Aprendendo a Desenvolver

Jogos Usando a engine GDevelop 5

Parallax Scrolling

O que é Parallax Scrolling

- É uma técnica utilizada na computação onde as imagens de plano de fundo se movem em velocidades diferentes, criando uma ilusão de profundidade em cenários 2D. Quanto mais perto, mas rápido é a velocidade de movimento do plano de fundo.

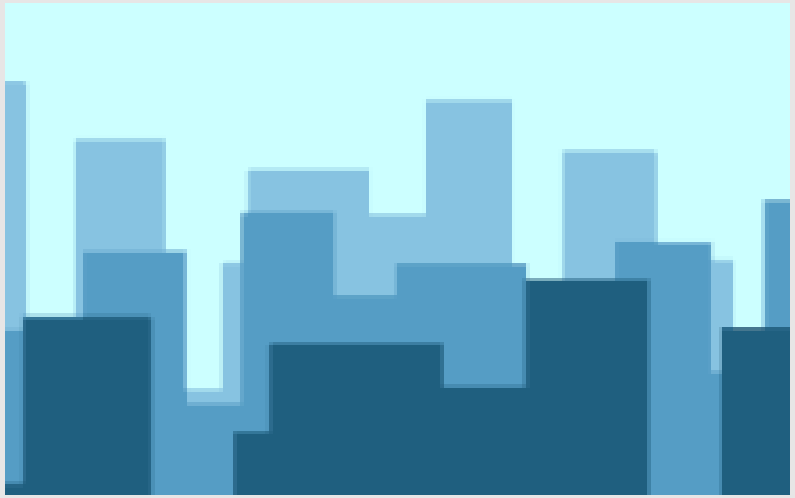


Figura: Parallax Scrolling de uma cidade.[1]

[1]https://en.wikipedia.org/wiki/Parallax_scrolling

O que é Parallax Scrolling

- O efeito Parallax Scrolling é utilizado de forma extensiva em jogos do gênero shoot-em-ups e plataforma.

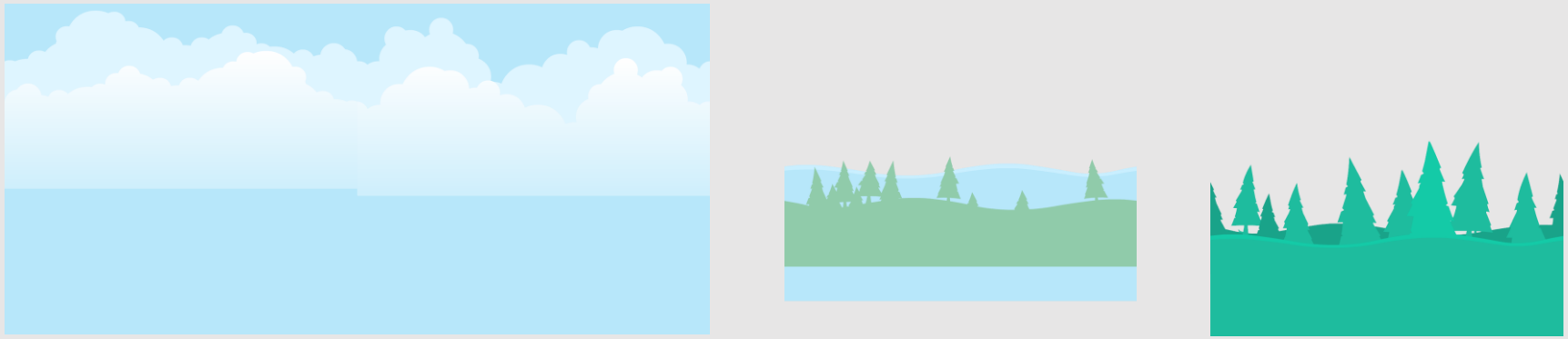
Figura: Parallax Scrolling em jogos de Plataforma.[2]

[2]https://gamemaker.io/it/blog/coffee-break-tutorials-parallax-scrolling-gml

1. Definindo e organizando os backgrounds para criar o efeito Parallax

Definindo os backgrounds

- Para criar um efeito Parallax Scrolling utilizando a engine GDevelop 5 iremos utilizar os seguintes planos de fundo.



a) Nuvens b) Montanhas c) Árvores

Figura: Planos de fundo utilizados para simular a profundidade do cenário

- Cada plano de fundo representará uma distância diferente, respectivamente, Nuvens (Longe), Montanhas (distância Média) e Árvores(Perto).

Definindo os backgrounds

- Os planos de fundo utilizados no exemplo possuem uma altura de 960px e larguras com dimensões diferentes. Entretanto, importante citar que, dependendo dos planos de fundo utilizados, esses **DEVEM** respeitar a resolução mínima do jogo para que não haja distorções.

- Para esse exemplo, o jogo possuirá uma resolução de 1440x900px.

- Nos precisamos também criar varias camadas em nossa cena para criar o efeito parallax.

Definindo os backgrounds

- Para uma melhor organização, os planos de fundo serão separados em Camadas(Layers).

- Abra o editor de camadas clicando na indicação da figura:

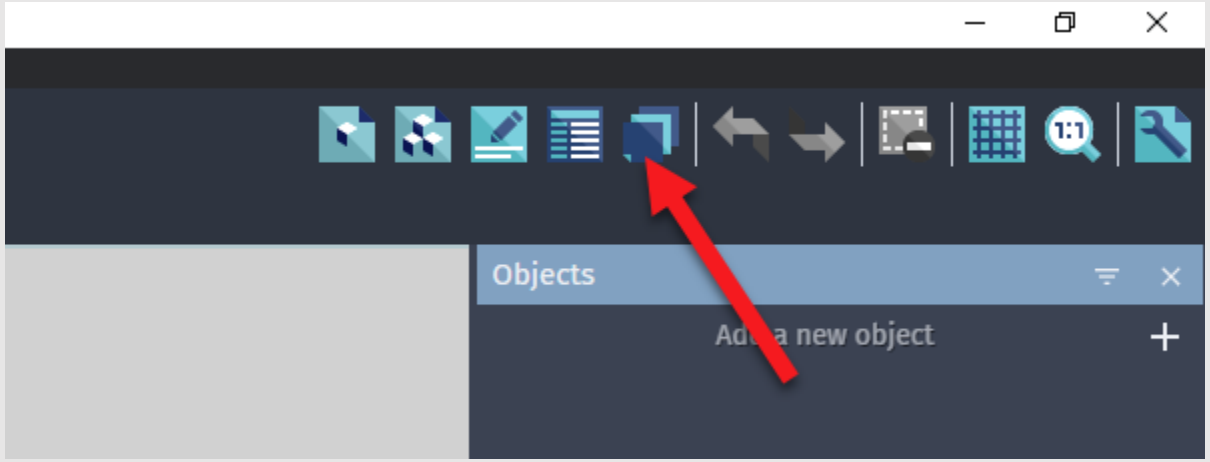


Figura: Abrindo o editor de camadas

Definindo os backgrounds

- Crie três camadas e as organize conforme a indicação da figura:

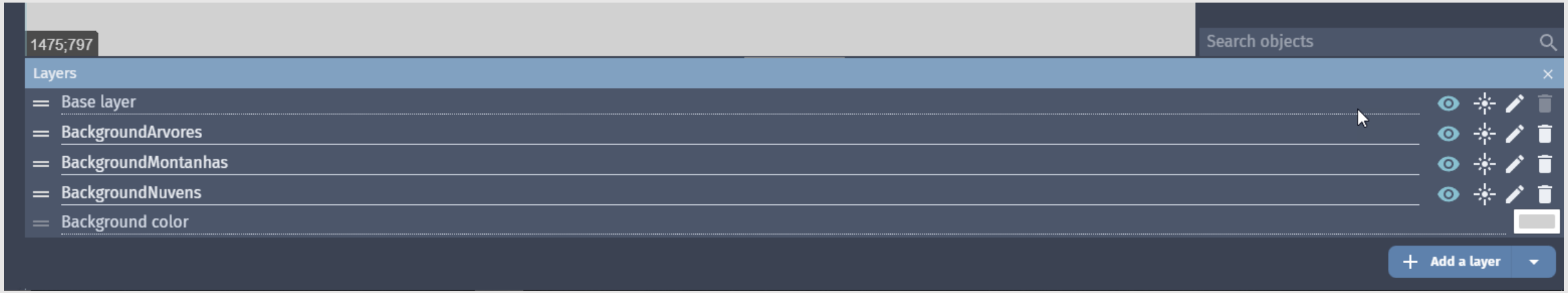


Figura: Criando e organizando as camadas

Definindo os backgrounds

- Quando queremos criar os backgrounds devemos utilizar o componente **Tiled Sprite**, pois ele nos permite duplicar a imagem na largura ou altura, dessa forma, evitando distorções na imagem. Devemos criar aqui três objetos do tipo **Tiled Sprite**.

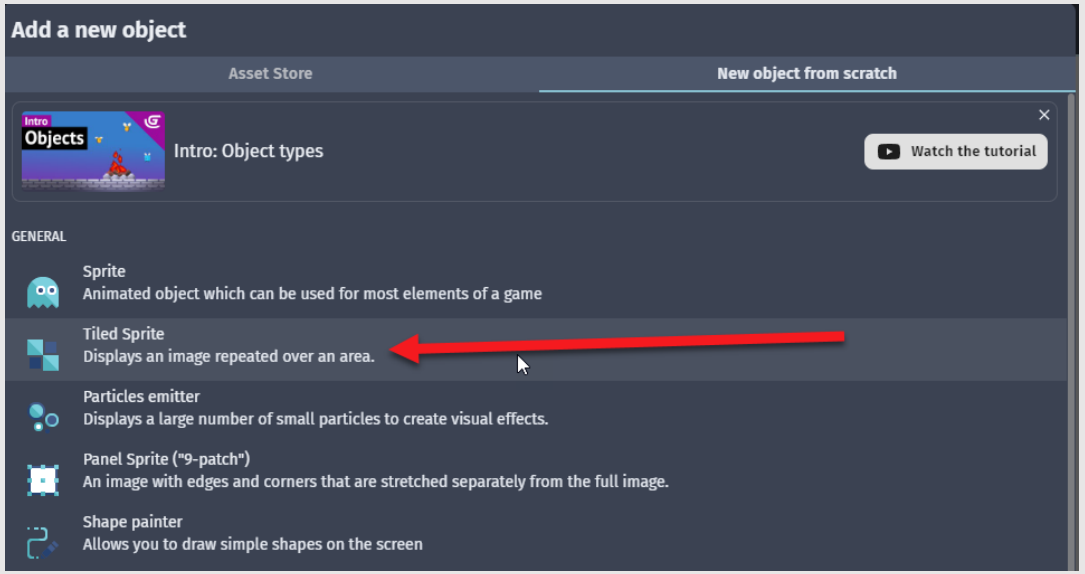


Figura: Definindo objetos do tipo Tiled Sprite

Definindo os backgrounds

- Para cada objeto **Tiled Sprite** devemos associar com o background correspondente.
- Um segundo ponto, é copiar as imagens dos backgrounds para dentro da pasta do projeto para evitar erros de referências de caminho nos diretórios.

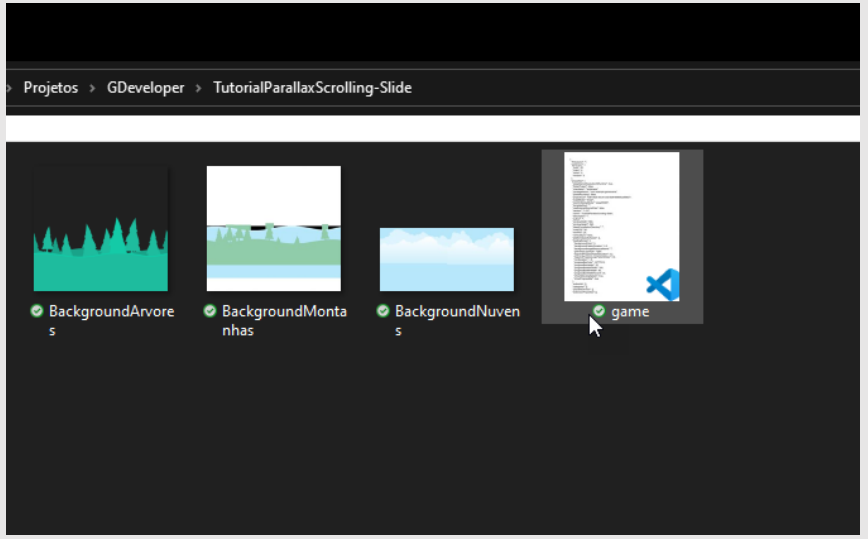


Figura: Copiando os backgrounds para a pasta do projeto

Definindo os backgrounds

- Caso não copie os arquivos para dentro da pasta do projeto, o GDevelop exibirá uma mensagem perguntando se deseja copiar a imagem para o diretório, nesse caso, devemos confirmar que desejamos realizar a cópia.
- Configure cada background conforme as imagens abaixo:

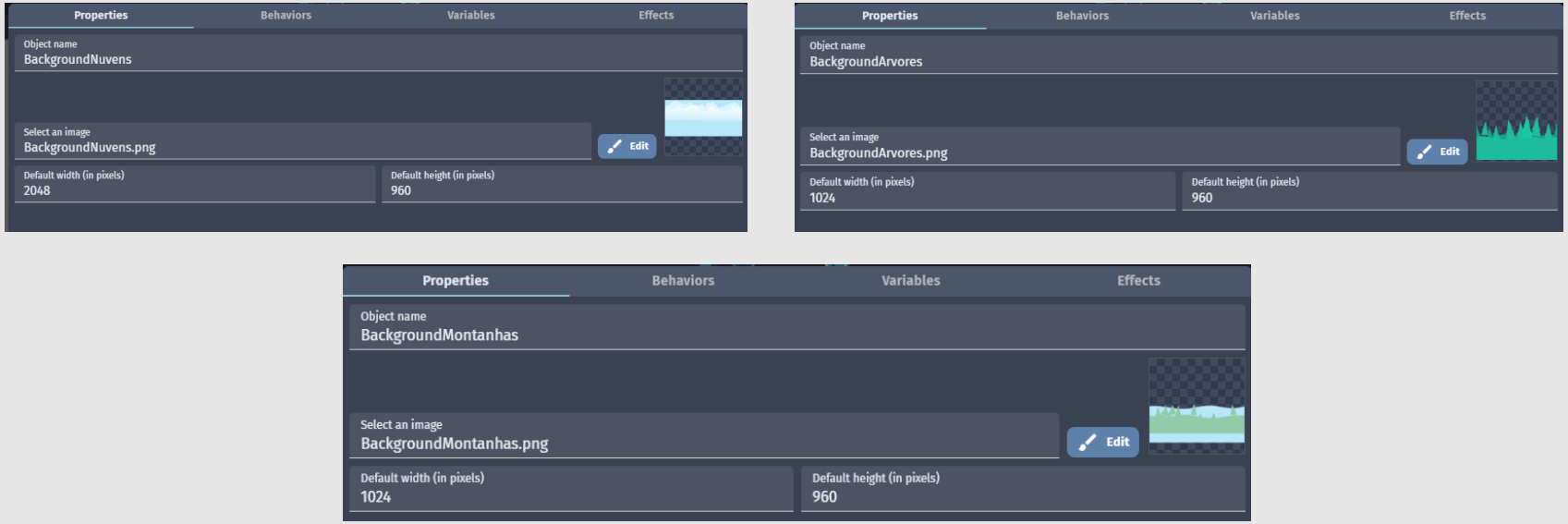


Figura: Criando os objetos Tiled Sprite para cada Background

Definindo os backgrounds

- Ao adicionar os backgrounds na cena, você observará que os backgrounds não possuem a mesma dimensão na largura.

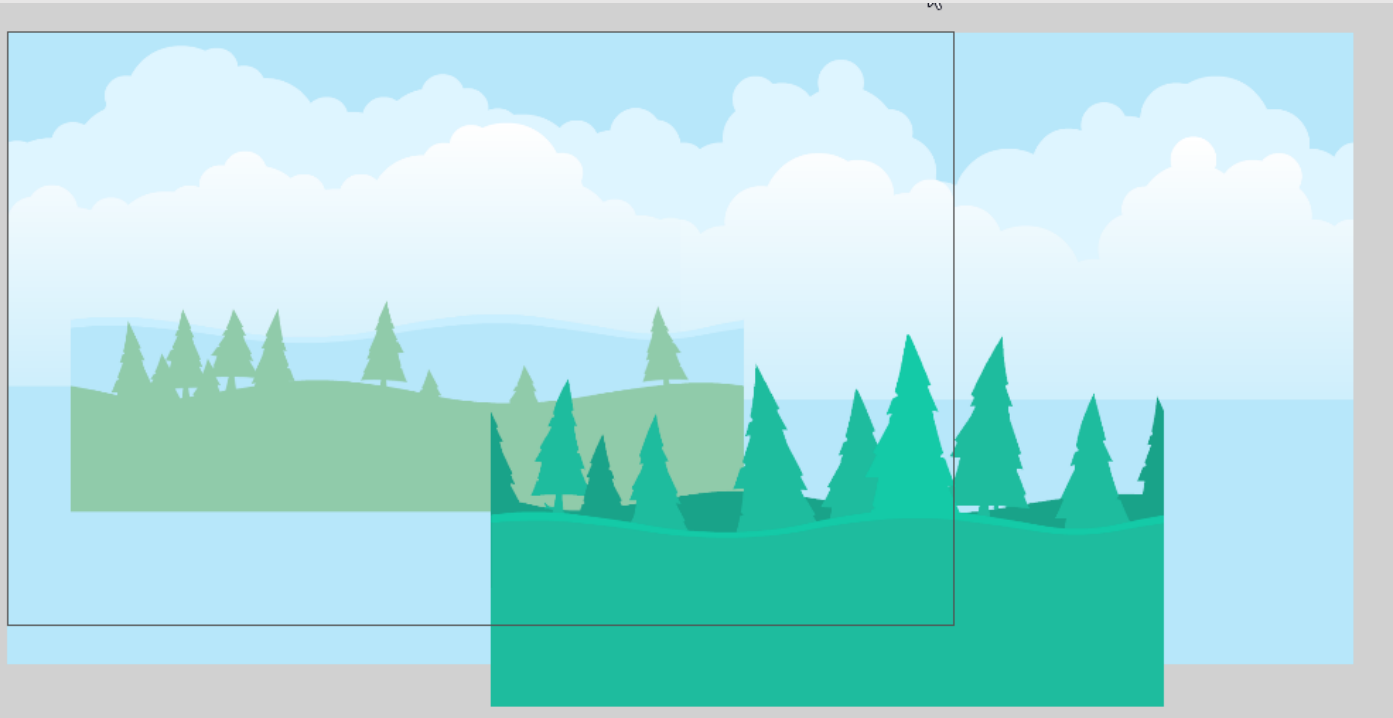


Figura: Backgrounds adicionados na cena

Definindo os backgrounds

- Antes de ajustar as dimensões dos backgrounds, devemos associar cada um deles nas suas respectivas camadas.

- Para cada background, associe com suas respectivas camadas conforme a figura abaixo:

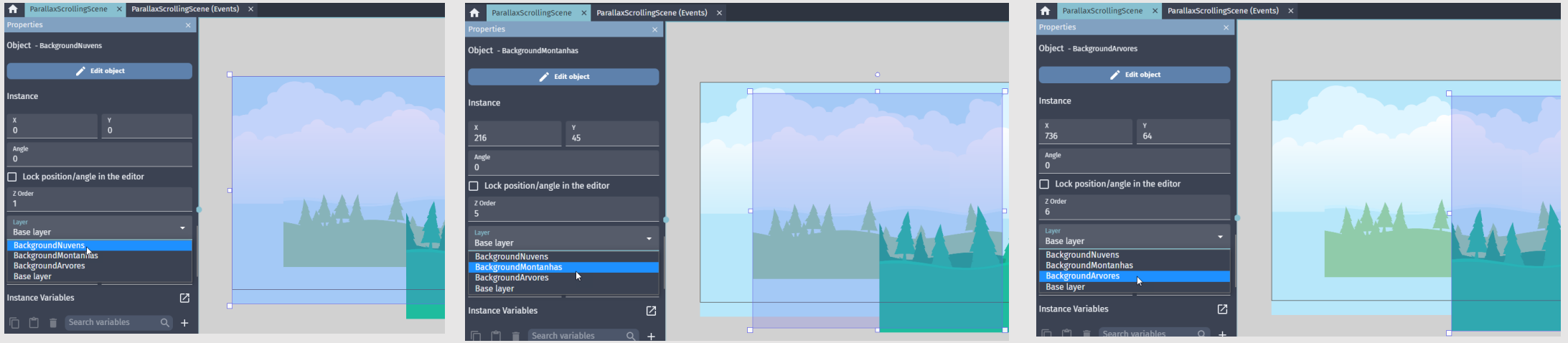


Figura: Associando cada background a sua respectiva camada

Definindo os backgrounds

- Ative a grade e ajuste os backgrounds na cena conforme a figura abaixo. Aqui definimos uma largura de 2048px para todos os backgrounds:

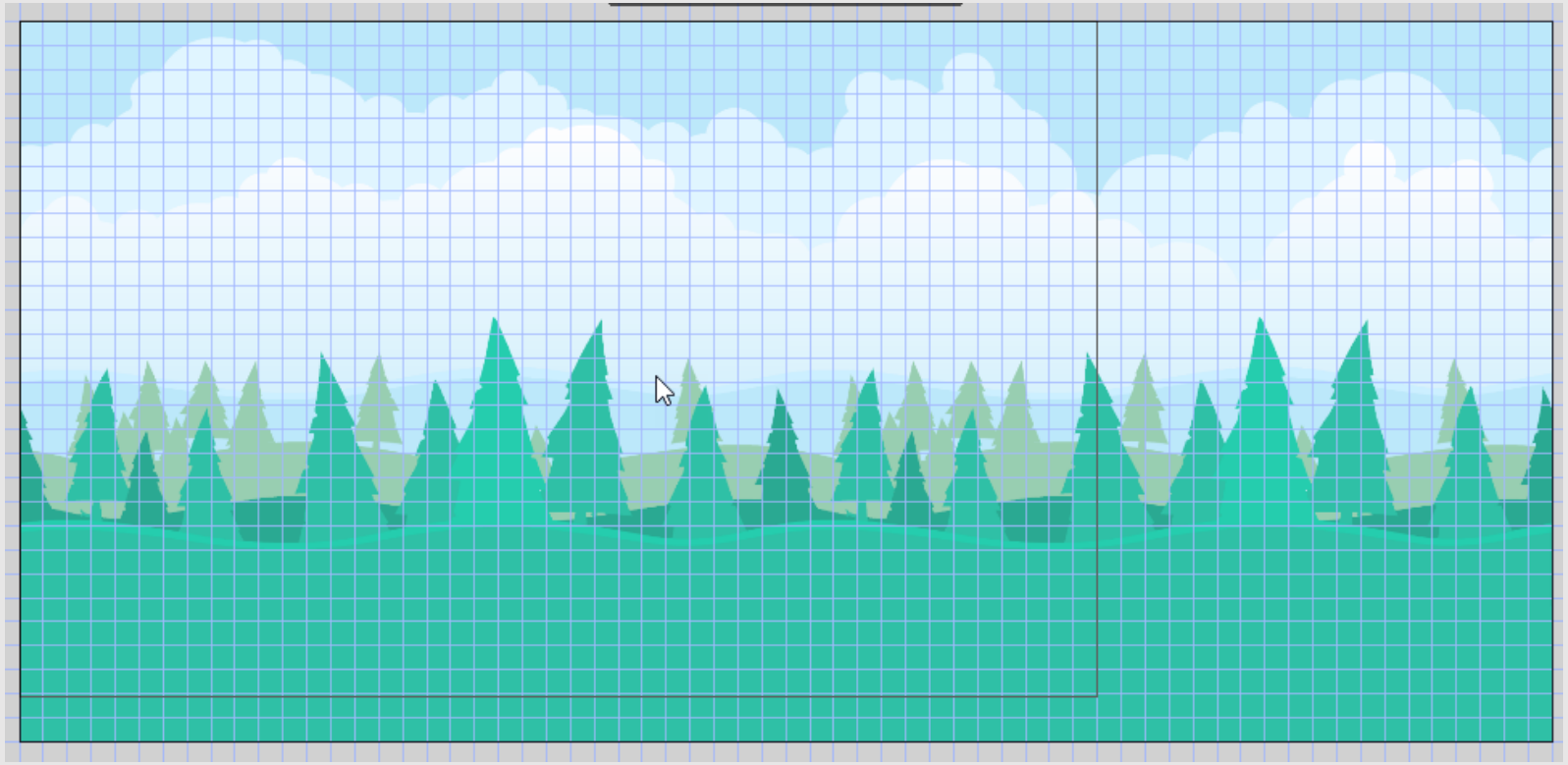


Figura: Backgrounds ajustados com a mesma largura

Definindo os backgrounds

- Por fim, adicione ao projeto objetos para representar o chão e o personagem controlado pelo jogador.

- Para esse exemplo, importei o objeto para representar o chão e o player da própria loja de assets store do GDevelop 5.

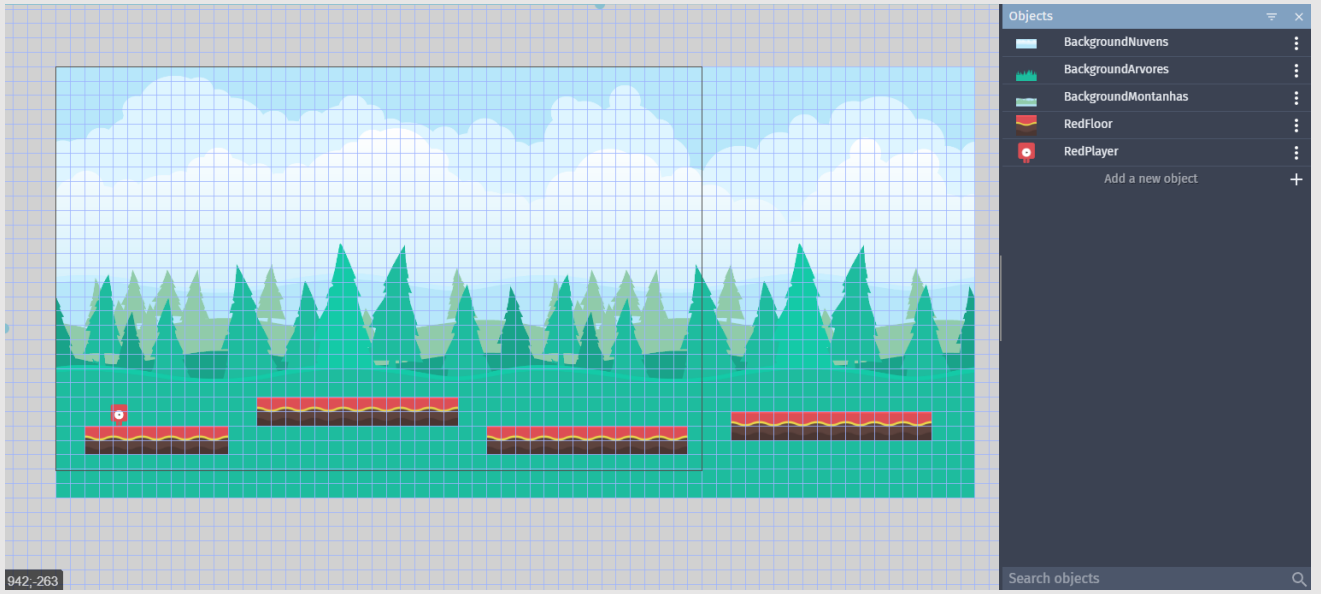


Figura: Cenário com a plataforma e o personagem controlado pelo jogador.

