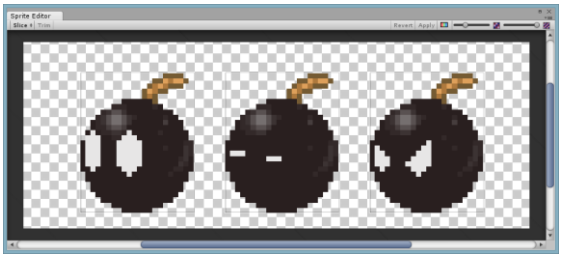
LABORATÓRIO: CRIANDO O JOGO MEGAMAN 2D

Parte-2

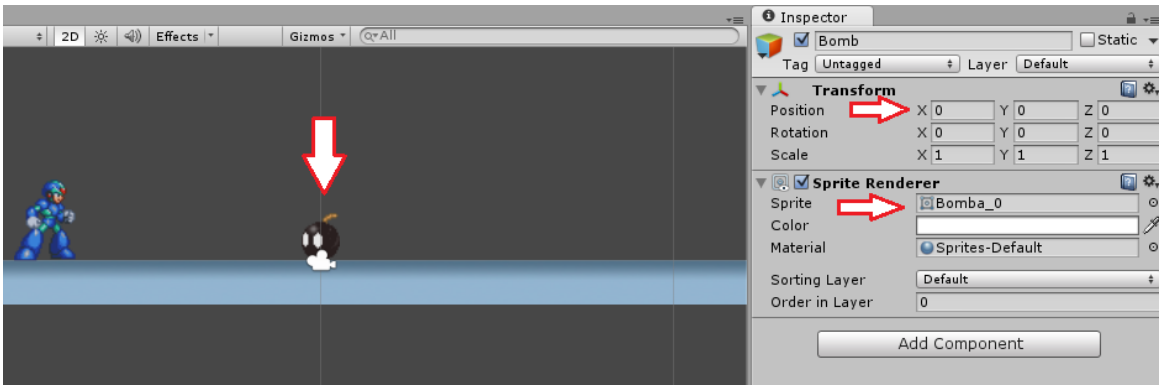
1. Agora iremos inserir no jogo os obstáculos que o nosso herói deverá pular para ganhar pontos. O obstáculo será uma bomba e primeiro vamos importar para o projeto o *sprite* "*Bomb.png*" (disponível no site do professor).

2. Após importar o *sprite, selecione-o e* clique no botão *Sprite Editor* para recortar as imagens que farão parte da animação. Recorte as imagens da mesma forma que fizemos para os outros *sprites* e não se esqueça de definir sua propriedade *Pixel per Unit* para "*20*" e *Sprite Mode* para "*Multiple*".

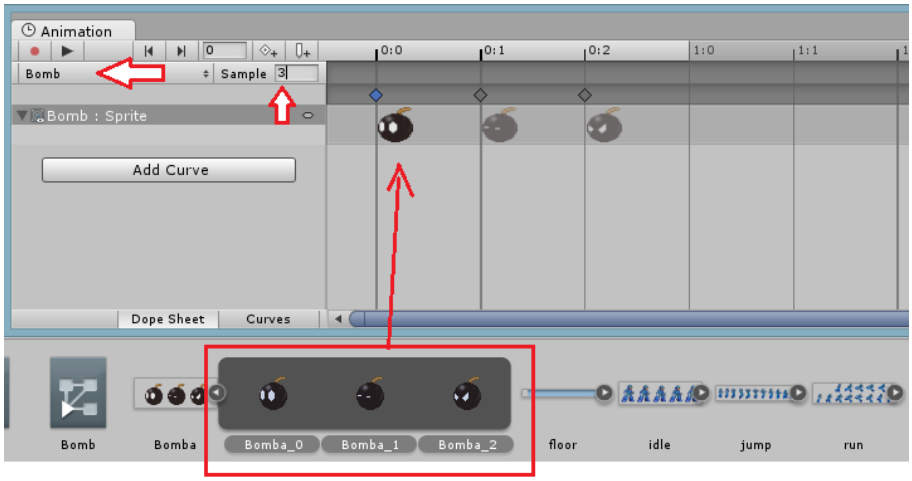


3. Crie um *game object* vazio e chame-o de Bomb. Utilizaremos este objeto como um *template* para definirmos as características das bombas e, depois, iremos criar um *Prefab* a partir dele.

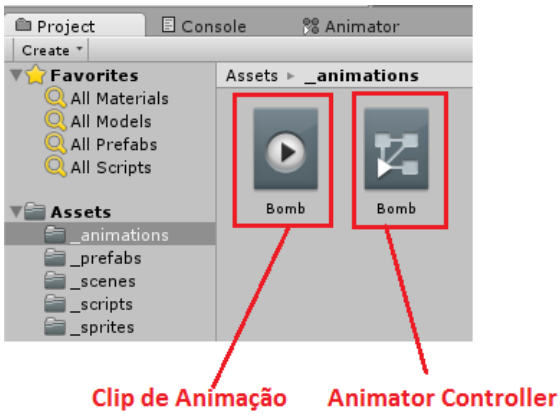
   Adicione um componente *Sprite Renderer* em *Bomb* e insira uma das imagens do *sprite* da bomba na sua propriedade *Sprite*. Faça um *Reset* no seu componente *Transform* para posicioná-lo na origem da cena e alinhá-lo com o piso.



4. Selecione o objeto *Bomb* na janela *Hierarchy* e abra a janela *Animation* (menu *Window-> Animation*). Crie um novo *clip* e salve-o na pasta *_animations* como "*Bomb.anim*". Em seguida, arraste as 3 imagens do *sprite* da bomba para dentro do *clip* e defina sua propriedade *Sample* para "*3*".



5. Acesse a pasta *_animations* na janela *Project* e veja que foram criados dois arquivos com nome "*Bomb*", sendo que um se refere ao clipe de animação que nós acabamos de criar e o outro se refere ao *Animator Controller*, o qual foi criado automaticamente e atribuído como componente ao objeto *Bomb*.

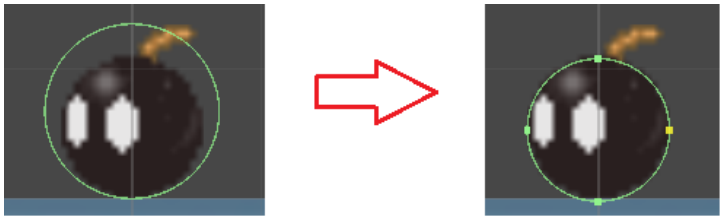


6. Agora vamos definir o comportamento da bomba. Crie um *script*, salve-o como *BombBehaviour* e insira o código a seguir dentro dele. Neste *script* movemos a bomba para direita e, caso ela fique numa posição fora da visão da câmera, destruímos seu objeto.

```
//velocidade da bomba
public var velocidade: float = 4;

function Update () {
    //obtem o valor da posiçao X do objeto Bomb
    var x = gameObject.transform.position.x;

    //se a posiçao X for maior que -15, move-a para a direita
    //fazendo o calculo utilizando o Time.deltaTime
    if (x > -15){
        x += -1 * velocidade * Time.deltaTime;
        gameObject.transform.position =
            new Vector3(x, gameObject.transform.position.y, gameObject.transform.position.z);
    }
    else {
        //destroi o objeto bomba porque ele saiu da visao da camera
        Destroy(gameObject);
    }
}
```

7. Execute e teste o jogo. Perceba que a bomba irá atravessar o herói e isto ocorre porque ainda não definimos um *collider* para a bomba.

8. Selecione o objeto *Bomb* e adicione nele um componente *Circle Collider 2D* (botão *Add Component->Physics 2D-> Circle Collider 2D*). Clique no botão *Edit Collider* e defina sua posição e escala para que se encaixe na circunferência da bomba, conforme figura. Adicione também um componente *Rigidbody 2D* ao objeto *Bomb*.

