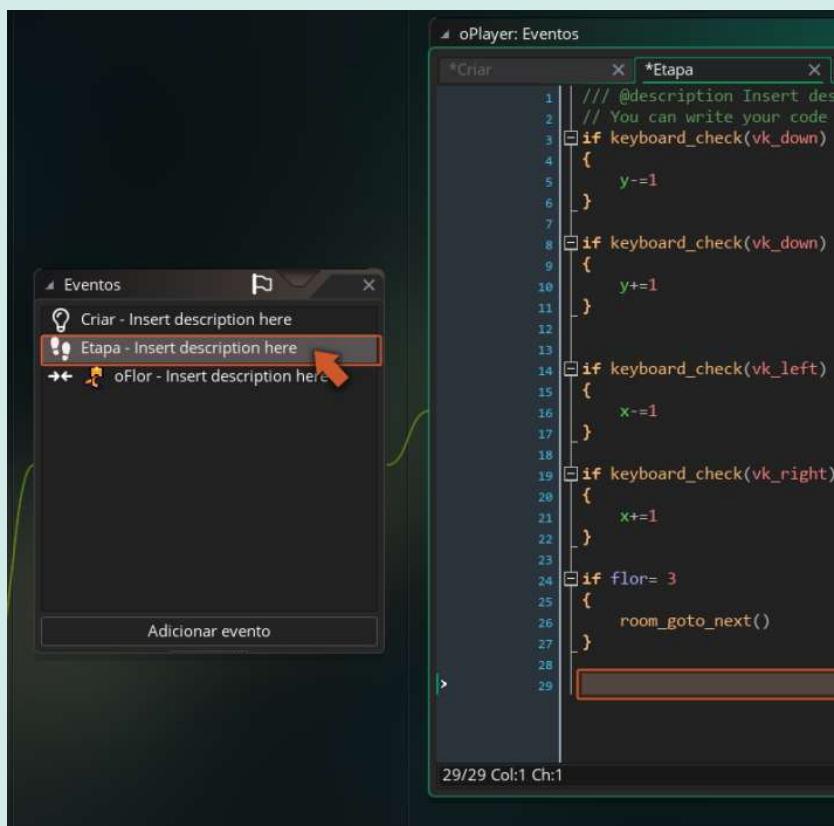
TUTORIAL GAME MAKER

31 – MOVIMENTO DO PLAYER - ANIMAÇÃO: Para movimentar nosso personagem corretamente temos que criar um Sprite, que vai ser o Sprite do nosso player correndo.



TUTORIAL GAME MAKER

32 – Para concluir a movimentação do nosso personagem, temos que voltar para a janela do “oPlayer”, e clicar em “Etapa”.



```
if keyboard_check(vk_anykey)
{
    sprite_index = sPlayerCorre
}
else
{
    sprite_index = sPlayer
}
```

Se apertarmos qualquer tecla, o Sprite do nosso objeto vai ser ele correndo. Se não (tradução de else), o Sprite dele vai ser ele parado.

