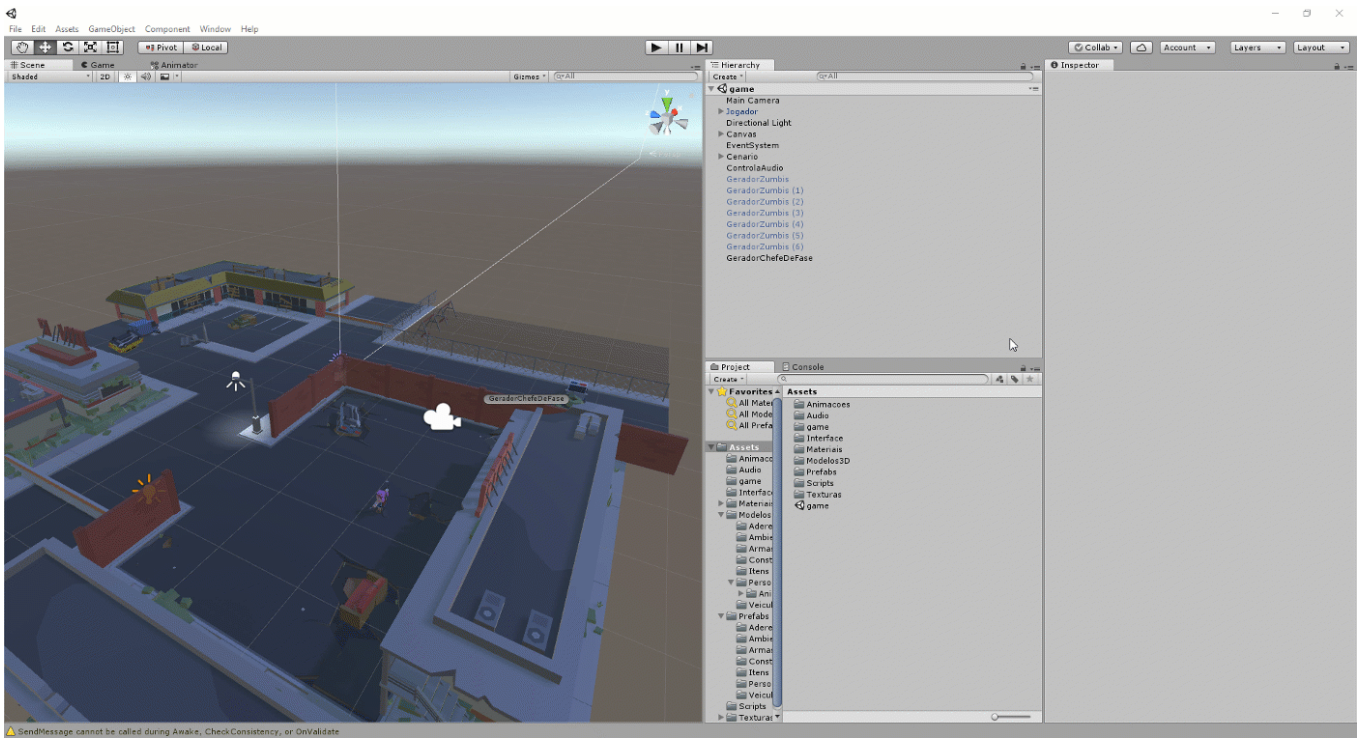
05

Gerador do Chefe

> Você pode fazer download do projeto Unity como ele está ao final desta aula [clicando aqui (https://github.com/alura-cursos/unity-shooter-4/raw/master/Final_Aula02.zip)](https://github.com/alura-cursos/unity-shooter-4/raw/master/Final_Aula02.zip)

Nosso chefe agora está surgindo, mas tem uma coisa chata: ele sempre sai do mesmo lugar, então ele pode aparecer enquanto o jogador estiver vendo. Seria interessante ele ser mais imprevisível e vir de lugares diferentes certo? Vamos fazer algumas alterações.

