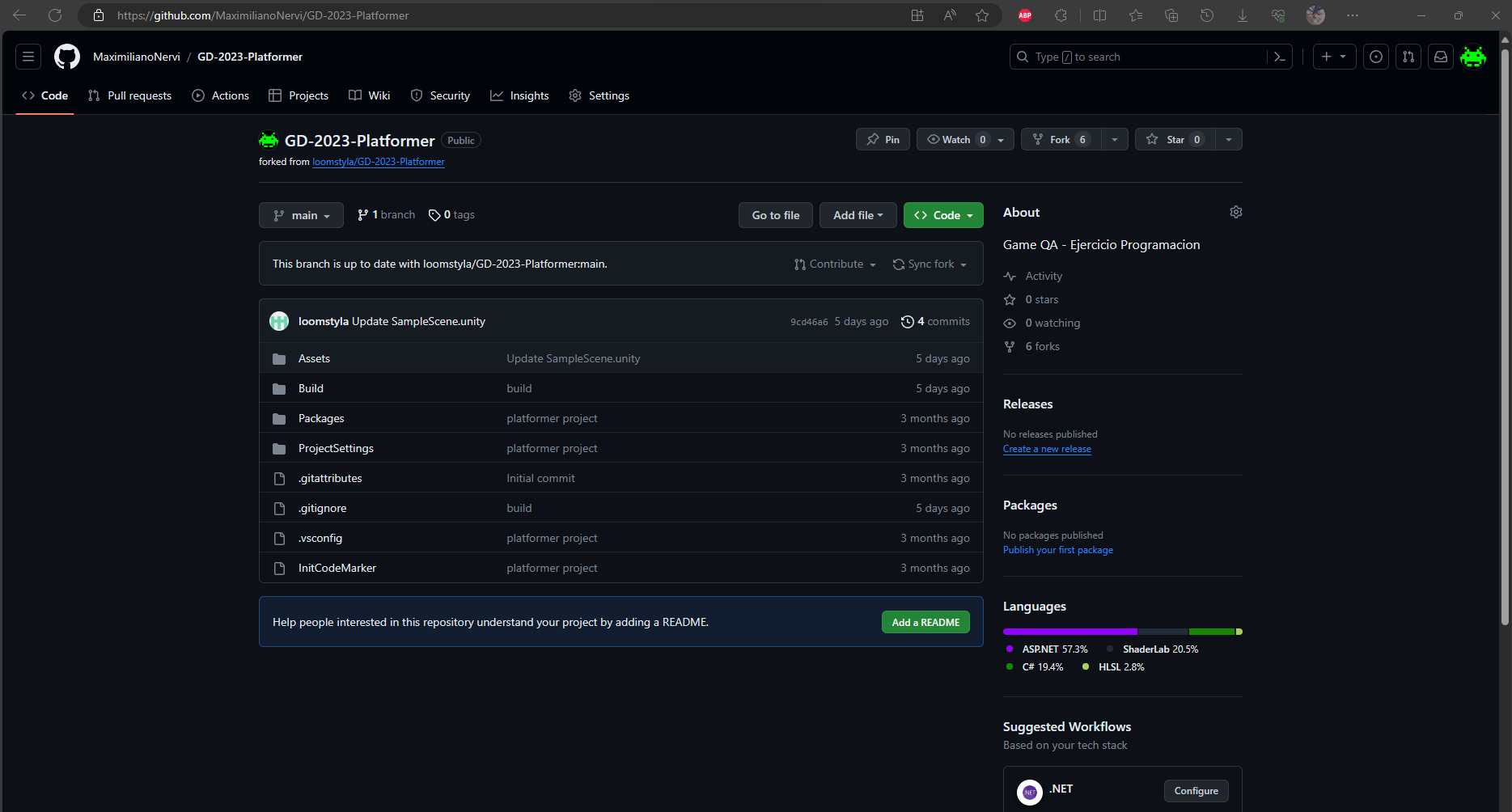
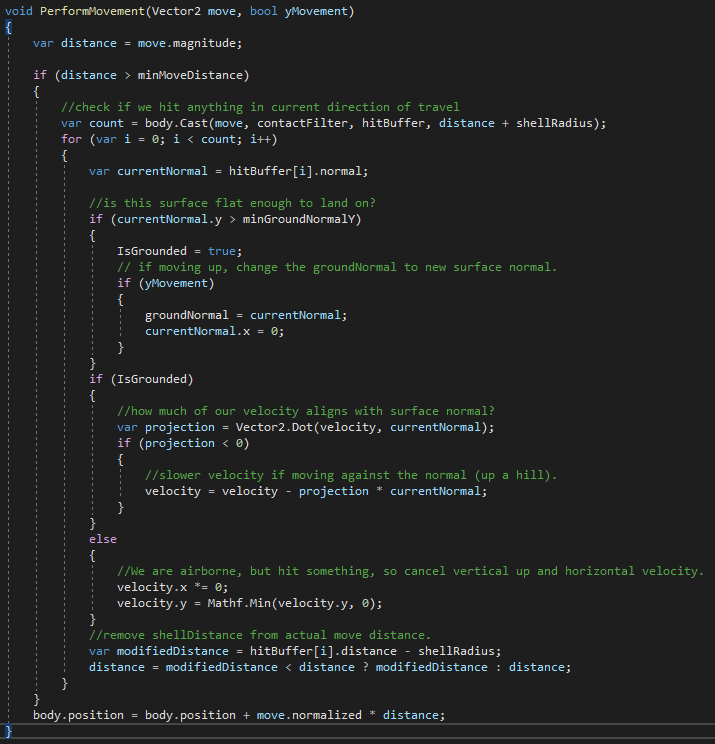
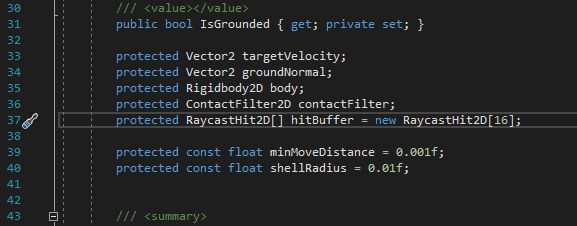
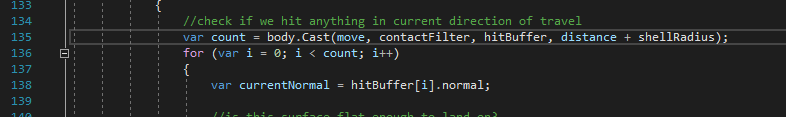
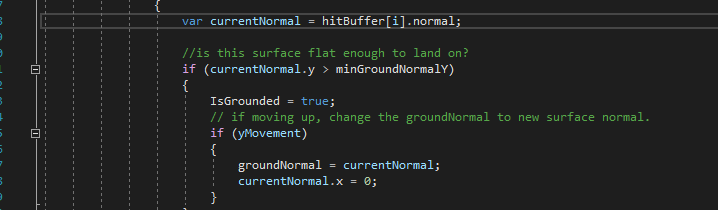
