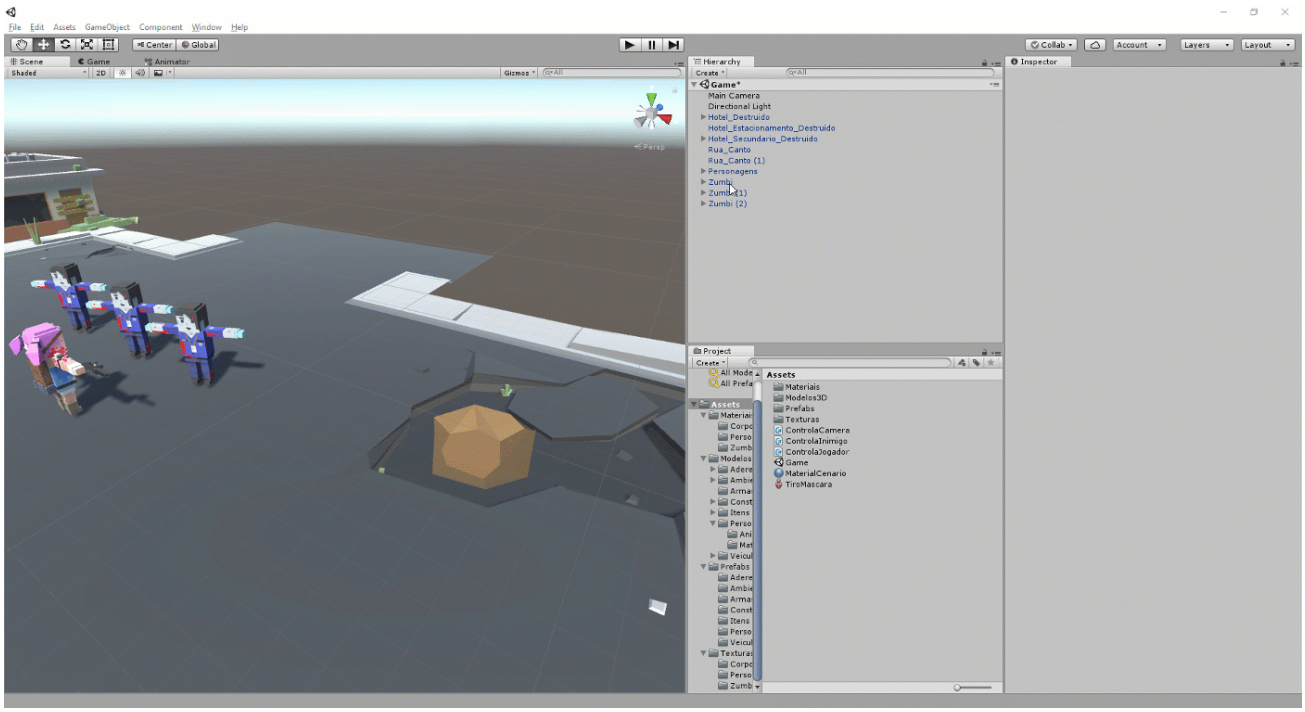
12

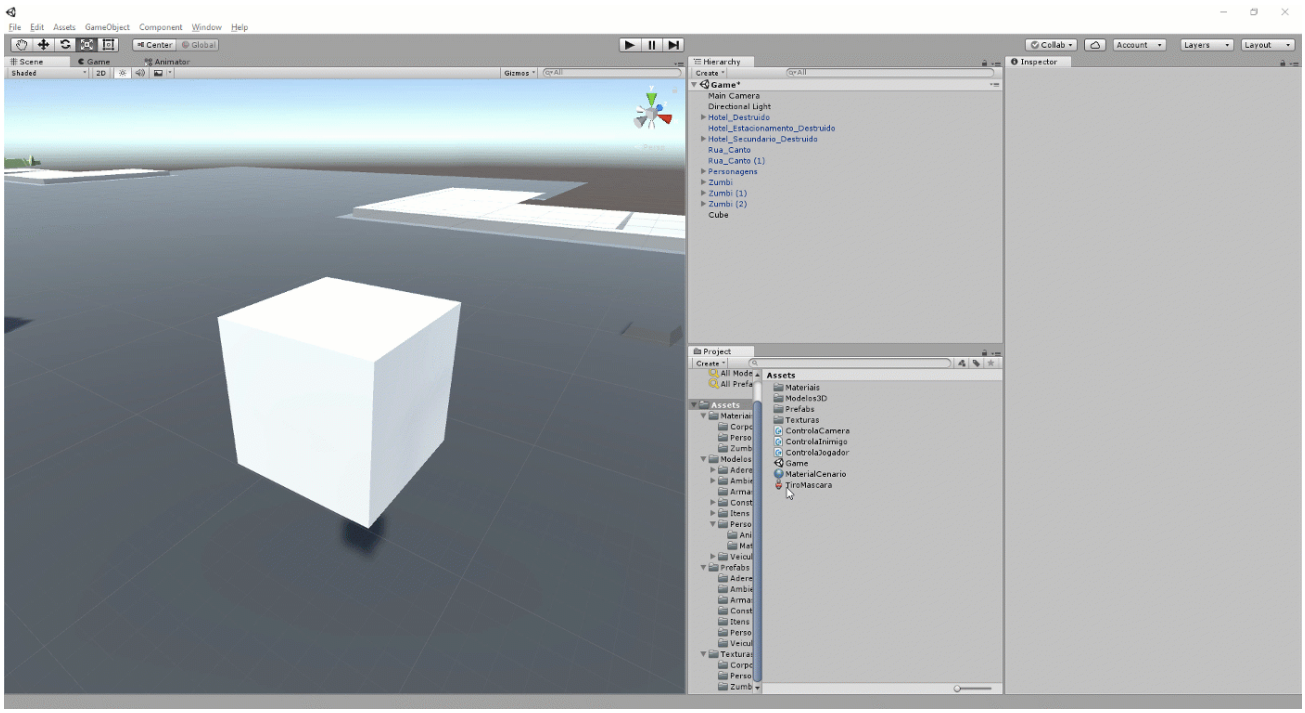
Personagem atirar

Agora temos a arma na mão do personagem vamos fazer ele atirar?

Para isso temos que ter a bala então vamos criar um cubo na *Hierarquia* indo em **Create > 3D Object > Cube**, já que nosso jogo é meio feito em cubos nada melhor para ser a bala do nosso personagem. Vamos diminuir esse cubo utilizando a ferramenta de escala que tem como atalho a tecla **R** clique no quadrado do meio da ferramenta para diminuir todos os lados por igual.



Para essa bala não ficar branca desta forma vamos criar um material para ela, na janela de *Projeto* vamos em **Create > Material** e vamos criar um material com o nome `BalaMaterial` mudando a cor dele para **Amarelo** e jogando ele na nossa bala.