TUTORIAL GAME MAKER

33 – Vamos adicionar “image_xscale=-1” e “image_xscale=1”, fazendo:

```
if keyboard_check(vk_left)
{
    x-=1 image_xscale=-1
}

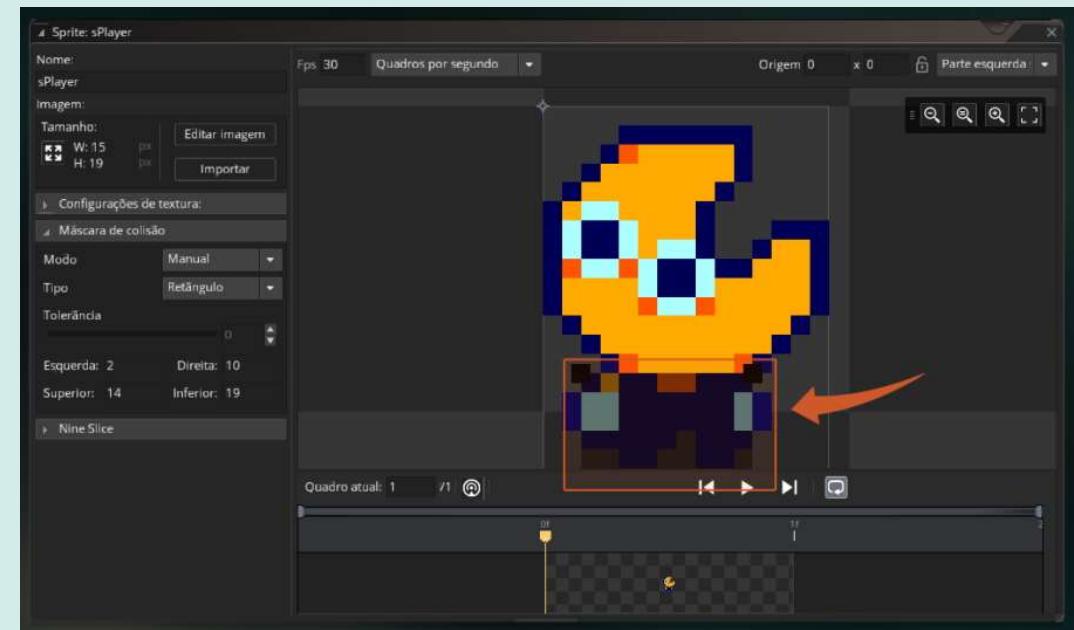
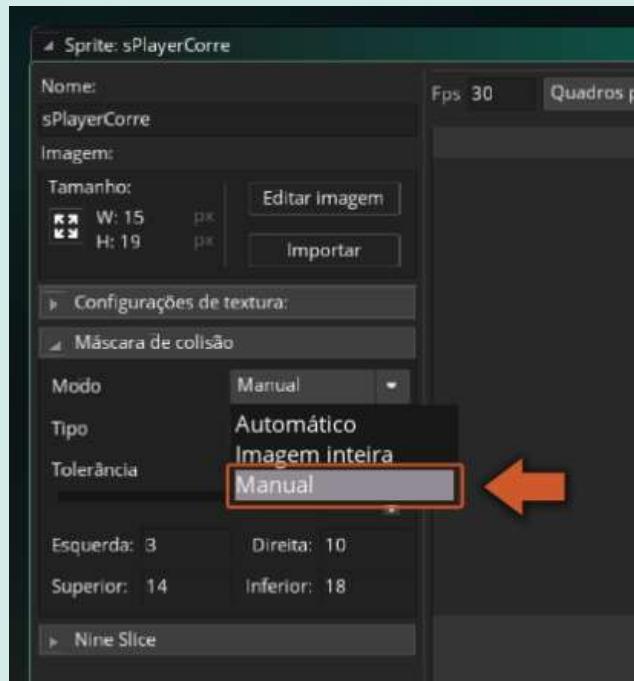
if keyboard_check(vk_right)
{
    x+=1 image_xscale=1
}
```

Esse “image_xscale” significa escala de imagem desse personagem. Se deixarmos 2, ele ficará duas vezes maior horizontalmente, 1, ele vai ficar normal. E se deixarmos -1, ele vai ficar invertido horizontalmente, sendo justamente o que nós queremos.