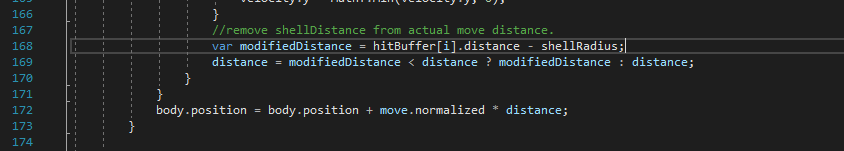
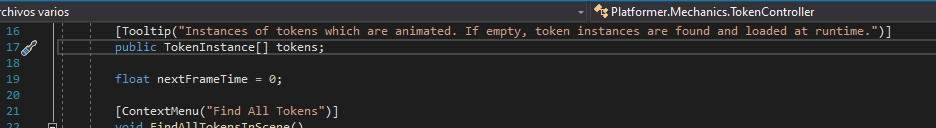
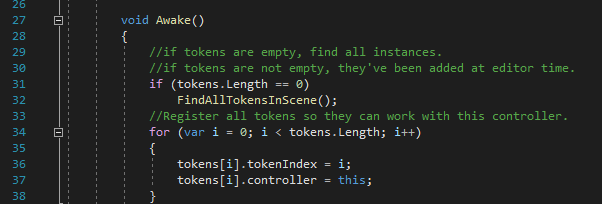
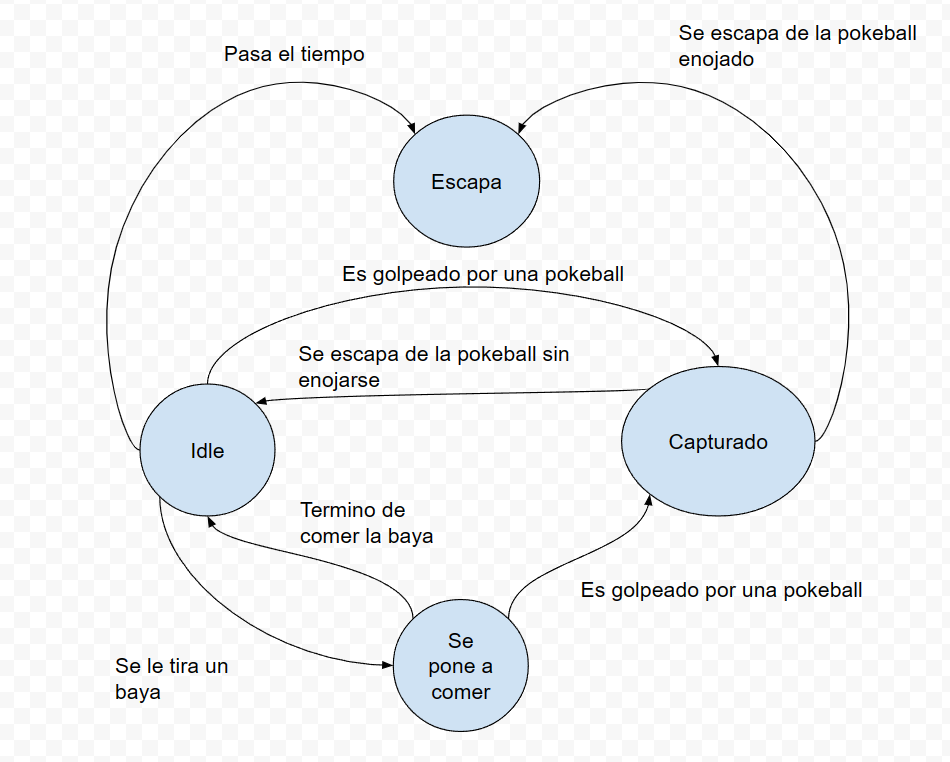
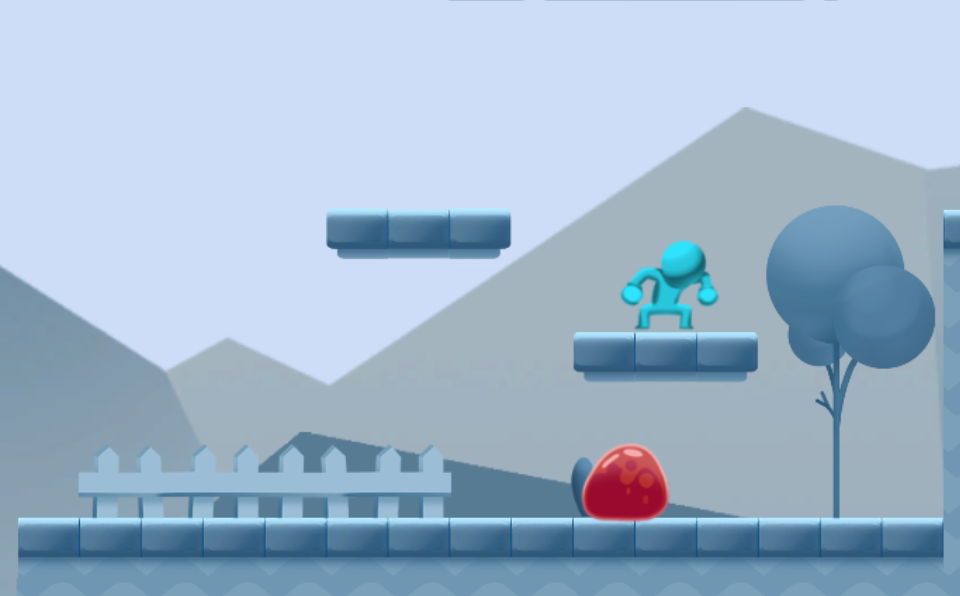
**1.**  