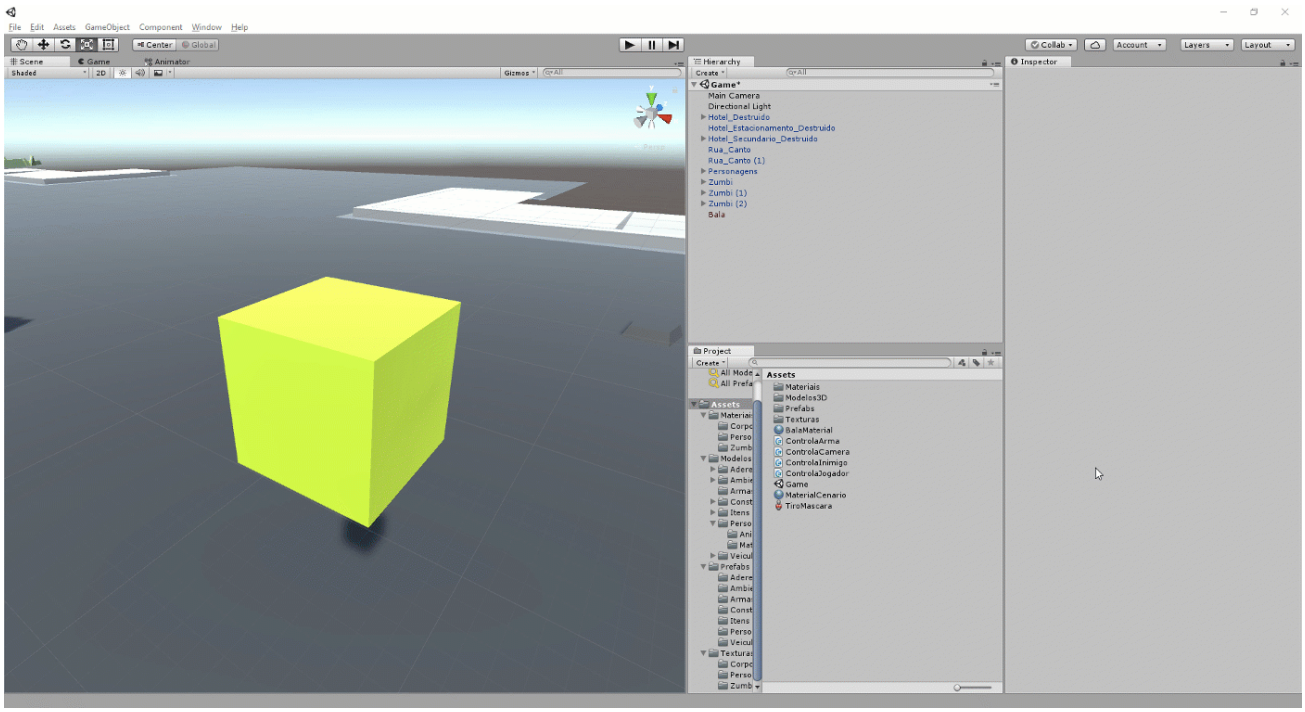
Vamos nomear a cubo para `Bala` e arrastar ele para a pasta **Prefabs** para salvar ela de uma forma já configurada.

Iniciaremos o trabalho no código de fazer o jogador atirar este código poderia até ficar no *Script* `ControlaJogador` mas isso tornaria nosso *Script* grande sem necessidade e fazendo muitas coisas diferentes, vamos então criar um novo código chamado `ControlaArma` e jogar no nosso personagem.

Para o jogador atirar vamos fazer isto quando o clique do mouse for apertado vamos voltar a utilizar a de `Input` que o Unity nos fornece utilizando o método `GetButtonDown` que pega um botão já pré definido pelo nome nos menus da Unity em **Edit > Project Settings > Input** usaremos o botão `Fire1` que simboliza o click do mouse, lembre-se que você tem que usar o `if` para verificar se o botão foi apertado e que isso deve ser testado o tempo todo logo utilizamos o método `Update`, então nosso código fica:

```
void Update ()
{
    if(Input.GetButtonDown("Fire1"))
    {

    }
}
```

Temos que então criar a **Bala** mas não temos uma referência dela no código, vamos então criar uma variável que receberia esse objeto? Crie a variável `public GameObject Bala` no topo do código e preencha com o *Prefab* da **Bala**, lembre-se que a bala não precisa estar no Jogo ela vai ser criada quando for necessária então podemos deletar ela da *Hierarquia* e podemos utilizar ela direto da janela de *Projetos* para preencher a variável.



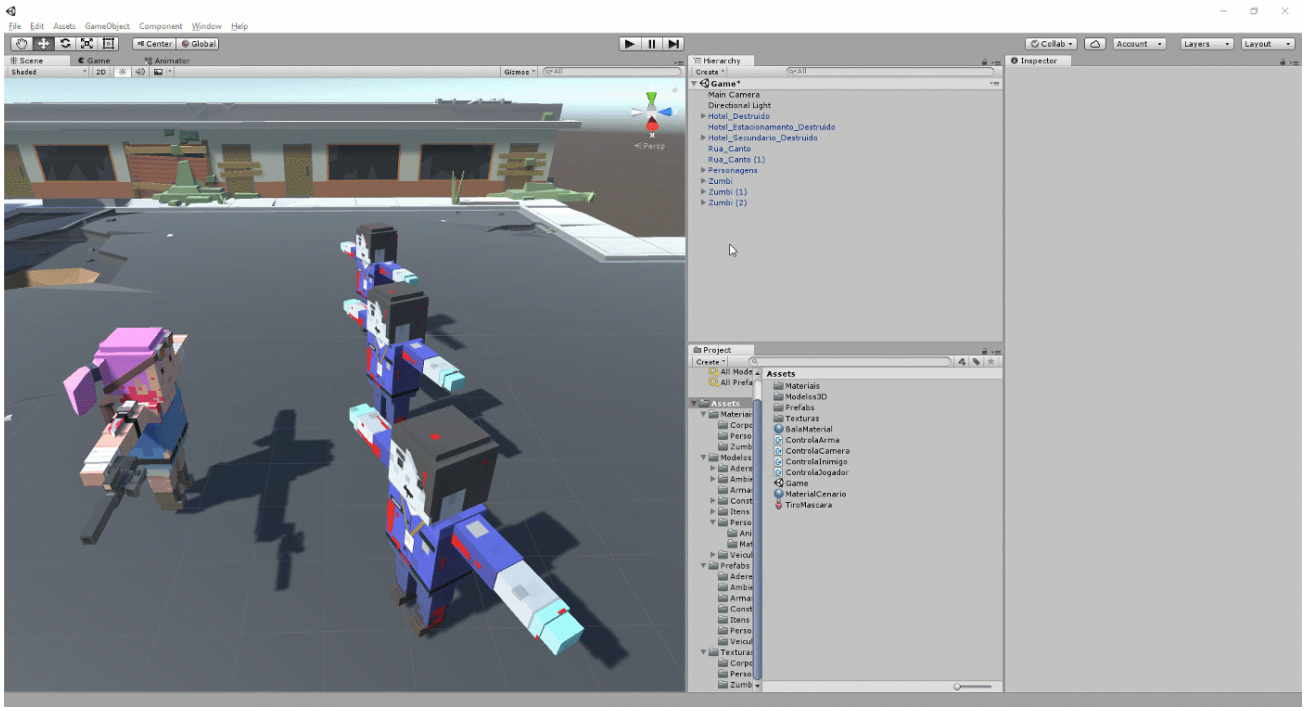
Já temos nossa bala no código e agora temos que criar ela, fazemos isso através do método `Instantiate` que *Instancia*, cria, gera objetos aqui no Unity. Nele temos que passar o objeto que queremos criar, posição e rotação que queremos criar separados por vírgula para posição e rotação vamos utilizar a do personagem?

Então nosso código fica da seguinte forma.

