02

Ajuste ZIndex

Transcrição

[00:00] Bom, a gente vai continuar nosso projeto da invasão zumbi, e hoje a gente vai começar fazendo uma parte de correção de um bug, na parte gráfica. Então, qual é esse bug? Se eu vier aqui no meu jogo, der um play, a gente vai ver que conforme a jogadora anda, os zumbis vão aparecer, tem o gerador de zumbis com eles aparecendo, e se a jogadora vai para o meio deles, ela fica soterrada. Independente de onde ela estiver no nosso quadro, ela vai ficar por baixo dos zumbis. Então, a gente não consegue ver a jogadora, o que não é legal.

[00:30] O que seria legal é, dependendo da posição de onde o zumbi está, então, se ele estiver ou para baixo dela, ou se ele estiver para cima dela, ela vai ser renderizada ou para cima ou para baixo dele. Então, depende da posição deles ali no Y. Bom, vamos desligar o áudio do nosso jogo, porque ele está um pouco alto. A gente não está verificando áudio agora, então a gente não precisa disso. Então, eu vou aqui no nosso áudio e vou desabilitar todos os nossos áudios, a trilha sonora, o áudio do coração e o áudio da morte do zumbi, o áudio do tiro da jogadora não porque eu teria que mexer em códigos, ele é mais complicado, então eu vou deixar.

[01:04] A gente vai precisar arrumar então esse bug, da nossa jogadora sendo soterrada pelos zumbis. Mas como que a gente vai fazer isso? A gente já discutiu que a ordem de renderização da Cocos depende da hierarquia então, então ela depende de quem está primeiro ou quem está depois, depende da ordem em que a gente colocou as coisas na cena.

[01:22] Só que o jogador e os zumbis estão desenhados dinamicamente, então como que a gente vai fazer para ajustar isso? No nosso componente load, então se eu vier aqui, a gente já discutiu isso, todo objeto tem um componente load, é o componente obrigatório aqui da Cocos esse primeiro componente. Ele tem uma propriedade chamada Z index, o Z index é justamente o que fala para o computador como que ele tem que renderizar aquele objeto, ou qual a ordem de renderização.

[01:49] Por padrão, ele segue a hierarquia, então se a gente vier aqui na nossa load trion, na hierarquia do nosso projeto, eu vou fechar tudo. A gente tem aqui, o primeiro, o Z index do Canvas aqui por exemplo, desse objeto Canvas, é 1, porque ele é o primeiro objeto da hierarquia. Do cenário é 2, do personagem é 3 e assim por diante. Ele sempre segue essa ordem. Só que assim, dentro do cenário eu tenho ainda o objeto chão, o loja, o mecânico, e ele estão sendo desenhados na ordem que a gente quer.

[02:14] O chão está por baixo de tudo, a loja está por cima, todos os prédios estão por cima. Então, como que funciona? Bom, o cenário, ele tem um Z index de 2, porque dentro da raiz da nossa cena, ele é o segundo objeto. O chão, ele tem um Z index de 0, porque dentro do cenário ele é o primeiro elemento, e a loja de 1, o mecânico de 2, e assim por diante. Mas como que a gente vai fazer então para alterar esse Z index? A gente pode alterar o código. Ok, então a gente pode alterar ele por código, mas o que a gente quer fazer?

[02:44] A gente quer que quando a jogadora tiver para cima do zumbi, ela seja renderizada atrás dele. Quando ela tiver para baixo dele no Y, - para cima e para baixo a gente sempre está falando do Y - ela esteja para frente dele. Então, vamos ver isso com mais calma. Bom, aqui eu tenho a jogadora e o zumbi numa numa situação hipotética, no fundo branco para a gente ver a posição relativa entre os dois. Então, a jogadora está para cima do zumbi e o Y dela, a posição Y dela, vamos falar que uns 300 pixels, e o zumbi está ali no 200.

[03:15] Nesse caso, a gente quer falar que o Z index do zumbi é maior do que o da jogadora, porque tem que ser desenhado para frente dela. Se ele tivesse em cima dela, se alguma imagem, por exemplo a cabeça do zumbi, tivesse ali perto do pé dela, ele teria que ir para cima dela. Para isso, a gente pode falar que a posição do Z index jogadora, é

menos 300, e a posição do Z index do zumbi é -200. Isso vai funcionar porque -200 é maior do que -300, então a jogadora vai ser desenhada primeiro e depois o zumbi.