**2a.**  
  
  
  
  
El programa utiliza **raycasts** para analizar la posición del personaje cada vez que llama a **“hitBuffer”** (un array de tipo RaycastHit)   
  
  
En la imagen de arriba se usa para calcular la distancia que hay entre el personaje y un objeto  
  
  
En la imagen de arriba se usa para analizar si la posición del personaje es mayor que la posición mínima del personaje en el eje y  
  
  
En la imagen de arriba se resta el radio del Shell del objeto para terminar calculando la nueva distancia

**2b.**  
  
  
  
Los tokens del juego se encuentran estructurados en un array ya que luego se puede navegar 1 por 1 y eliminar para luego cuando no hay más apagar la animación

**2c.**   
El juego que elegí es Pokemon Go! y el NPC es: cualquier Pokémon (básicamente), el state machine es igual para todos.  
  
   
  
[Link del state machine en mi Google drive](https://docs.google.com/drawings/d/11wTSlDXayKg3bSMX72nyZKMCagT04qDitwyu8dF4eEw/edit?usp=sharing)

**2d.**   
Bugs encontrados:  
  
la primer plataforma, no tiene colisión: 



El primer token que está pasando las 2 primeras plataformas, no tiene colisión:  




El 4to enemigo que se encuentra en el juego, no tiene colisión:

