



Documento Nuclear Juan

Juan Laurino

28/11/2021

ESCOLA PIA GRANOLLERS - DAMVIOD

0. Introducción	3
1. GDD Inicial	3
1.1 Resumen del juego	3
1.2. Resumen del flujo del juego (Final en 2.2)	3
1.3. Juego y mecánica	4
1.3.1. Progresión del juego	4
1.3.2. Movimiento en el juego	5
1.3.3. Objetos	5
1.3.3.1. Cofres	6
1.3.4. Acciones	6
1.3.5. Combate	6
1.4. Áreas y personajes	6
1.4.1. Áreas	6
1.4.3. Personajes	6
1.5. Controles	7
1.6. Enemigos	7
1.7. Extra	8
2. GDD Final	9
2.2. Resumen del flujo del juego	9
2.3. Juego y mecánica	10
2.3.1. Progresión del juego	10
2.3.2. Movimiento en el juego	10
2.3.3. Objetos	10
2.3.3.1. Cofres	12
2.3.4. Acciones	12
2.3.5. Combate	13
2.4. Áreas y personajes	13
2.4.1. Áreas	13
2.4.3. Personajes	13
2.5. Controles	13
2.6. Enemigos	14
2.7. Extra	14
3. Esquemas	15
3.1 Esquema de escenas:	15
3.2 Estados de arquitectura:	15
3.3 Estados personaje principal:	16
3.4 Estados un enemigo:	16
3.5 Estados objeto del juego:	17
4. Explicación	17
5. Fuente de información	19

0. Introducción

Proyecto de fin de curso SDL2 y C++

Elegí el videojuego “Nuclear Throne” porque me parece un juego relativamente sencillo pero muy divertido, lo he jugado mucho y pensé que sería un buen juego para intentar hacer en C++.

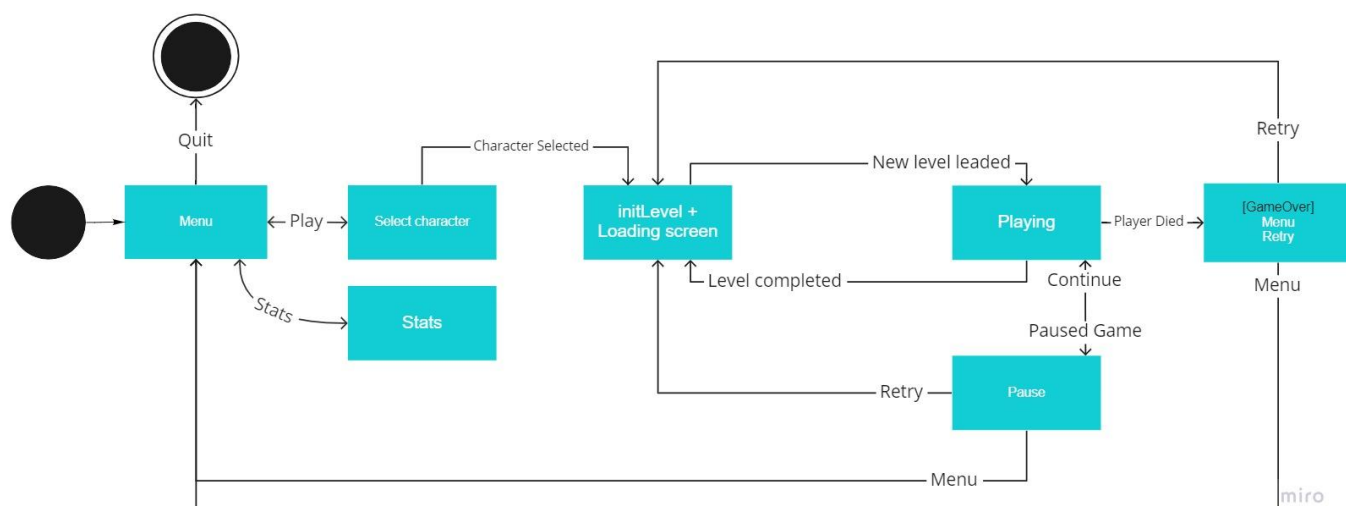
El juego tiene bastantes desafíos como puede ser que la “cámara” se mueva con el movimiento del personaje y el mouse a la vez, la generación aleatoria de niveles o la inteligencia artificial de los enemigos. Espero poder hacer un juego completamente jugable y que además pueda ser bastante rejugable a pesar de tener poco contenido gracias al RNG. Ya que he jugado bastante al juego original sé las mecánicas pero usaré para cosas que no sepa la wiki del juego, los gráficos los tomaré de Spriter resource y los que falten los tomaré de “yal.cc” una página para coger todas las imagenes de los juegos.

1. GDD Inicial

1.1 Resumen del juego

Nuclear Throne es un videojuego de acción post-apocalíptico roguelike donde tienes que destruir a la mayor cantidad de enemigos usando una variedad de armas y pasar la mayor cantidad de niveles, dependiendo de los enemigos que hayas destruido tendrás una cantidad de “Rads” que se guardará como puntaje. Mientras más niveles pasas, más aumenta la cantidad de enemigos y las probabilidades de que salgan enemigos más fuertes.

1.2. Resumen del flujo del juego (Final en 2.2)



En el juego original solo se puede salir del juego desde el menú principal (o dándole a la X de la pantalla)

Cuando se abre el juego por primera vez, se ve una pequeña “animación” del menú en la cual se ven estas imágenes (o similares):



Luego sale el título del juego y se le tiene que dar a alguna tecla para comenzar e ir al menú principal



En el menú se puede ir a Play, Stats y Quit (las otras en principio son de decoración xd)



1.3. Juego y mecánica

1.3.1. Progresión del juego

La progresión es en runs, empiezas la partida y no termina hasta que mueres.

Dentro de una run, el matar enemigos te da Rads lo que te hace subir de nivel y se guarda tu puntuación total al terminar una run, ese es el sistema de rankings y puedes ver tus puntuaciones máximas en "Stats", el juego no tiene guardado de partida.

1.3.2. Movimiento en el juego

El personaje puede moverse a todas direcciones y su rotación sigue al mouse (con su arma también). Si el mouse está del centro del personaje a la derecha, el personaje está mirando a la derecha y si el mouse está del centro del personaje a la izquierda, el personaje mira a la izquierda, esa sería su "rotación" y las armas apuntan a donde esté el mouse en todo momento, si el mouse está desde el centro del personaje para arriba, las armas se pintan abajo del personaje y sino, se pintan arriba. El mouse tiene un sprite propio de una mira de arma y puedes moverlo como quieras con el mouse o un mando.

1.3.3. Objetos

Los objetos recogibles son:

- **Rads pequeños**
Son soltados por los enemigos más débiles y las cápsulas de rads, aumentan tu nivel y puntaje.
- **Rads grandes**
Son soltados por los Jefes, enemigos más poderosos como el Scorpion o Golden Scorpion y las cápsulas de rads.
- **Munición**
Aumentan tu munición cuando las recoges. (hasta un máximo dependiendo de cada personaje)
- **Vida**
Aumentan tu vida cuando las recoges. (hasta un máximo dependiendo de cada personaje)
- **Armas**
Cambian la manera en que enfrentas a tus enemigos, hay armas con distintos tipos de municiones.

Las armas son:

Bullet:

- **Revolver**
El arma inicial, esta dispara balas normales, no es automática (no puede mantenerse el click pulsado) y si le das al click lo más rápido que puedes solo dispara 5 balas por segundo.
- **Assault Rifle**
Arma que puede salir en cofres, dispara balas normales, no es automática y por cada disparo suelta 3 balas. Dispara unas 3 veces por segundo. las balas tienen una mínima desviación.

Pellets:

- **Shotgun**
Arma que puede salir en cofres, dispara Pellets (Perdigones), no es automática y si le das al click lo más rápido posible dispara hasta 2 tandas de perdigones por segundo. Por cada tanda son unos 7 perdigones). Cuando se dispara las balas se dispersan bastante,

1.3.3.1. Cofres

- **Cofre de armas**

Aparece en cada sala. Te puede tocar aleatoriamente un arma y viene con balas, el Assault Rifle viene con 64 balas y la Shotgun viene con 12 Pellets.

- **Cofre de munición**

Aparece en cada sala. Te da munición aleatoria, en este caso de Pellets o Bullets, la cantidad también es aleatoria, da entre 30 y 80 Bullets y entre 6 y 18 Pellets.

- **Cofre de vida**

Puede aparecer hasta uno por nivel (20% de probabilidad), recupera 6 de vida.

- **Cofre de regalo**

Puede aparecer hasta uno por nivel (10% de probabilidad), recupera 4 de vida y te da 64 Bullets y 12 Pellets)

- **Cápsula de Rads**

Aparece en cada sala y te da entre 10 y 16 Rads (puntuación)

- **Cápsula de Rads sorpresa (ver en enemigos)**

Puede aparecer hasta uno por sala pero sustituye a la cápsula normal de rads hay un 25% de probabilidades, no da Rads pero spawna enemigos que si dan Rads.

1.3.4. Acciones

Moverse, ataque normal, habilidad especial, interactuar (coger arma, cofre..).

(Teclas en Controles)

1.3.5. Combate

Tienes que eliminar a todos tus enemigos, algunos te disparan y otros te hacen daño por contacto y algunos ambas. ver en Enemigos.

1.4. Áreas y personajes

1.4.1. Áreas

Las áreas son infinitas y se generan aleatoriamente al comenzar cada nivel, solo hay un tipo de "mundo" y es el desierto, hay bloques donde puedes caminar y bloques donde no, en los bloques donde puedes caminar se spawnan todos los items y enemigos además de cosas de entorno como cactus o plantas. Cada área viene con un cofre de armas, un cofre de munición y una cápsula de rads. Además de eso pueden spawnear hasta un cofre de vida y de regalo y hasta una cápsula de rads sorpresa pero esta sustituye a la cápsula de rads normal.

1.4.3. Personajes

Todos los personajes tienen la misma velocidad y pueden hacer las mismas cosas exceptuando su habilidad especial.



Fish

- Comienza con 8 de vida.
- Habilidad especial de Rodar



Steroids

- Comienza con 10 de vida.
- Habilidad especial de disparar un poco más rápido y hacer a todas las armas automáticas

1.5. Controles

Teclado y mouse:

- Movimiento: W-A-S-D
- Habilidad especial: Espacio/segundo click
- Apuntar: Mouse
- Ataque base: Click izquierdo
- Interactuar: E
- Pausar: TAB/P
- Salir del juego: Escape

Mando:

- Movimiento: Stick izquierdo
- Habilidad especial: A/X
- Apuntar: Stick derecho
- Ataque base: Gatillo derecho
- Interactuar: B/Circulo
- Pausar: Start

1.6. Enemigos



Bandit

Bandidos con un arma básica muy tontos que no te dan un golpe ni aunque quieras que te den, se mueven aleatoriamente y te intentan dar, no te hacen daño por contacto, sueltan 2 Rads cuando les matas. QUITAN 2 de vida y tienen 2 de vida.



Maggot

Gusanos que se mueven aleatoriamente, tienen 2 de vida y quitan 1 de vida, solo hacen daño por contacto. Suelten 1 Rad.



Rad Maggot

Gusanos radioactivos que se mueven aleatoriamente, tienen 2 de vida y quitan 2 de vida, solo hacen daño por contacto. Suelten 3 Rads.



Maggot Nest

Tiene 4 de vida. Cuando lo destruyes salen entre 4 y 8 Maggots. No da rads pero los Maggots que spawna sí. Tiene 2 de daño por contacto.



Big Maggot

Gusano gigante que va hacia tí y cuando muere deja entre 2 y 4 Maggots) hace 2 de daño por contacto. Tiene 4 de vida Suelta 3 Rads.



Scorpion

Escorpión que escupe radiación, tiene 6 de vida y escupe radiación que quita 4 de vida y también quita 4 de vida por contacto. Suelta 6 Rads. // 10 vida



Golden Scorpion

Igual que el Scorpion pero con 8 de vida, 6 de daño y suelta 10 Rads. // 16 vida



Cápsula de Rads

Es como un cofre de rads pero con otro sprite y en vez de soltar rads directamente, spawna entre 6 y 8 Rad Maggots.

1.7. Extra

Los extras no serán añadidos a menos que me sobre tiempo :p

Armas:

Bullet:

- Machinegun
- SMG

Pop:

- Pop Gun
- Pop Rifle

Pellets:

- Double Shotgun
- Auto Shotgun

Bolts:

- Crossbow
- Auto Crossbow
- Heavy Crossbow

Melee:

- Screwdriver
- Chicken Sword
- Wrench
- Shovel

Personajes



Crystal



Y.V



Robot

Boss

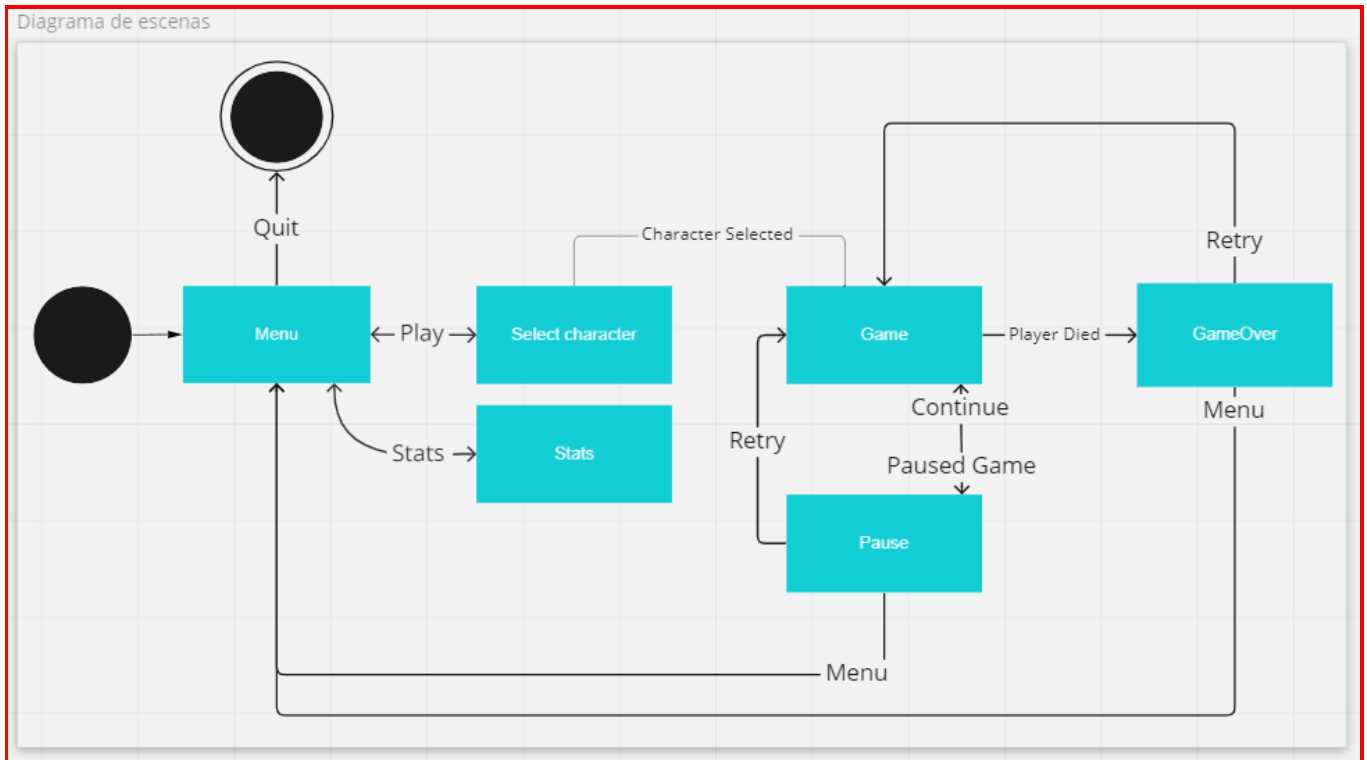


Big Bandit:

El único boss, cuando estás cerca, intenta cargar hacia tí y cuando estás lejos dispara muchas balas a la vez (sin cambiar la dirección de las balas luego de empezar a disparar) cuando termina de disparar esas balas tiene unos 2 segundos que no hace nada. Droppea muchos Rads, vida y munición