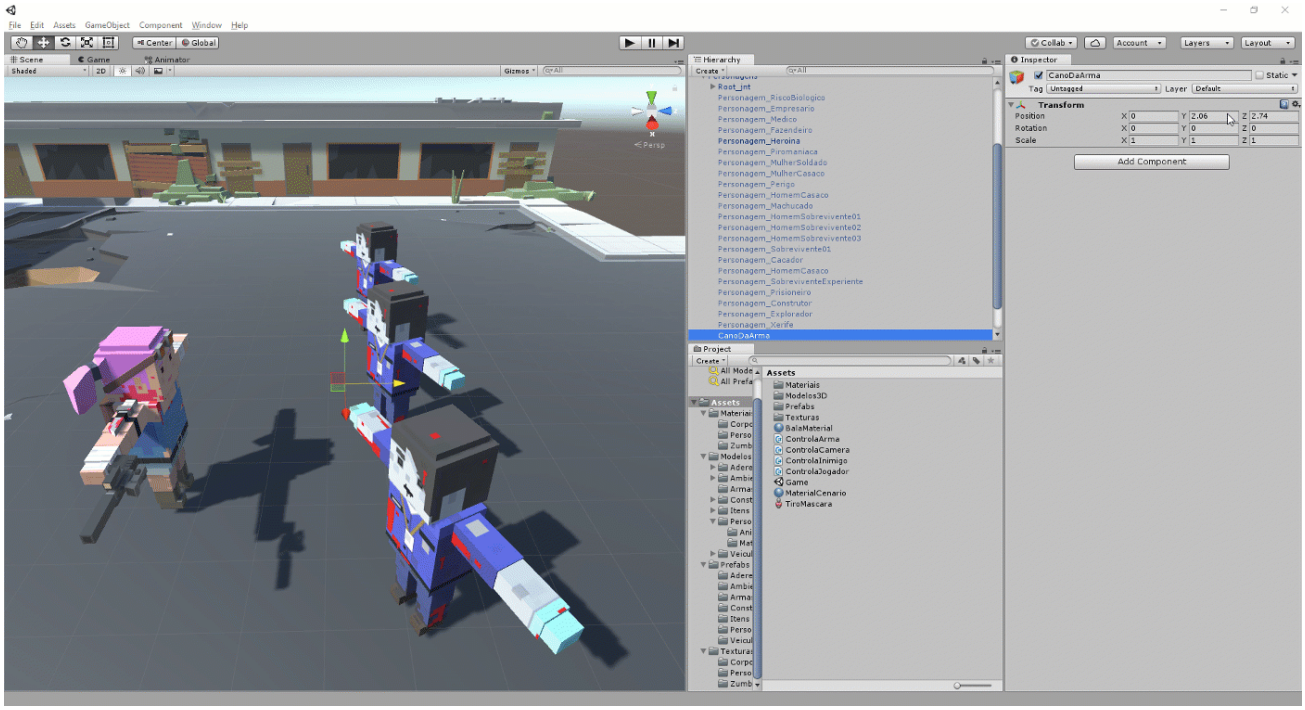
```
void Update ()
{
    if(Input.GetButtonDown("Fire1"))
    {
        Instantiate(Bala, transform.position, transform.rotation);
    }
}
```

Agora nossa **Bala** está sendo criada mas temos dois problemas, ela não sai da arma e nem se move para frente.

Vamos começar fazendo ela sair da arma para isso vamos criar um objeto que marca a posição do cano da arma de onde sai o tiro. Vamos na *Hierarquia* e criamos um objeto em **Create > Create Empty** isso gera um objeto vazio que só tem uma posição no espaço, nomeie esse objeto como `CanoDaArma` e arraste ele para dentro do personagem para ele seguir o jogador.



Agora zere todas a posição em X, Y e Z e posicione no cano da arma este objeto dando **Pause e Play** para ter uma ideia da posição quando posicionar copie os valores do *Componente* de **Transform** clicando na engrenagem e selecionando **Copy Component** e cole novamente quando retirar o **Play** selecionando **Paste Component**.