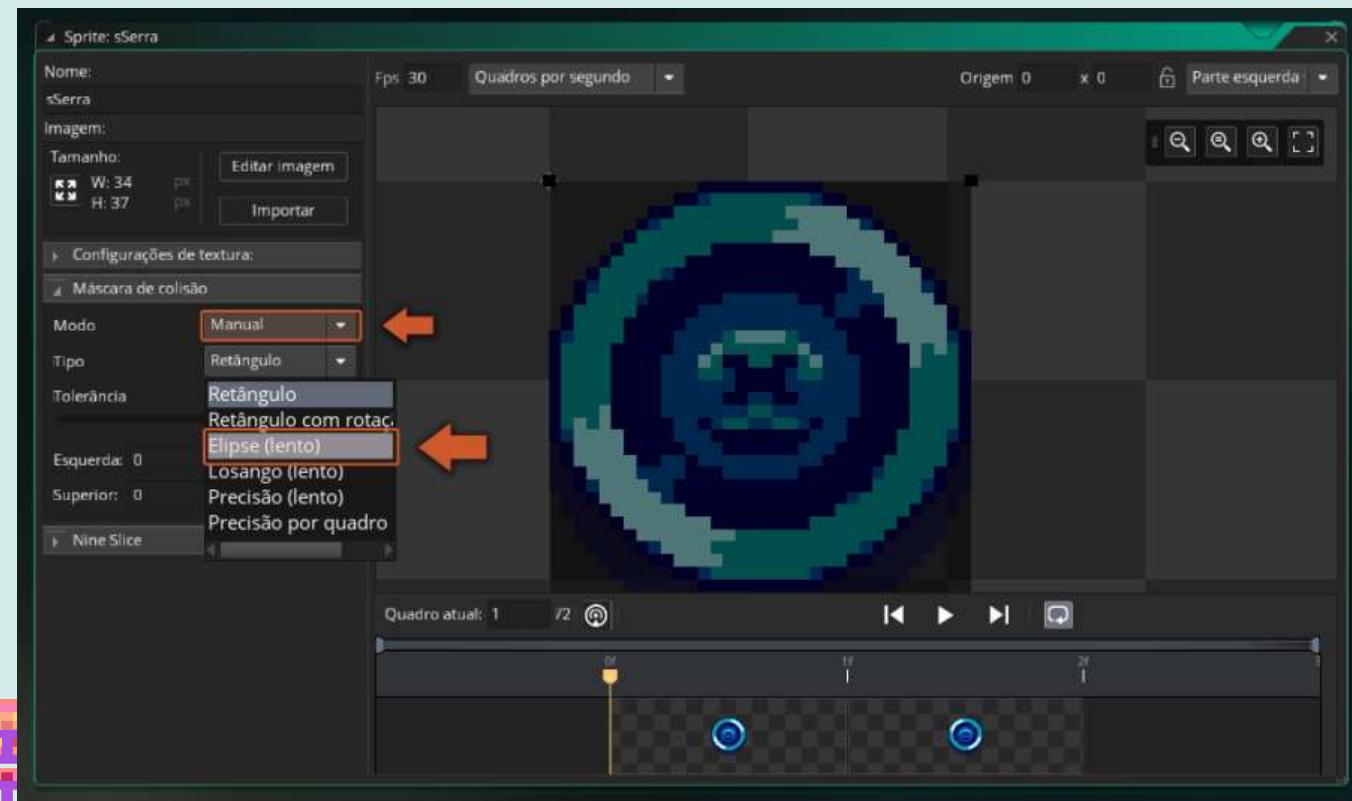
TUTORIAL GAME MAKER

34 – MÁSCARA DE COLISÃO: Considerando que o jogo é visto de cima, vamos mudar essa máscara de colisão, para a parte de baixo do nosso personagem.



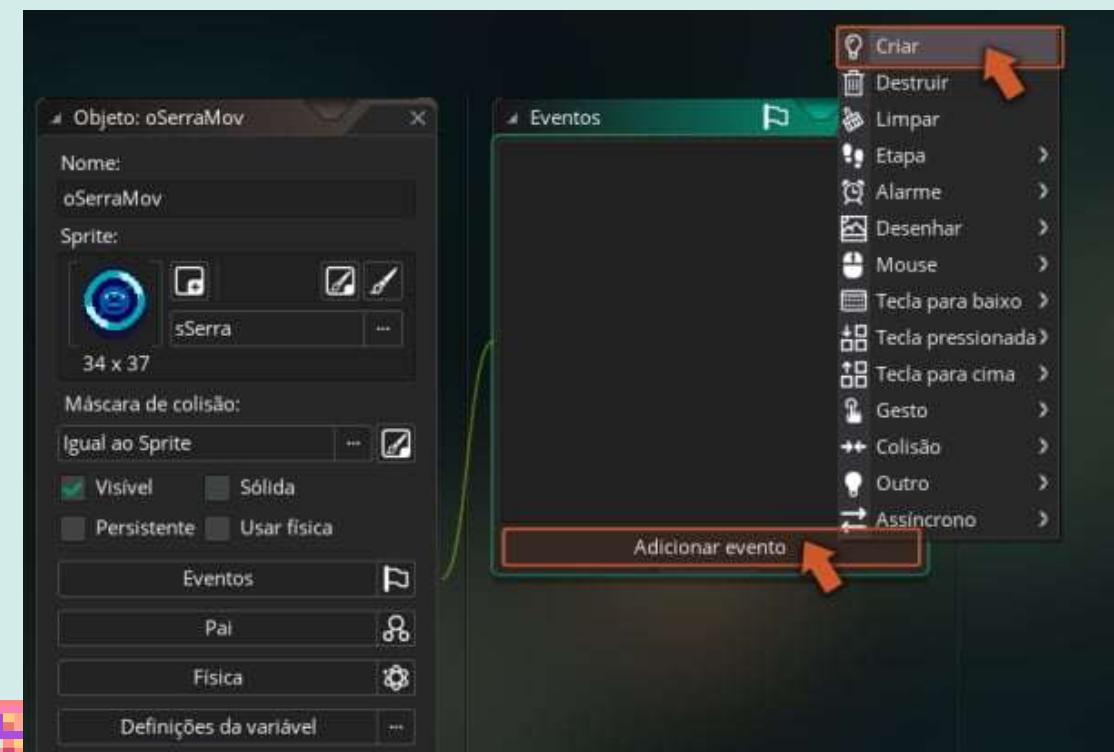
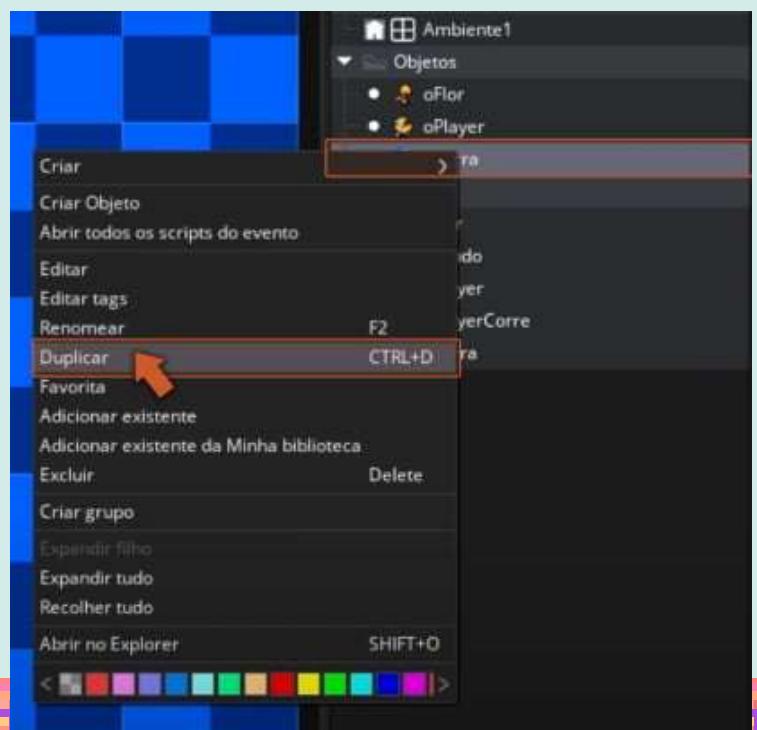
TUTORIAL GAME MAKER