2. Criando um efeito de movimento Smoth Camera.

Movimento Smoth Camera

- A configuração básica do projeto até esse ponto está pronta. Agora com poucas linhas de código iremos configurar o efeito **Smoth Camera**.
- Adicione a primeira ação chamada **Camera center X position.**

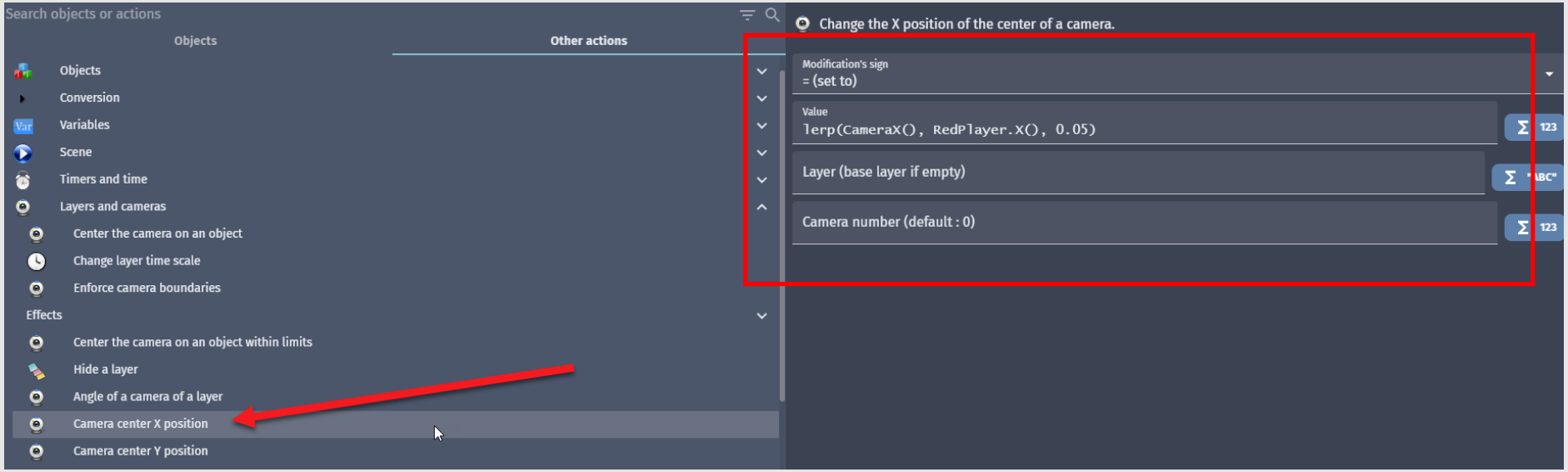


Figura: Definindo a ação para o movimento da câmera no eixo X

Movimento Smoth Camera

- Adicione uma nova ação, mas dessa vez, repetimos o mesmo procedimento para o eixo Y, utilizando a ação **Camera center Y position**.

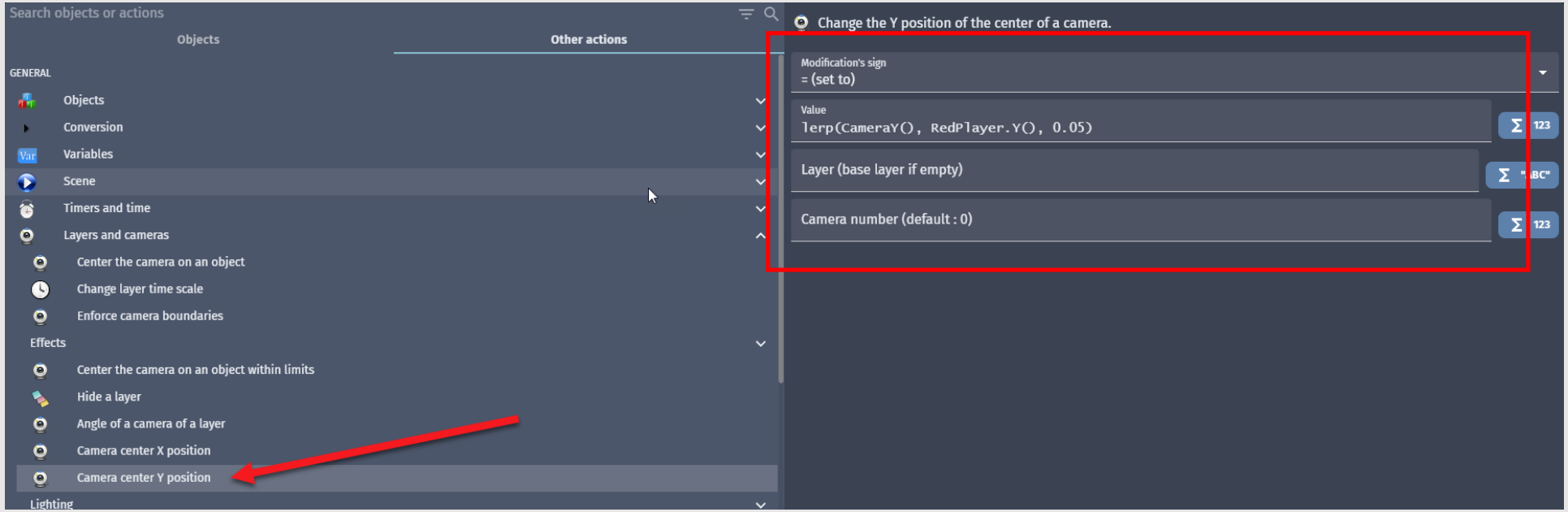


Figura: Definindo a ação para o movimento da câmera no eixo Y

Movimento Smoth Camera

- O código definido para o movimento da câmera utiliza uma função chamada **lerp** (Interpolação Linear). Não irei entrar em detalhes matemáticos sobre essa função, mas de forma simplificada, ela permite que dado dois pontos, um inicial e outro final, faz com que o ponto inicial se movimente até o ponto final de forma mais suave de acordo com o valor que deseja interpolar entre o ponto inicial e final.
- Para entender melhor essa função acesse os seguintes links:
  - Linear interpolation - Wikipedia
  - Interpolação linear – Wikipédia
  - (36) Cálculo Numérico - Aula 17 - Interpolação linear - YouTube

Movimento Smoth Camera

- Resultados da implementação do movimento **Smoth Camera**.

Movimento Smoth Camera

- Como próximo passo, vamos definir uma borda de limite de visualização da nossa câmera. Para o nosso exemplo, iremos limitar na largura 2048px e na altura 900px.
- Importante: Essa limitação irá depender do tipo de jogo que você irá desenvolver. Entretanto, esse limite a ser definido deve ser maior ou igual a resolução do seu jogo.

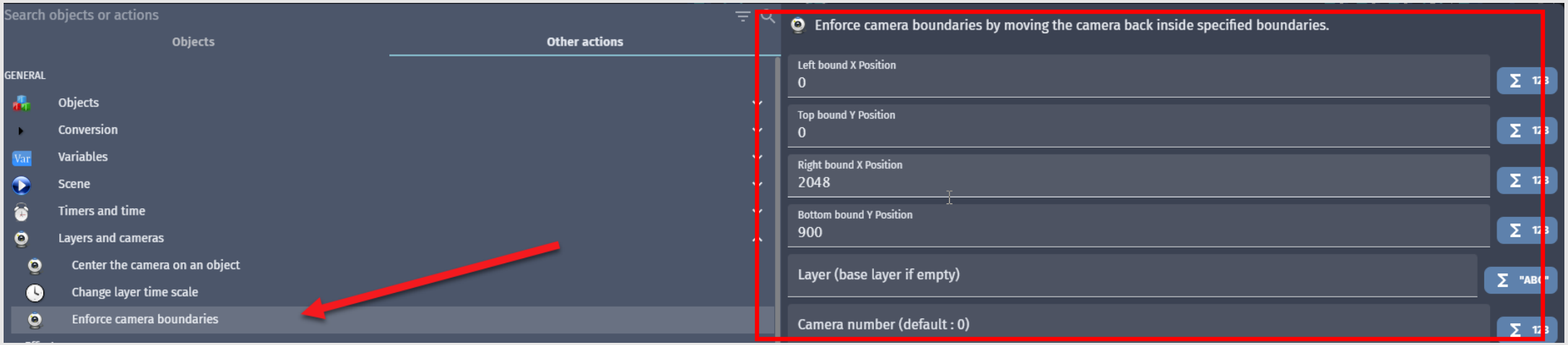


Figura: Limitando o movimento da câmera.

Movimento Smoth Camera

- Observe a animação abaixo demonstrando o limite da borda de câmera.

Figura: Sem limite da borda de câmera

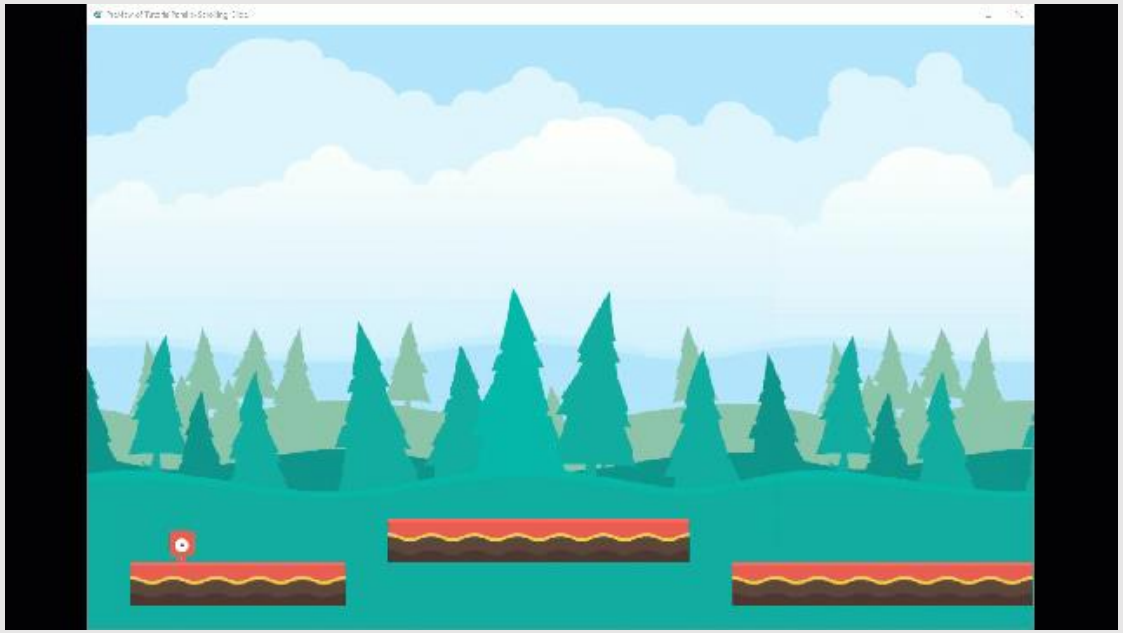


Figura: Câmera com limite de borda

3. Criando o efeito Parallax Scrolling

Criando o código para o efeito Parallax

- Agora, de fato, vamos definir o código que aplicará o efeito parallax nos planos de fundo. Nos devemos alterar a velocidade que o plano de fundo se move relativo a posição atual da câmera, neste caso, relativo ao eixo X.

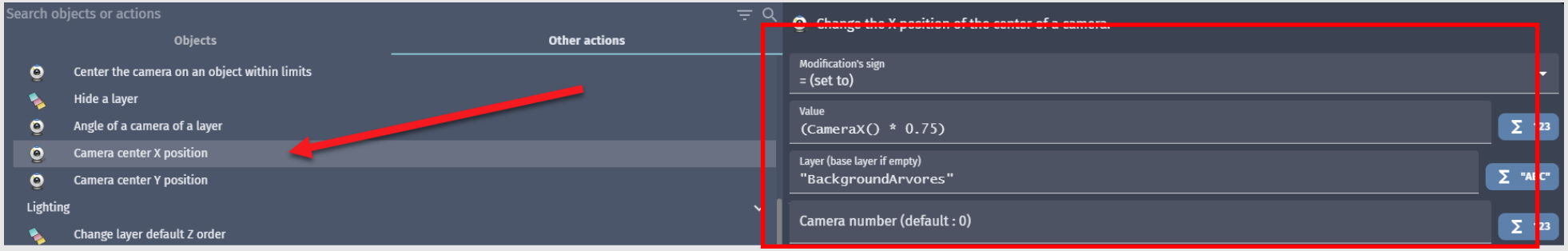


Figura: Movimento câmera para o plano de fundo das Árvores.

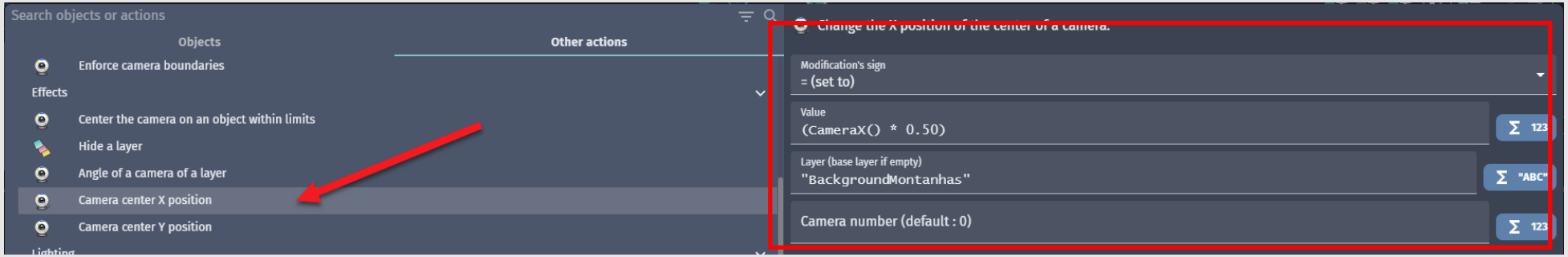


Figura: Movimento de câmera para o plano de fundo das montanhas

Criando o código para o efeito Parallax

- Observe que, para cada plano de fundo multiplicamos a posição do eixo X da câmera por um valor que representa a porcentagem que queremos mover o plano de fundo. Quanto menor a porcentagem, mais devagar o plano de fundo moverá, quanto maior a porcentagem, mais rápido é o movimento.

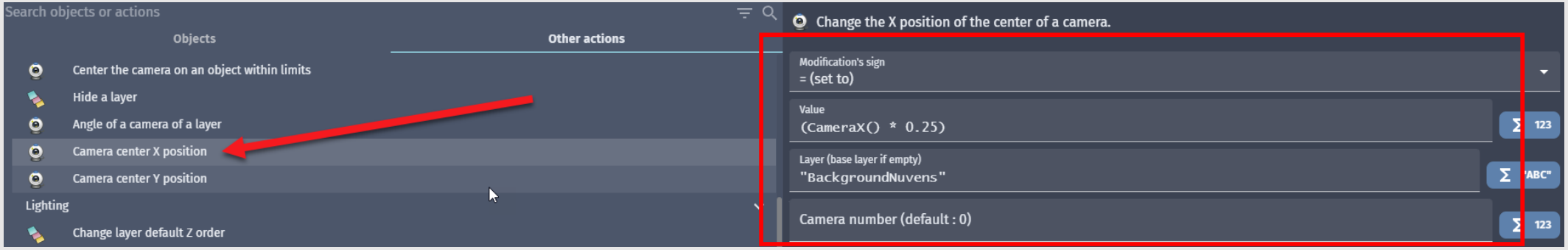


Figura: Movimento de câmera para o plano de fundo das nuvens.

Criando o código para o efeito Parallax

- Abaixo temos o código final implementado para o efeito Parallax.

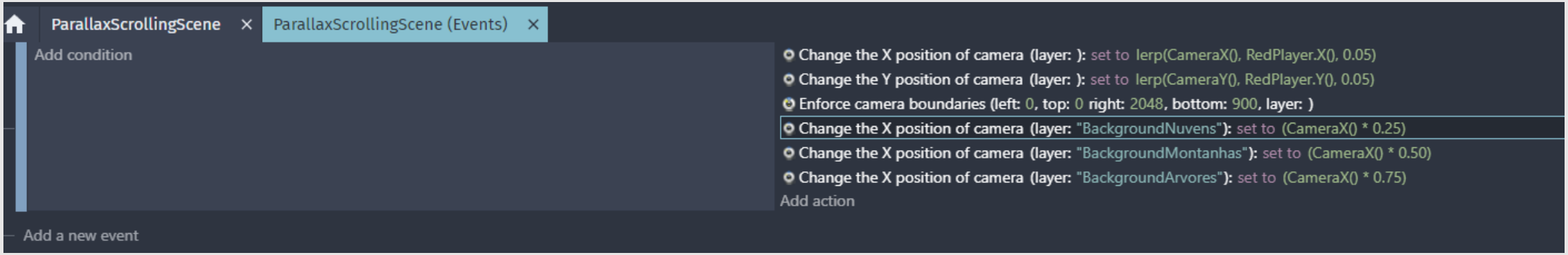


Figura: Código do efeito parallax completo.

Criando o código para o efeito Parallax

- Ao executarmos o projeto, verificamos ainda um problema relativo a posição do eixo X dos backgrounds, já que, cada um deles se movimenta a uma velocidade diferente.

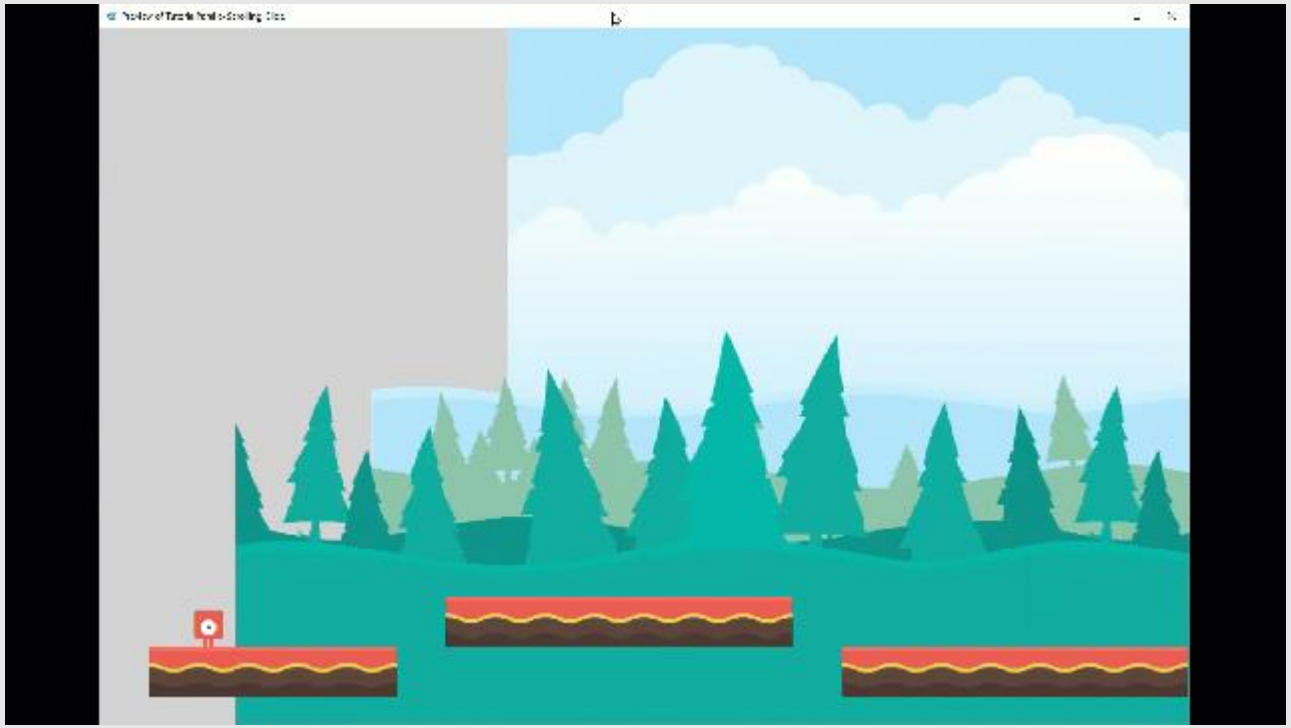


Figura: Efeito Parallax Scrolling com problemas de posicionamento.

Criando o código para o efeito Parallax

- Para revolver esse problema devemos ajustar a largura do plano de fundo na cena. Você pode ajustar na própria cena até encontrar um tamanho que se ajuste adequadamente ou podemos aplicar simples cálculos matemáticos para resolver o problema.
- Utilizamos o evento **At the beginning of the Scene** para ajustar a dimensão dos planos de fundo. Aumentamos a dimensão relativo ao tamanho da resolução de tela.

Figura: Ajustando a dimensão dos backgrounds

Criando o código para o efeito Parallax

- Por fim, nas ações que alteram a posição no eixo X dos planos de fundo, ajustamos a posição deles relativo a metade da resolução de tela para que não aparecem as áreas cinzas do cenário.

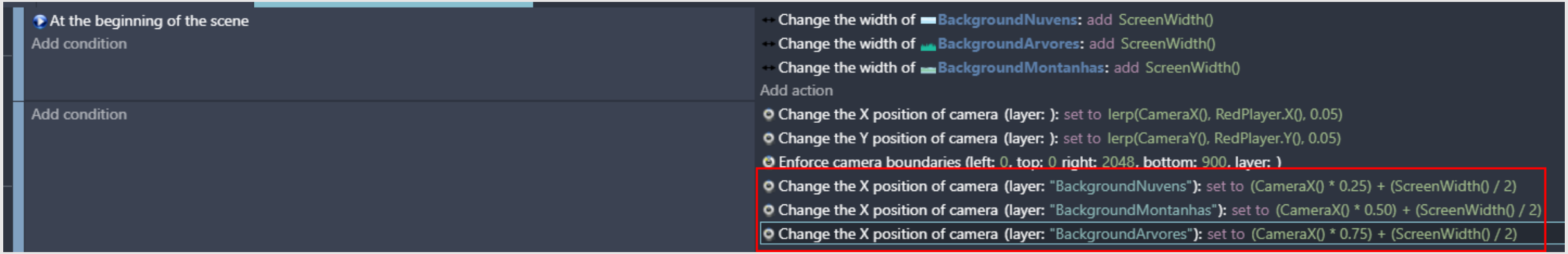


Figura: Ajustando a posição dos backgrounds na cena

Criando o código para o efeito Parallax

- Resultado final do efeito parallax:

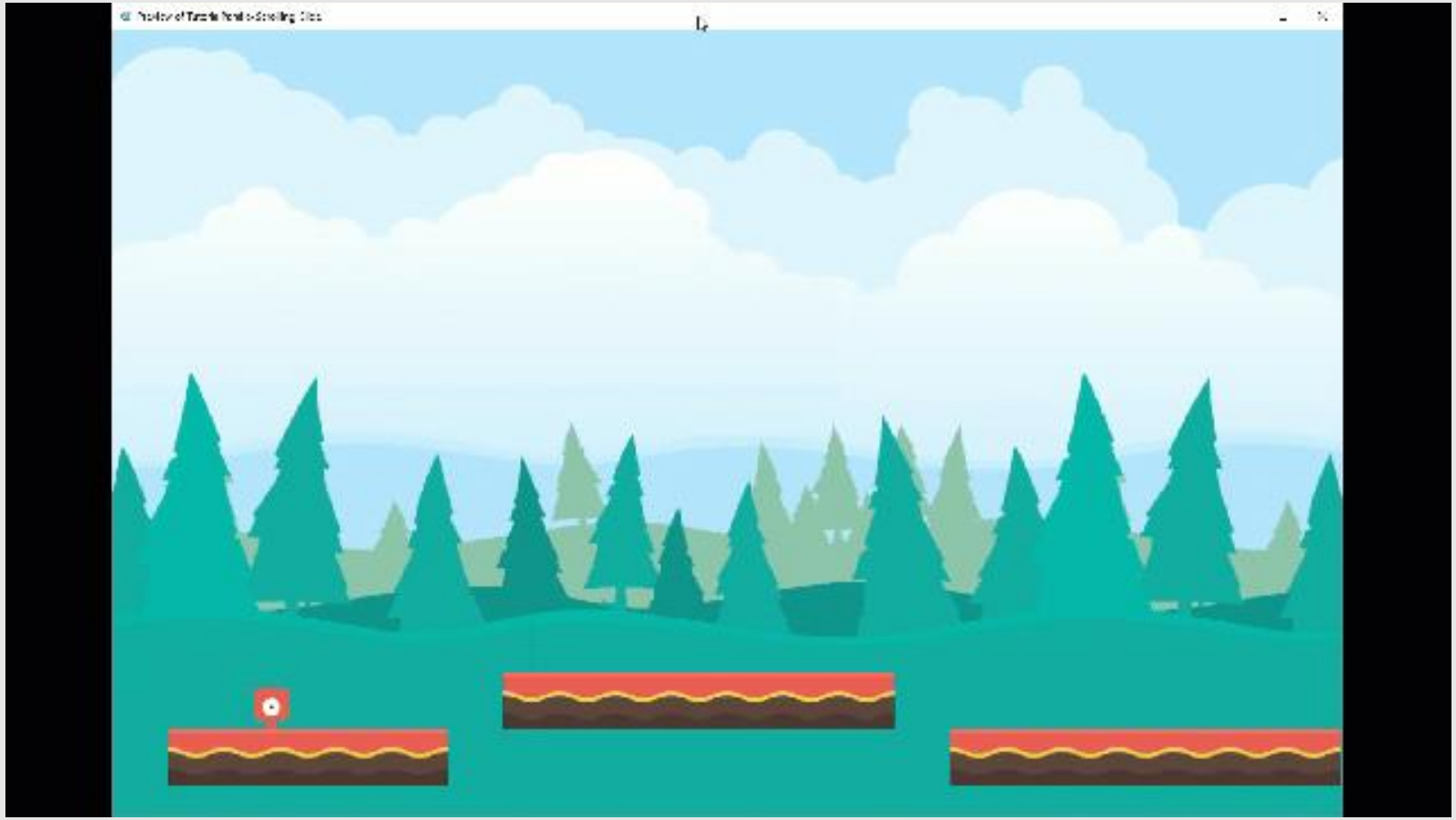
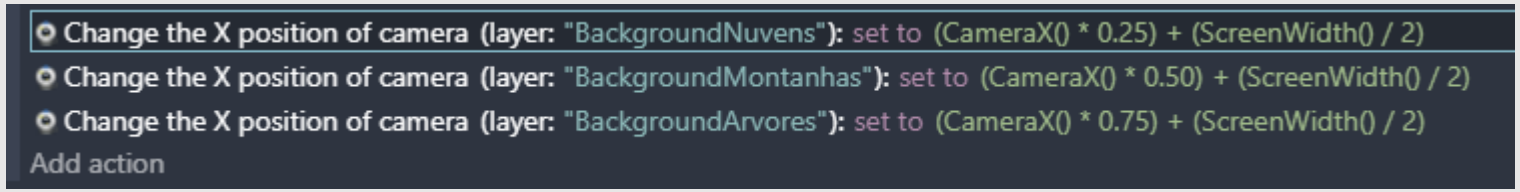


Figura: Efeito Parallax Scrolling

4. Considerações finais

Considerações finais

- A velocidade de movimento dos backgrounds dependerá da quantidade de planos de fundos existentes e também da resolução do jogo.



- Neste caso, não existe um valor correto, esse valor **DEVERÁ** ser testado pelo desenvolvedor para encontrar uma velocidade de movimento ideal.
- De formal geral, comece com valores de porcentagem entre 0 e 1. Podendo também ser utilizados valores acima de 1.