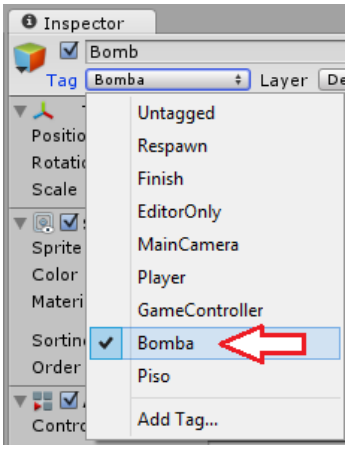


9. Execute o jogo e veja que agora a bomba irá colidir com o herói, arrastando-o para fora da cena.

10. Vamos inserir o *script* para detectar a colisão da bomba com o herói. Abra o *script* *HeroBehaviour* e faça as seguintes alterações na função *OnCollisionEnter2D()*:

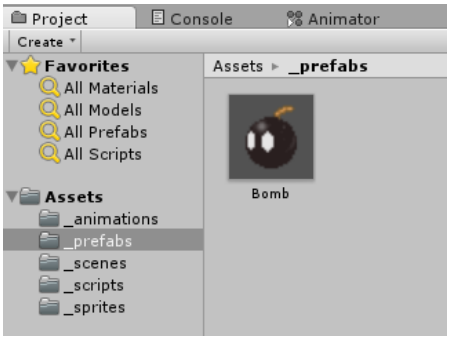
```
//Esta funçao eh executada quando o player toca no piso
function OnCollisionEnter2D(coll: Collision2D) {
    if ( coll.gameObject.CompareTag("Bomba") ){
        Destroy(coll.gameObject);
    }
    else if ( coll.gameObject.CompareTag("Piso") ){
        noPiso = true;
        pulando = false;
    }
}
```

11. Agora selecione o objeto *Bomb*, insira uma nova *Tag* chamada "*Bomba*" e a atribua para o objeto *Bomb*. Essa *Tag* foi utilizada na função *OnCollisionEnter2D()* para verificarmos o tipo do objeto antes de destruí-lo.

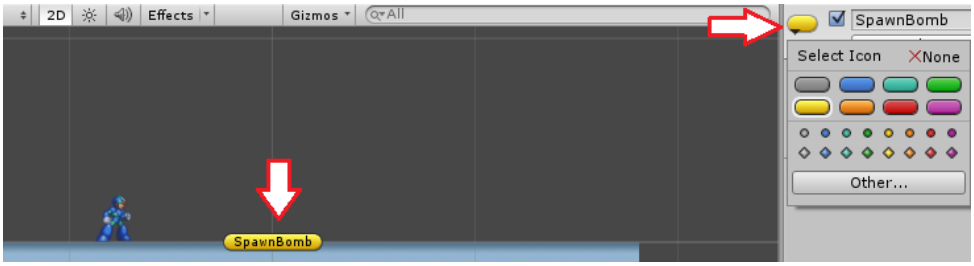


12. Execute o jogo e verifique se a bomba irá sumir quando colidir com o herói.

13. Até aqui já fizemos todas as definições necessárias para os objetos do tipo Bomb do nosso jogo. Agora criemos um *Prefab* (*template*) a partir do objeto Bomb para podermos criá-lo aleatoriamente durante o jogo. Crie uma pasta chamada *_prefabs* e arraste e solte o objeto *Bomb* dentro dela. Em seguida, delete o objeto *Bomb* da cena do jogo.



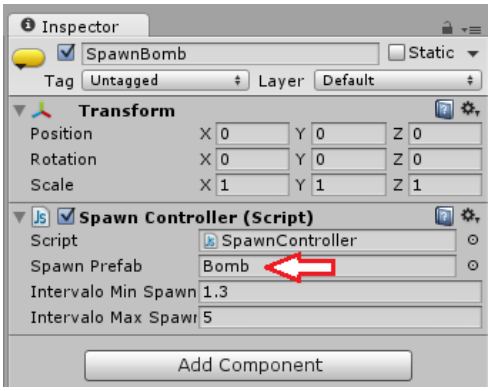
14. Agora vamos criar o objeto que será responsável por gerar (*spawn*) as bombas no nosso jogo. Crie um objeto vazio e chame-o de *SpawnBomb*. Posicione-o na origem da cena e, para podermos visualizá-lo, adicione um ícone amarelo para ele através do botão ao lado do nome do objeto na janela *Inspector*, conforme figura a seguir.