35 – Agora vá para o Sprite da serra, clique em “máscara de colisão”, mude do modo “automático” para o “manual” e escolha “Elipse” (visto que estamos falando de um círculo).



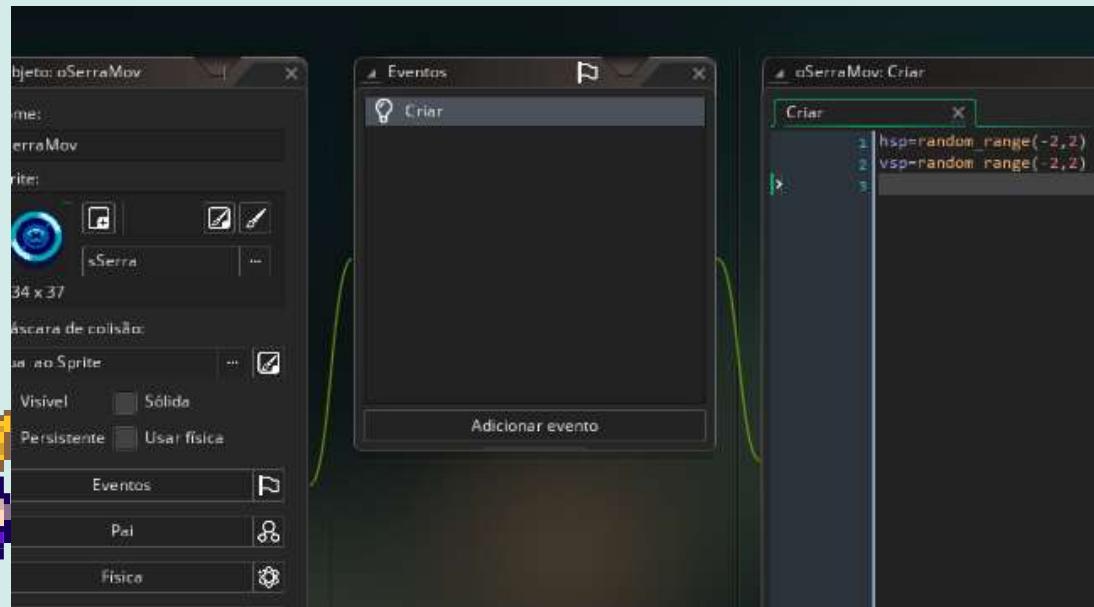
TUTORIAL GAME MAKER

36 – SERRAS QUE MOVEM: Clique no “oSerra” pelo navegador de recursos usando o botão direito, e clique em “duplicar”. Vamos chamar esta nova serra de “oSerraMov”