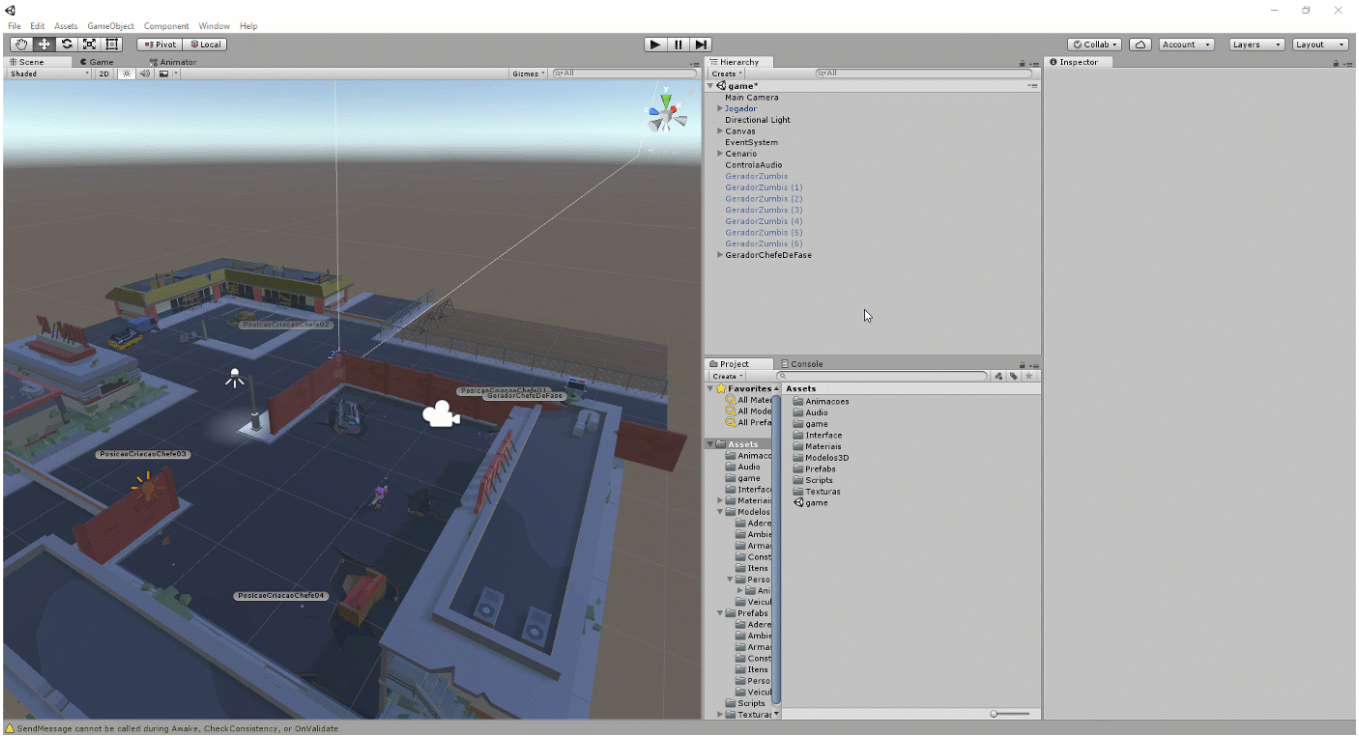
Primeiramente, vamos criar alguns objetos vazios e colocar em alguns lugares do cenário:



Agora vamos criar uma variável diferente para guardar essas posições. Essa variável é conhecida como `Array`, que é um conjunto de elementos.

```
public Transform[] PosicoesPossiveisDeGeracao;
```

Agora vamos preencher essa variável com as nossas posições possíveis.



Após isso, vamos criar um método que passa por cada uma dessas posições possíveis e verifica a que está mais distante do nosso jogador, para criar o Chefe utilizando `foreach` e retornando a posição com a maior distância.

```
Vector3 CalcularPosicaoPossivelMaisDistanteDoJogador ()
{
    Vector3 posicaoDeMaiorDistancia = Vector3.zero;
    float maiorDistancia = 0;
    foreach(Transform posicao in PosicoesPossiveisDeGeracao)
    {
    }
    return posicaoDeMaiorDistancia;
}
```

Para fazer as verificações, vamos em cada uma das posições calcular a distância. Vamos verificar se essa distância é maior do que a distância guardada anteriormente, e se podemos salvar essa posição como a maior e continuar a verificação.

```
Vector3 CalcularPosicaoPossivelMaisDistanteDoJogador ()
{
    Vector3 posicaoDeMaiorDistancia = Vector3.zero;
    float maiorDistancia = 0;
    foreach(Transform posicao in PosicoesPossiveisDeGeracao)
    {
        float distanciaPosicaoJogador = Vector3.Distance(jogador.position, posicao.position);
        if(distanciaPosicaoJogador > maiorDistancia)
        {
            maiorDistancia = distanciaPosicaoJogador;
            posicaoDeMaiorDistancia = posicao.position;
        }
    }
    return posicaoDeMaiorDistancia;
}
```

Agora no `Update` podemos chamar este método como a posição para criar o Chefe.

```
void Update ()
{
    if(Time.timeSinceLevelLoad > tempoParaProximaGeracao)
    {
        Vector3 posicaoDeCriacao = CalcularPosicaoPossivelMaisDistanteDoJogador();
        Instantiate(ChefePrefab, posicaoDeCriacao, Quaternion.identity);
        tempoParaProximaGeracao = Time.timeSinceLevelLoad + TempoEntreGeracoes;
    }
}
```