15. Crie um novo script e salve-o como *SpawnBombBehaviour*. Insira nele o código exibido a seguir.

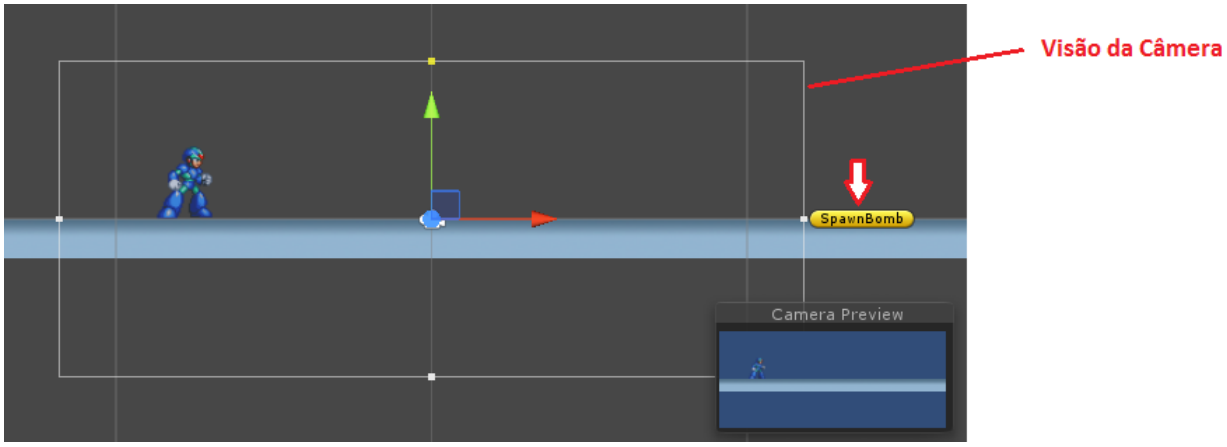
```
public  var spawnPrefab: UnityEngine.GameObject;
public  var intervaloMinSpawn: float = 1.3;
public  var intervaloMaxSpawn: float = 5;
private var tempoDecorrido: float = 0;
private var intervaloSpawn: float;

function Start () {
    //obtem um valor aleatorio dentro do intervalo de spawn minimo e maximo
    intervaloSpawn = Random.Range(intervaloMinSpawn, intervaloMaxSpawn);
}

function Update () {
    //armazena o tempo decorrido
    tempoDecorrido += Time.deltaTime;
    //se o tempo decorrido for >= que o intervalo aleatorio de spawn
    if( tempoDecorrido >= intervaloSpawn) {
        //cria um objeto Bomb e posiciona-o junto com o SpawnBomb
        var obj: GameObject = Instantiate(spawnPrefab);
        obj.transform.position = gameObject.transform.position;
        //zera o tempo decorrido e gera um novo intervalo aleatorio
        tempoDecorrido = 0;
        intervaloSpawn = Random.Range(intervaloMinSpawn, intervaloMaxSpawn);
    }

}
```

16. Agora selecione o objeto *SpawnBomb* na janela *Hierarchy* e insira o script criado dentro dele. Defina a propriedade *Spawn Prefab* do script para o *prefab* da bomba que nós criamos.



17. Execute o jogo e veja que as bombas já estão sendo criadas aleatoriamente na cena.

18. Agora vamos ajustar a posição do objeto *SpawnBomb* para fora da visão da câmera, para que as bombas sejam criadas fora as visão do jogador.



19. Execute e teste o jogo. Perceba que ao pular já podemos evitar que o herói colida com a bomba.

20. Na próxima parte deste tutorial, iremos fazer com que as bombas fiquem paradas quando o herói estiver parado e que só se movam quando o herói estiver correndo. Iremos definir também a pontuação e vidas do herói (HUD), além das interfaces de usuário para criar o menu inicial, a cena do jogo e a tela de *Game Over*.