TUTORIAL GAME MAKER

37 – Já que queremos manipular a velocidade desse objeto, vamos criar uma variável de velocidade horizontal, que vamos chamar “hsp”, e uma variável de velocidade vertical, que vamos chamar “vsp”.

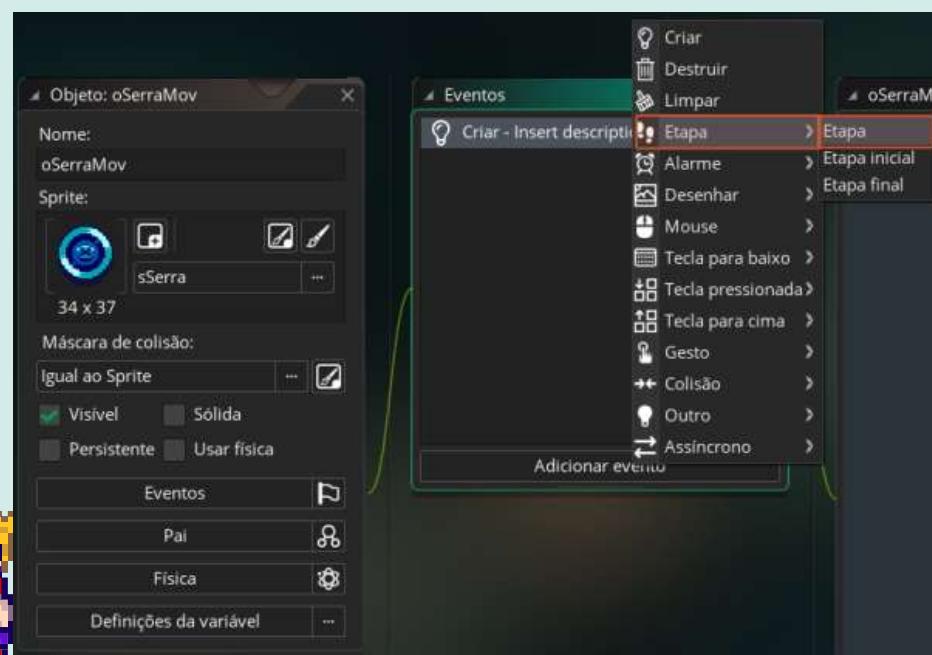


```
hsp=random_range(-2,2)
vsp=random_range(-2,2)
```

Essa função o que vai fazer é trazer um número aleatório entre -2 e 2, ou seja, 1.5, 0.75, -1, -2.

TUTORIAL GAME MAKER

38 – Voltando ao “adicionar um evento”, clique em “Etapa”, e vamos começar a escrever os códigos dessa serra.



$x+=hsp$

$y+=vsp$

Assim podemos modificar os valores do X e do Y usando as variáveis de velocidade. Por exemplo, usando o número 2 como “hsp”, o X da serra sempre vai somar com 2, com um “vsp” de -2 por exemplo, o Y da serra sempre vai ser subtraído por 2.

TUTORIAL GAME MAKER

39 – Clique em “Etapa” e vamos fazer um código para ela quicar pela tela. Escreveremos um código que faça o “hsp” da serra se inverter quando seu X for abaixo de 0 e quando seu X for acima do tamanho da fase.