[03:46] E aí como que isso ficaria? Bom, se a gente aproxima o zumbi, aí ele vai ter uma posição de 290 e o Z index de -290. O zumbi vai ser desenhado para frente da jogadora, então a cabeça dele cobre parte da imagem da jogadora. Já quando ele vai para trás, se ele sobe um pouco mais, o Y dele vira 310. O Z index dele vai ser -310, então a jogadora vai começar a cobrir a imagem do zumbi, isso fica mais natural, fica mais fácil de entender o que está acontecendo na nossa cena.

[04:15] Você percebeu que a gente está usando a posição Y para fazer o Z index. Então, a gente tá, ao invés de pegar a posição Y normal, a gente está transformando ela no negativo. Então, antes estava como 290 o zumbi, a Z index estava -290, quando ele fala para cima, o Y dele virou 310 e o Z index virou -310. Sendo isso, é muito fácil de a gente arrumar isso, a gente arrumar esse Z index do zumbi e da jogadora.

[04:42] A gente só precisa de um script que faça isso, então vamos voltar aqui na Cocos, vou na minha pasta de scripts, a gente clica com o direito, vamo criar um novo java script e esse javascripts vai fazer o ajuste dos Z index. Então, vamos chamar de ajuste Z index, vamos abrir ele para a gente começar a editar. Bom, a Cocos de novo vem com um monte de comentário que a gente não vai precisar, então eu posso jogar tudo isso fora e quando que eu quero alterar o meu Z index da jogadora e do zumbi?

[05:10] Sempre que eles se moverem, afinal eu estou usando a posição Y deles. Então, toda vez que eles se moverem ou todo frame eles podem se mover, eu vou querer ajustar isso. Então, a gente vai usar o método update e aqui a gente vai precisar capturar o Z index e configurar ele para ser menos a posição Y. Então, eu posso escrever this.mode.zindex, que é a propriedade que a gente quer alterar, vai ser igual a -this.mote.y, que é a posição que a jogadora em Y.

[05:39] Então, ajustando isso, a gente consegue agora, a gente vai ter, a gente vai falar para Cocos, qual que é a ordem de renderização das coisas. Então, vamos voltar lá na Cocos, vamos pegar a nossa jogadora, nossa personagem, e vamos arrastar aqui o Z index para lá, então agora ela já tem o Z index. Só que eu preciso fazer isso com os zumbis também, então eu vou trazer o prefab deles aqui, aliás, eu posso clicar duas vezes para alterar um prefab. A gente vem aqui para a aba prefab, a gente consegue ver nossa cena e eu consigo trazer também o script para cá, o ajuste do Z index para o meu zumbi.

[06:12] Então, ele já está aqui com o meu ajuste do Z index, eu posso clicar em salvar, fechar, a gente já alterou o prefab e a gente pode ver agora como que isso vai ficar na nossa cena. Então, eu vou esperar a Cocos ler, nossa jogadora está aqui, o Z index dela está sendo atualizado o tempo inteiro. Quando ela está para cima do zumbi, ela é desenhada por baixo, e quando ela está para baixo dele, ela é desenhada por cima. Só que está um pouco estranho.

[06:37] Porque ela demora um pouco para passar dele, ela não está exatamente para cima dele quando ela está desenhada por trás, ela também não está exatamente para baixo, quando ela está desenhada por cima. O que acontece, a nossa jogadora ela tem uma âncora, ela tem um ponto do mapa que ela está sendo desenhada. Por exemplo, se eu coloco a minha jogadora no ponto zero zero, na posição zero zero do nosso mapa, a gente vai ver que ela é desenhado.Alías, vamos tirar o meu cenário, porque a gente consegue ver mais fácil a nossa jogadora.

[07:07] Ela é desenhada, a cara dela dela está no ponto zero zero, é onde está o gizmo, o gizmo mostra para gente onde é o ponto que a gente escolheu, o ponto zerozero. Não é isso que a gente quer, quando a jogadora vai passar do zumbi, a gente tem que ver pelo pé dela, e como que a gente faz isso na Cocos? A gente precisa mudar a âncora dela, a âncora fala para Cocos em que ponto da imagem a gente tem que colocar o nosso gizmo, a gente tem que colocar a nossa posição.