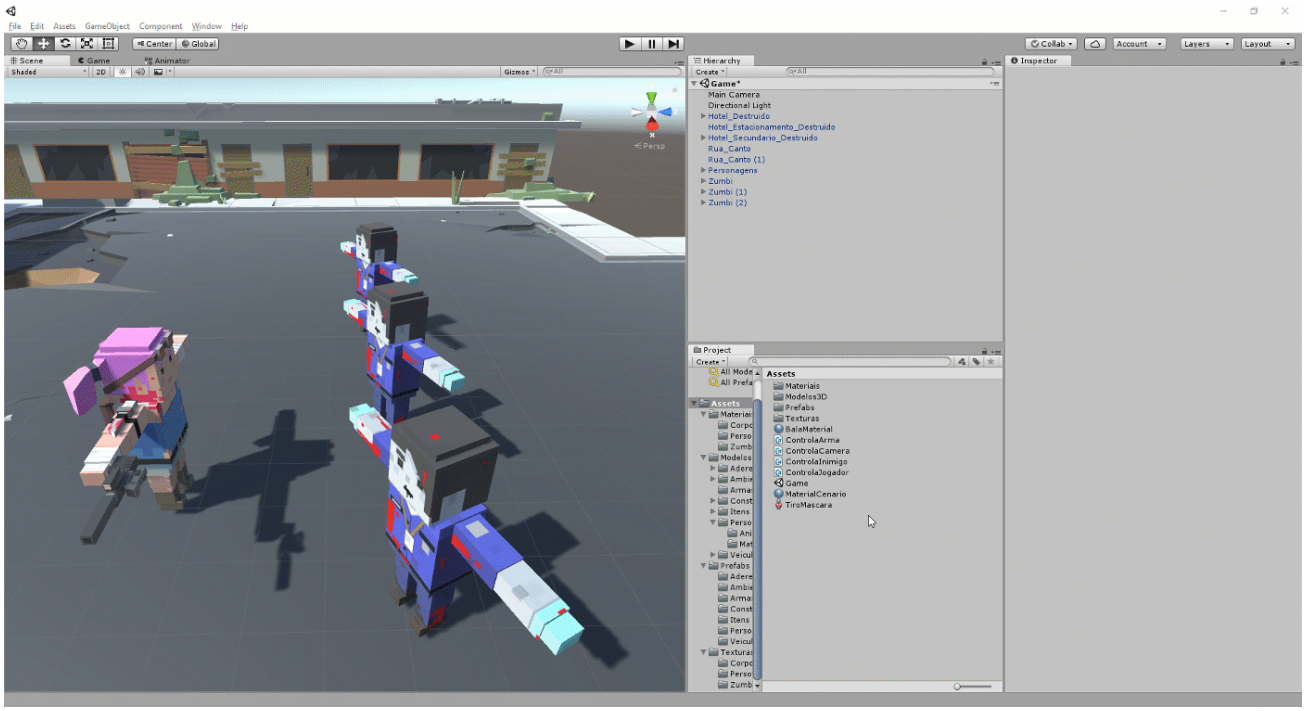
Temos então uma posição marcada como o lugar que a bala sai através desse objeto. Vamos referenciar ele no código criando uma variável `public GameObject CanoDaArma` no topo do código abaixo da variável `Bala` e vamos preenchê-la com o objeto criado no *Inspetor*.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class ControlaArma : MonoBehaviour {

    public GameObject Bala;


    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {
        if(Input.GetButtonDown("Fire1"))
        {
            Instantiate(Bala, transform.position, transform.rotation);
        }
    }
}

```

Agora temos que mudar a linha onde criamos a arma para utilizar a posição deste objeto! Então temos o seguinte:

```
void Update ()
{
    if(Input.GetButtonDown("Fire1"))
    {
        Instantiate(Bala, CanoDaArma.transform.position, CanoDaArma.transform.rotation);
    }
}
```

E para finalizar vamos fazer a bala ir para frente vamos no *Prefab* dela na pasta **Prefab**, adicionar um componente de **Rigidbody** para ela se movimentar lembrando de desmarcar a opção **Use Gravity** pois não queremos que a bala caia com a gravidade e vamos criar um novo código chamado `Bala` e vamos colocar na Bala indo em **Add Component > Scripts > Bala**.



Para fazer a **Bala** ir para frente é algo que já fizemos antes no personagem mas agora vamos fazer utilizando o **Rigidbody** vamos criar uma variável para podemos trocar o valor da Velocidade de forma rápida no *Inspetor* através da linha `public float Velocidade` no topo do *Script*.

Agora vamos criar o método `FixedUpdate` para fazermos ela se deslocar pela frente utilizando o **Rigidbody** com o método `MovePosition`.

```
void FixedUpdate ()
{
    GetComponent<Rigidbody>().MovePosition (GetComponent<Rigidbody>().position + transform.forwa
}
```

Aqui utilizamos `transform.forward` ao invés de `Vector3.forward` porque o `Vector3.forward` não leva em consideração a rotação do objeto já o `transform.forward` joga a **Bala** frente levando sua rotação em consideração.

Você pode fazer download do projeto Unity desta aula [clicando aqui (https://github.com/alura-cursos/unity-shooter/raw/master/Final_Aula3.zip)](https://github.com/alura-cursos/unity-shooter/raw/master/Final_Aula3.zip)