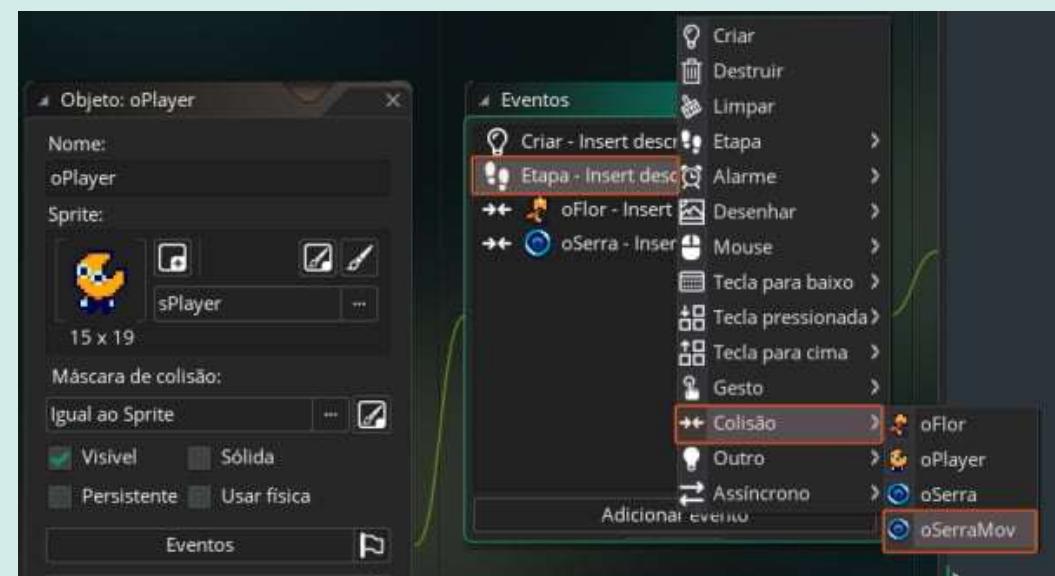
```
oSerraMov: Eventos
  Criar
  Etapa
  1 | x+=hsp
  2 | y+=vsp
  3 |
  4 | if (x<0) hsp=-hsp
  5 | if (x>room_width) hsp=-hsp
  6 |
  7 | if (y<0) vsp=-vsp
  8 | if (y>room_height) vsp=-vsp
  9 |
```

if ($x < 0$) $hsp = -hsp$
if ($x > room_width$) $hsp = -hsp$

if ($y < 0$) $vsp = -vsp$
if ($y > room_height$) $vsp = -vsp$

TUTORIAL GAME MAKER

40 – Feito isso, vamos fazer o player morrer para essa serra também. Para isso vamos outra vez para “oPlayer”, “Adicionar Evento”, “Colisão” e escolher o objeto “oSerraMov”.



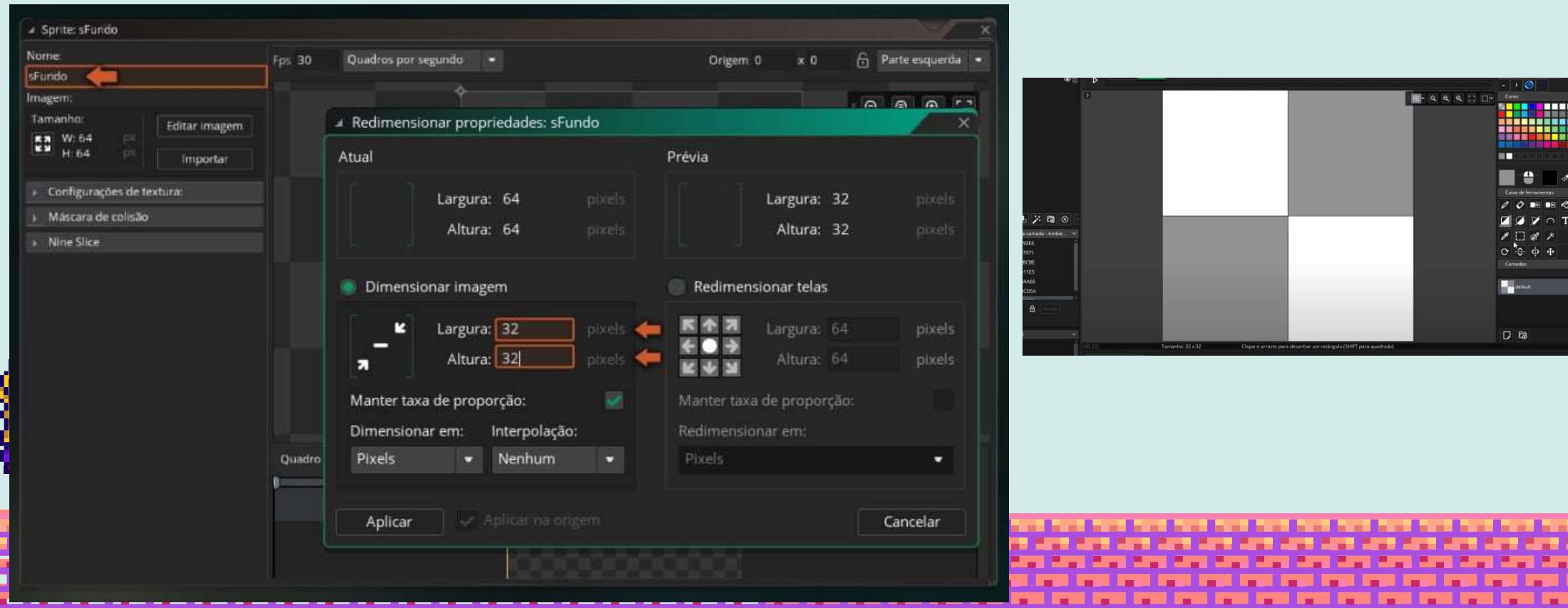
E mais uma vez vamos escrever:

`room_restart()`

Agora é só ir para o “Ambiente”, clicar na camada “Instances” e colocar nossa nova serra pela fase.

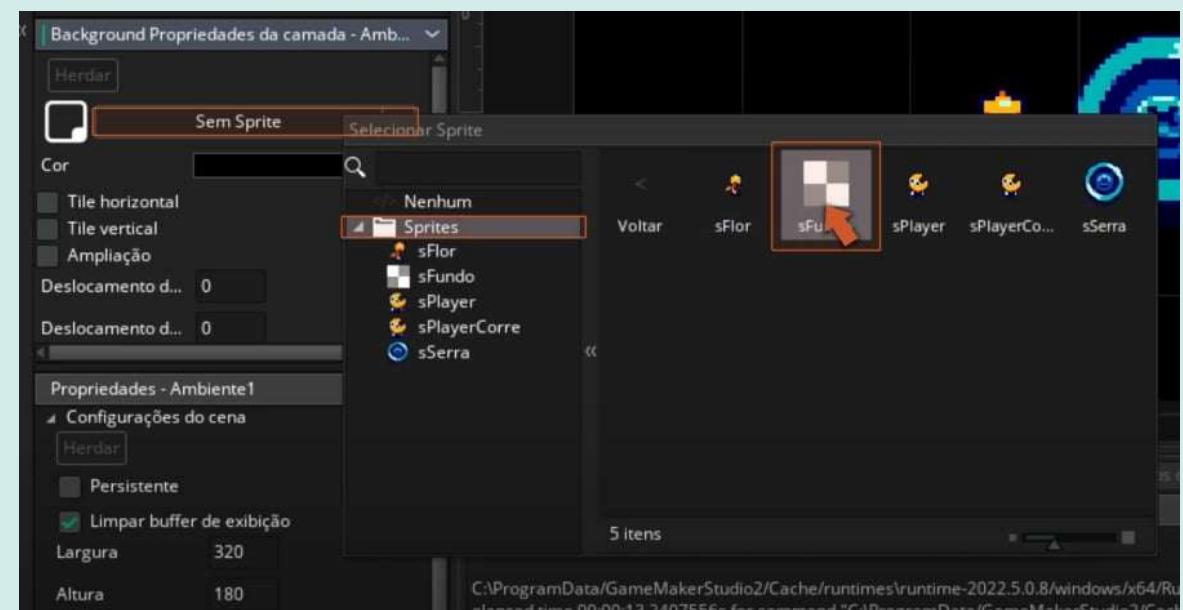
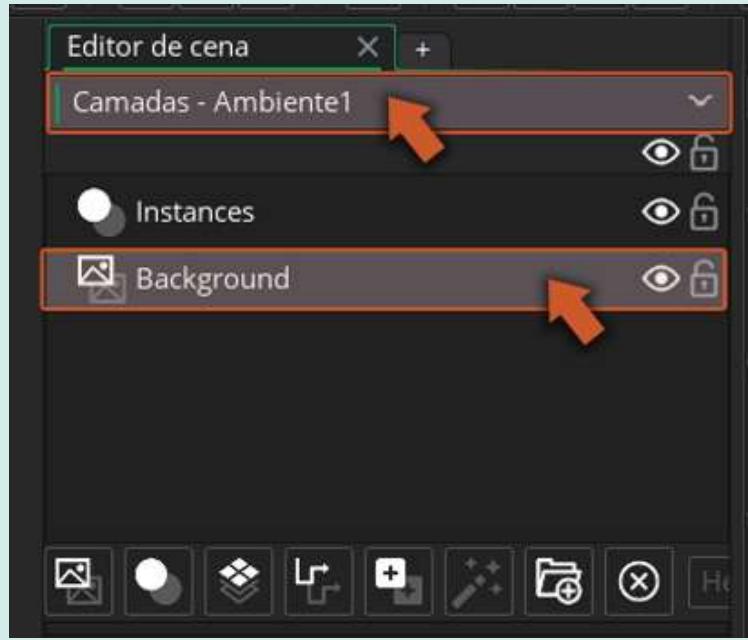
TUTORIAL GAME MAKER

41 – FUNDO: Vamos criar um Sprite para o fundo, que vai ser Sprite bem simples, do tamanho 32 por 32. Eu fiz uma grande branca e cinza que vamos chamar: “sFundo”.



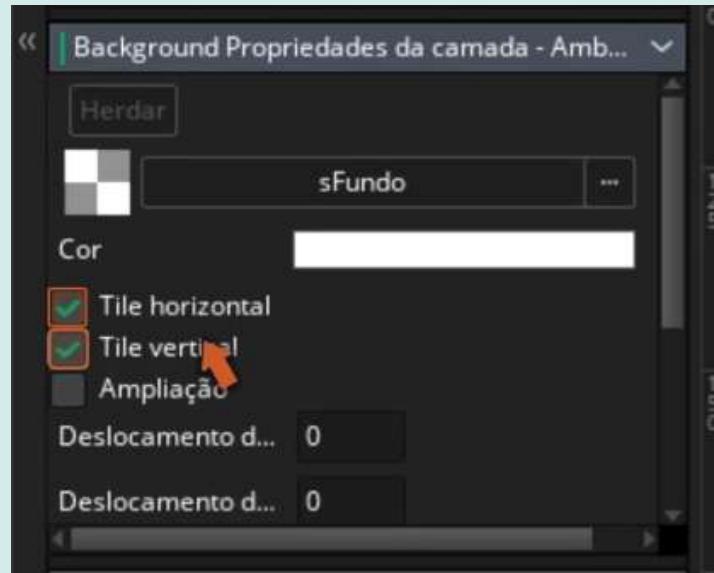
TUTORIAL GAME MAKER

42 – Clique no “Ambiente”, vá para as “Camadas” e “Background”, sendo a camada da nossa cena que representa o fundo da fase.



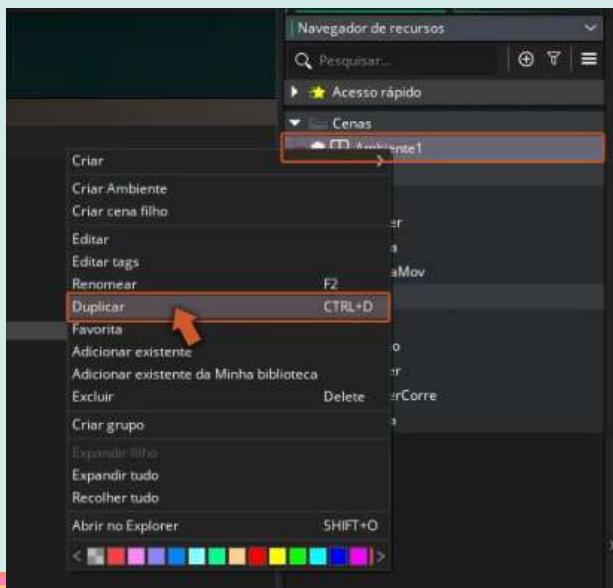
TUTORIAL GAME MAKER

43 – Clicando aqui nas propriedades dessa camada, vamos selecionar o “sFundo” e fazer ele se repetir horizontal e verticalmente. Para mudar a cor dele, clique em “Cor” e escolha uma cor de sua preferência.



TUTORIAL GAME MAKER

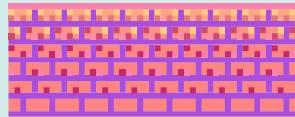
44 – PRÓXIMO NÍVEL: Vamos clicar com o botão direito em cima do ambiente que já temos, duplicar e renomear “Ambiente 2”. Aqui vamos botar serras paradas, serras que se movimentam e flores em locais diferentes da primeira fase. Podemos até clicar aqui em “background” e mudar a cor desse fundo.



TUTORIAL GAME MAKER

CONCLUSÃO

O resultado é um jogo bem simples, que servirá de introdução ao GameMaker. Muitos códigos escritos nesse tutorial não são extremamente otimizados, porém, são simples e relativamente fáceis de serem compreendidos.



ATÉ A PÓXIMA AULA!