[07:32] Então, por enquanto, a gente tem uma âncora de meio no X e meio no Y, se a gente colocar 0 no Y, a gente vai ver, eu pedi para ele atualizar, a gente vai ver que a nossa jogadora sobe, e agora o nosso gizmo está no pé dela, o que

isso significa? Ele vai pegar a posição 0 pixels da imagem e vai colocar na posição do 0 no position, na propriedade position. Então, com isso, a nossa jogadora agora vai verificar a posição dela sempre pelo pé dela, e não pela cara dela.

[08:05] Então, agora a gente precisa atualizar a nossa caixa de colisão, nosso box collider, porque ela ficou fora aqui da imagem da jogadora, então vamo editar ela e subir, então a gente pode tirar a edição, e a gente precisa arrumar a âncora do zumbi também, para ele verificar a posição dele sempre pelo pé e não pela cabeça dele, porque ele está do mesmo jeito. Vamos salvar, eu vou abrir de novo o prefab do zumbi, a gente pode ver que o gizmo, a posição dele, está sendo marcada na cabeça dele, eu não quero isso, eu quero que ele seja marcado no pé, então eu vou por a âncora Y = 0.

[08:39] Se eu pusesse a âncora X= 0 também, a gente pode ver que a posição dele vai ficar no canto inferior esquerdo, e se eu pusesse 1 e 1 na âncora, ele vai ficar no superior direito. A gente consegue controlar onde que a gente quer que essa imagem seja desenhada só através da âncora, sem mudar a posição do nosso zumbi, da nossa imagem.Então, a gente quer 0.5, a gente quer meio no X e 0 no Y, para que a verificação da posição dele seja sempre no centro dele e no pé. E agora a gente também precisa atualizar a nossa box collider.

[09:13] Então, box collider, vamos clicar em editar, subir ela aqui, pronto, estamos salvando o nosso prefab, e a gente pode dar um play, e a gente consegue ver agora, opa, eu tirei o meu cenário, vamos voltar o cenário lá para a gente conseguir ver o que a gente está fazendo, vamos ativar o cenário. E a nossa jogadora não consegue andar porque eu coloquei ela no zero zero em cima do motel, não é o motel que eu quero. Eu quero pegar a jogadora, personagem, e é a jogadora que eu quero mover.

[09:47] Então, agora sim, vamos testar, ela consegue andar porque ela está fora da caixa de colisão do prédio. Você pode ver que como a gente estava atirando, fazendo os tiros nascerem na posição da jogadora, ele saiu agora do pé dela e não do centro. Mas quando ela está por cima ou por baixo dos zumbi, a gente consegue ver, fica muito mais fácil da gente entender o que está acontecendo. Então, a gente já conseguiu tirar esse bug, agora a gente vai continuar o desenvolvimento do nosso jogo.

Arma Disparando 00:10:09 [00:00] Agora que a gente já arrumou o problema do Z index na nossa jogadora, agora que ela não é mais soterrada pelos zumbis, a gente volta aqui para o nosso jogo, e vamos testar? A gente tem outro problema, porque a gente mudou a âncora da jogadora, ou seja, a imagem dela é desenhada em outro ponto, e agora todos os nossos tiros, eles estão saindo do pé dela, o que é muito legal.

[00:19] O que a gente quer fazer, é que o tiro saia da arma da jogadora, afinal ela tem uma arma que atira, não é o pé dela que está atirando, é a arma. Então, o que a gente precisa fazer, é algum modo de a gente tirar a posição do tiro da posição da jogadora, e mover ele para a posição da ponta da arma, onde que a arma vai estar. Então, como que a gente vai fazer isso?

[00:39] Vamos voltar aqui para Cocos e vamos pensar, no nosso script da jogadora, então se eu abrir o script dela aqui, no método atirar a gente está pegando e falando para o tiro ser inicializado no pai da jogadora, ou seja, dentro daquele objeto personagem, ele vai ser criado um novo objeto tiro ali dentro, na posição da Jogadora. Então, se eu pegar aqui, aí está quebrando minha linha, vamos dar um enter para ficar mais fácil de ver na posição da jogadora.

[01:07] Então, eu estou pegando o pai da jogadora e a posição dela, só que a posição dela é a posição aqui onde está no gizmo, então é o pai dela. A gente não quer isso, a gente quer que ele saia de outro ponto, por exemplo, na verdade quando ela está olhando para baixo, o pé dela está praticamente aonde está a arma, só que quando ele está para o lado não é mais essa posição. Então, a gente precisa de alguma coisa que se mova e siga a posição da arma. Vamos fazer então um novo objeto vazio, que vai servir da posição da arma, então a posição da arma vai ser esse novo objeto vazio que a gente vai criar.

[01:42] Então, eu vou aqui na jogadora, vou criar um novo objeto vazio dentro dela, a gente chamar ele de posição da arma, e esse objetivo vazio eu posso mexer ele livremente. Então, vamos deixar ele aqui em cima da cabeça da jogadora,

e vamos buscar ele e fazer com que o meu tiro nasça em cima desse objeto, vamos salvar minha cena

[01:58] Então, eu vou querer na verdade, ao invés de falar o this.load.position, para passar para o meu disparo, para o meu tiro, eu vou querer passar o this.posição armapositiom. Então, a gente vai pegar o load, o componente load, e vai pegar a posição dele ponto position. Então, essa posição dar armva vai ser um objeto que a gente vai criar, vai ser uma propriedade.

[02:28] Então, a gente vai lá nas propriedades e vamos criar uma nova propriedade posição arma. Aliás, ela vai ser privada, ela não vai ser pública, porque só o jogador é que vai acessar ele. Ele vai ser um tipo cc.load, aliás, vamos organizar isso, porque eu tenho o meu tiro aqui embaixo, vou criar uma propriedade pública, eu tenho misturado aqui, então vamos aproveitar para organizar o nosso código.

[02:52] Eu gosto de deixar todas as variáveis públicas já juntas, então eu tenho a minha vida máxima e o meu tiro ali já ali no começo. E todas as variáveis que são privadas, que tem underline, a gente deixa aqui embaixo porque aí a gente consegue ver uma diferença, a gente consegue achar mais fácil.

[03:09] Eu sei que se eu for mexer numa propriedade privada, eu vou procurar nesse bloco, e se eu for mexer numa propriedade pública, eu mexo no bloco de cima, quando a gente tem as que estão sem underline. Bom, então agora eu tenho a minha posição da arma, a posição da arma ela é uma propriedade privada, eu pus lá embaixo sem o underline, então a posição da arma ela é uma propriedade privada e o que eu quero fazer? Eu quero buscar essa posição.

[03:31] Mas quem que é a posição da arma? A posição da arma é o primeiro filho, é o objeto que é o primeiro filho da minha personagem, da minha jogadora. Então, como que eu faço isso? Ir no método download, que é onde a gente busca todas as referências que a gente quer, com todas as dependências que a gente tem. Eu posso falar que o this.posição arma vai ser igual ao this.load.children, que são os filhos da minha jogadora, children são filhos em inglês. Só que children é no plural, então isso daqui é uma ray, e como era a primeira posição, o primeiro filho dela, é a primeira posição da ray.

[03:35] Todos os rays no java script começam com 0, então eu posso pegar e falar que a minha posição da arma, é o meu primeiro filho da minha jogadora. Agora a Cocos vai lá, quando eu começar o script, vai buscar aquele filho e vai trazer para essa posição da arma. Então, a gente já tem a posição da arma, e aí a gente já colocou ela aqui no nosso atirar. Vamos testar para ver como isso ficou? Lembrando que a posição da arma, se a gente for aqui na jogadora, está na cabeça dela, ele tem que começar a atirar para cima da cabeça jogadora.

[04:38] Então, vamos testar, a nossa jogadora está aqui, e eu estou clicando e na verdade ele parou totalmente de atirar, vamos ver se a gente tem algum erro no console? Aparentemente eu não tenho nada, eu não tenho nenhum erro no console. O que pode estar acontecendo então? Vamos voltar lá, então ele deveria estar atirando, eu estou falando que a posição do meu tiro vai ser igual à posição da arma.

[05:04] Então, a posição da arma é -1,7 e 75 no Y, -1,7 no X e 75 no Y. Mas essa não é a posição da arma real, porque ela está dentro da jogadora e a jogadora está no X 128 Y 153, a gente consegue ver aqui. Então, o que está acontecendo? Eu estou colocando o meu tiro lá no começo do meu mundo. Eu estou pegando meu tiro e colocando ele aqui perto 00, que é perto dessa área. Como ele está dentro do prédio, o tiro já nasce, já colide com prédio, e ele some, então a gente nem chega ver o tiro aparecendo, ele não aparece onde a gente quer.

[05:41] E por que a posição da arma, por que esse objeto, ele está dentro, ele está com essa posição estranha? Porque a posição que ele tem, esse load.position da arma, como ele é filho da jogadora, é a posição local dela, ou seja, é como se o pé da personagem, a posição da personagem, fosse o 00 para a posição da arma. Então, para a arma ela está 75 pixels para cima do 00, ou seja, 75 pixels para cima do pé da jogador, e -1,7 para esquerda da jogadora, ou seja -1,7 em X do pé da jogadora, que é o00 dela. O que a gente vai fazer então?

[06:17] A gente vai somar as duas posições, eu vou somar a posição da jogadora com a posição da arma, porque aí eu vou pegar uma posição global da minha arma. Então, a gente pode vir no meu código, e ao invés de pegar a posição da arma sozinha, eu vou somar, então eu vou por um position.add junto com o this.load.position, que é posição da minha jogadora. Então, eu estou somando a posição da arma com a posição da jogadora, aí eu vou ter a posição real, a posição global dela.

[06:51] Vamos salvar e aí vamos testar, e agora o tiro tem que está saindo da cabeça da jogadora. Agora ele está saindo exatamente onde está a minha posição da arma, aquele objeto posição da arma. Só que ainda a gente não conseguiu o que a gente quer, porque ele não está variando conforme a minha jogadora se move, o que a gente precisa fazer? A gente precisa de alguma maneira de alterar isso junto com a animação da jogadora.

[07:15] Então, já que é junto com a animação da jogadora, vamos ver nossa timeline, vamos ver se apareceu alguma coisa nova. Então, eu vou selecionar a minha personagem e vou na timeline, aí a gente pode ver se eu clicar aqui na parte de edição, que eu tenho uma posição da arma, aqui na minha timeline, e eu consigo adicionar a animação dentro desse objeto. Então, a animação da jogadora, afeta os filhos dela, eu consigo mudar as propriedades dele, então eu vou pegar e vou em todas as animações dela, que ela muda de posição, eu vou colocar a posição da arma naquela animação.

[07:48] Então, o parado para baixo, se eu tirar um pouco aqui o zoom, a posição da arma é aqui embaixo. E aliás eu tenho que adicionar, então eu vou selecionar a posição da arma, porque eu preciso adicionar a propriedade que eu quero alterar, vou adicionar a position. Vou clicar no mais para ele criar um quadro chave, e aí nesse quadro chave eu consigo alterar a minha posição da arma

[08:07] Você vê que ele cria uma bolinha aqui aonde vai ser a posição exata, então a gente vir e trazer para ponta da arma. Só que isso aqui é só uma animação, então eu tenho que fazer isso para todas, então a gente vai fazer isso para todas as animações dela, andando e parada, porque são todas essas que ela move, e aí a gente consegue testar e ver que o nosso tiro agora vai estar momento junto com a jogadora. Então, vamos salvar, e eu vou pedir para acelerar o vídeo porque é um processo bem repetitivo.

[08:31] Então, o mesmo processo olha, eu vou selecionar a minha animação, andar para esquerda, eu vou clicar na posição da arma, no objeto posição da arma, vou adicionar propriedade position e aí eu vou arrastar aqui para onde eu quero. Salvar e ir para próxima, então eu vou acelerar daqui a pouco a gente volta. Bom, agora que eu já terminei todas as animações, a gente pode testar.

[09:27] A mesma coisa, eu peguei todas as animações, posicionei onde que eu quero a posição da arma naquela animação através do animator, e agora a gente pode ver que sempre que a gente está movendo, o nosso tiro sai da posição daquela arma, da posição onde eu selecionei. Então, nosso jogo está muito melhor, agora o tiro não sai mais do pé da jogadora, fica muito mais legal. É um detalhe simples, mas que afeta muito a jogabilidade, a sensação dos jogadores desse jogo, porque nenhum tiro deveria sair do pé da jogadora ou da cabeça dela, afinal ela tem uma arma. Então, a gente arrumou esse detalhe, arrumou já o Z index dela, ela não é mais soterrada, está muito legal e a gente consegue continuar para a próxima sua parte.