

Game Over pela Vida

Transcrição

[00:00] Vamos fazer então game over acontecer pela nossa nova variável de vida? Vou abrir o código do controle inimigo e aí essas três linhas aqui são as linhas que davam o game over anteriormente.

[00:11] O que eu vou fazer? Vou tirar o comentário dessas três linhas, estou tirando o barra barra aqui da frente e vou recortar elas aqui, vou dar um control x, então abri aqui o controle o inimigo, estou tirando do método do ataca jogador as três linhas do game over, e vou abrir o script de controlar jogador aqui.

[00:31] No método de "tomardano", eu vou jogar essas três linhas. Como eu já estou aqui no meu código do jogador, eu não preciso vir aqui "jogador.GetComponent" controla o jogador, porque eu já estou no script de controle o jogador. Então, é só utilizar o texto e só utilizar o vivo.

[00:52] Então pronto, é as mesmas três linhas de game over. Se eu salvar isso aqui, vamos ver o que que está acontecendo, salvei, vamos dar o play. Zumbi bateu, eu perdi novamente e eu recomeço. Por que eu perdi com um ataque só? Porque quando eu estou rodando o método de "tomardano" aqui no ataque o jogador, estou rodando o método "tomardano" e eu rodo ele, aí eu tiro vida e dou game over.

[01:17] O que eu tenho que fazer aqui para dar o game over só quando a vida chegar a 0? Eu vou fazer um if aqui, eu vou testar isso. Se a vida for igual a zero, faz isso aqui, e vou jogar essas três linhas para dentro do if. Salvo isso aqui e vamos dar um play, vamos lá, vou clicar aqui no jogador para a gente ver a vida, a vida é 100, zumbi bateu 70, aí chegou em negativo e não deu game over. Por quê isso está acontecendo?

[01:55] Porque que aqui eu fiz, olha, aqui a vida tem que ser exatamente 0. Só que se ele perde 30, de 30 em 30, ele vai perder 30, aí ele vai ficar com a vida 70, aí depois 40, depois 10 e depois já vai para -20. Então, toda vez que você quiser fazer esse teste de, se a vida for 0, ao invés de testar se a vida for exatamente igual a zero, com igual igual, a gente testa se a vida é menor ou igual a zero.

[02:27] Então se a nossa vida for menor ou igual a zero, aí sim, eu faço o nosso game over. Vamos dar o play aqui de novo, olha lá, vamos testar, chegou em menos 20 aqui, agora perdeu. Clica para Recomeçar. Já está funcionando. Só que eu preciso agora dessa variável de vivo aqui ou não precisa? Na verdade eu não preciso dela, por que? Porque a vida menor igual a zero aqui está fazendo exatamente isso aqui, então eu vou lá em cima e vou apagar a nossa antiga variável de vivo e vou salvar aqui.

[03:06] Nosso código vai quebrar em alguns lugares, você já pode ver pelo pontinho vermelho aqui que ele quebrou em dois lugares. Vamos nesses lugares e aqui eu estou falando: se vivo for igual ou igual a falso, ou seja, se vivo for falso, o que eu vou fazer? Espera que eu aperte um botão e aí reinicia o jogo.

[03:22] Então a invés de vivo ser falso eu vou testar se a vida é menor ou igual a 0. Vai acontecer a mesma coisa, só que eu estou utilizando uma nova variável numérica aqui. Se a vida for menor ou igual a zero, faz isso aqui, espera que ele clique com o botão e reinicia o nosso jogo. E essa linha aqui de baixo, eu não preciso mais, porque eu não estou utilizando mais a nossa variável de vivo.

[03:50] Agora eu já estou dando game over pela variável vida, só que quando eu chamo aqui no ataca jogador, eu vim aqui e chamo o método "tomardano" e lá no jogador eu dou o dano nele que aqui eu estou dizendo que é 30.

[04:07] Só que se eu quiser definir um dano lá no zumbi - porque dano é uma parte do zumbi - o zumbi é que está atacando o jogador, então não faz sentido eu dar o dano aqui no jogador, faz sentido eu definir o dano lá no zumbi e só dizer para o jogador, você tem que tomar 30 de dano aqui na sua vida.

[04:26] Como que a gente pode fazer isso com os métodos.? A gente pode utilizar os parâmetros. A gente já utilizou parâmetros antes no script da bala aqui no on trigger enter, a gente utilizou parâmetro collider objeto de colisão. Só que esse parâmetro a Unity que definiu para a gente que quando o objeto colidir eu vou ter um objeto de colisão.

[04:48] No objeto collide eu tenho o objeto com quem eu colidi. Aqui não, aqui a gente que criar o nosso parâmetro. E aí vocês já viram lá que o parâmetro em geral definido aqui, e aí vocês já viram lá que os parâmetros são definidos dentro de parênteses e eles são bem parecidos com a variável. Na verdade eles são variáveis que a gente vai utilizar somente dentro desse método.

[05:09] Então vou criar um parâmetro aqui inteiro dano, porque inteiro? Porque a nossa vida é inteira então faz sentido eu tirar dela um valor inteiro, e vou dar um nome para esse parâmetro, o nome vai ser dano, então tem o parâmetro inteiro com o nome dano. Aí aqui dentro eu uso essa variável. Vou falar, olha, ao invés de passar 30 de dano, eu vou passar a variável dano.

[05:34] Aí como é que eu sei qual que é o valor de dano aqui? Como que eu vou saber de onde vem o valor de dano? Quando eu chamo o método. Você vê aqui que no ataca jogador já deu até erro, ele fala , está faltando um parâmetro aqui. Como você passa o parâmetro? Você fala que dano é igual a 30, algo assim? Não, na verdade você só vai passar o número aqui.

[05:57] Só vou passar o 30 e aí a Unity já é bem espertinha, ela já vai entender que você jogou 30 e eu tenho uma variável aqui que é inteira, que casa com esse 30, então o dano na verdade já é 30. Aí ele pega e faz tudo que você tava acontecendo antes, a vida menos o igual a 30 aqui, porque o dano agora vale 30.

[06:18] Então, o código já é espertinho, ele já interpretou, você passou um número aqui, eu tenho um parâmetro que é o número, joga esse número nesse parâmetro e ele vai utilizar dentro do código. Só que, suponhamos que eu não quero sempre passar 30 de dano. Isso aqui vai estar funcionando se eu der um play, está tudo ok.

[06:36] Mas vamos supor que eu quero fazer um dado meio variável algum tipo 20 a 30 de dano. Então, eu posso criar uma variável aqui dentro do método ataca jogador que vai guardar o dano que eu quero passar e eu vou usar o random para gerar um valor aleatório, eu não falei que eu vou utilizar um valor de 20 a 30? Vou utilizar a classe da Unity que chama random, que gera valores aleatórios para mim, entrando nela vou utilizar o método range onde eu passo um valor mínimo e um valor máximo, eu quero que a variável dedão receba um número aleatório de 20 a 30.

[07:16] Então toda vez que eu rodar o método ataca jogador, ou seja, toda vez que o meu personagem atacar, eu vou vir aqui gerar um número aleatório de 20 a 30, guardar esse valor na variável de dano que agora é do meu inimigo e vou passar esse valor aqui. Apesar dessa variável aqui e essa aqui ter o mesmo valor, elas não são a mesma coisa, isso aqui podia ser qualquer coisa aqui, desde que esse aqui fosse igual, estaria tudo ok.

[07:47] Porque aqui eu estou falando, olha, eu tenho uma variável que é o dano, passando ela como parâmetro aqui - aí eu falei, o código já é espertinho, ele vai saber, ah, o valor de dano é 25 - eu tenho um parâmetro que chama qualquer coisa aqui que recebe o número então joga nesse parâmetro o número 25 e faz o que tem que executar aqui dentro do nosso método.

[08:11] Apesar delas aqui como eu tinha antes, terem o mesmo nome, elas não são a mesma variável. Dessa forma, o que eu estou falando? Olha, eu posso ter um dano aqui no zumbi, o zumbi gerou um dano para mim e eu estou passando esse dano para o jogador. Então, assim a gente pode comunicar e utilizar métodos de uma forma muito mais

interessante. Aí a gente evita também uma outra prática bem legal quando a gente está fazendo o jogo, que é ter tudo dentro do update.

[08:39] Aí eu não teria aqui, por exemplo, quando o zumbi ataca, que esperar um ataque no update e eu ter que dar um dano dentro update. eu estou usando essas chamadas de método justamente para não ter que ter tudo dentro do update, porque é uma boa prática você não utilizar tanto assim o update senão você vai ter muita coisa acontecendo a todo frame.

[08:57] Aqui não, só quando zumbi ataca que eu vou lá e rodo o método de tomar dano. E aí agora se gente salvar tudo e testar, o que está acontecendo? Vamos dar um play aqui, vou clicar no jogador, a variável está tirando aqui, você viu que o valor está 15 aqui, já está gerando um número aleatório.

[09:19] Vamos de novo, 100, zumbi me ataca, 77, gerou um número aleatório então de 20 a 30 aí. Aí ataca de novo 54, 32, 21. Aí roda o nosso game over, então agora a gente tem dinamismo bem maior e o código bem mais definido nessa parte de tirar a vida ou não do jogador.