2. GDD Final

2.2. Resumen del flujo del juego



El flujo de juego ya no incluye initGame, la regeneración del mapa se hace en el mismo Game

En el juego original solo se puede salir del juego desde el menú principal (o dándole a la X de la pantalla)

Cuando se abre el juego por primera vez, se ve una pequeña “animación” del menú en la cual se ven estas imágenes (o similares):



Luego sale el título del juego y se le tiene que dar a alguna tecla para comenzar e ir al menú principal



En el menú se puede ir a Play, Stats y Quit (las otras en principio son de decoración xd)



2.3. Juego y mecánica

2.3.1. Progresión del juego

La progresión es en runs, empiezas la partida y no termina hasta que mueres.

Dentro de una run, el matar enemigos te da Rads lo que te hace subir de nivel y se guarda tu puntuación total al terminar una run, ese es el sistema de rankings y puedes ver tus puntuaciones máximas en "Stats", el juego no tiene guardado de partida.

2.3.2. Movimiento en el juego

El personaje puede moverse a todas direcciones y su rotación sigue al mouse (con su arma también).

Si el mouse está del centro del personaje a la derecha, el personaje está mirando a la derecha y si el mouse está del centro del personaje a la izquierda, el personaje mira a la izquierda, esa sería su "rotación" y las armas apuntan a donde esté el mouse en todo momento, si el mouse está desde el centro del personaje para arriba, las armas se pintan abajo del personaje y sino, se pintan arriba.

El mouse tiene un sprite propio de una mira de arma y puedes moverlo como quieras con el mouse o un mando. - Solo con el mouse

2.3.3. Objetos

Los objetos recogibles son:

- Rads pequeños

Son soltados por los enemigos más débiles y las cápsulas de rads, aumentan tu nivel y puntaje.

- Rads grandes

Son soltados por los Jefes, enemigos más poderosos como el Scorpion o Golden Scorpion y las cápsulas de rads.

- **Munición**

Aumentan tu munición cuando las recoges. (hasta un máximo dependiendo de cada personaje)

- **Vida**

Aumentan tu vida cuando las recoges. (hasta un máximo dependiendo de cada personaje)

- **Armas**

Cambian la manera en que enfrentas a tus enemigos, hay armas con distintos tipos de municiones.

Las armas son:

Bullet:

- **Revolver**

El arma inicial, esta dispara balas normales, no es automática (no puede mantenerse el click pulsado) y si le das al click lo más rápido que puedes solo dispara 5 balas por segundo.

- **Assault Rifle**

Arma que puede salir en cofres, dispara balas normales, no es automática y por cada disparo suelta 3 balas. Dispara unas 3 veces por segundo. las balas tienen una mínima desviación.

- **Pop gun**

Arma que puede salir en cofres, dispara balas normales y es automática, (se puede mantener el click pulsado y disparará hasta quedarse sin balas) dispara unas 14 balas por segundo. las balas tienen una mínima desviación.

-Arma añadida

Pellets:

- **Shotgun**

Arma que puede salir en cofres, dispara Pellets (Perdigones), no es automática y si le das al click lo más rápido posible dispara hasta 2 tandas de perdigones por segundo. Por cada tanda son unos 7 perdigones). Cuando se dispara las balas se dispersan bastante,





Weapon	Description	Damage per shot	Reload time	Ammo per shot	Spread angle
 Pop Gun		2[3] automatic	0.07 sec	Ammo: 1	4°
 Assault Rifle	Fires a three round burst each click. Starting weapon for all characters	3 x 3 burst	0.37 sec	Ammo: 3	2°
 Revolver		3 single shot	0.20 sec	Ammo: 1	4°
 Shotgun	Fires a random spread of pellet projectiles using one shell per shot.	2[3] x 7 single shot	0.57 sec	Ammo: 1	20°

Imagen añadida para representar mejor las armas

2.3.3.1. Cofres

- **Cofre de armas**
Aparece en cada sala. Te puede tocar aleatoriamente un arma y viene con balas, el Assault Rifle viene con 64 balas y la Shotgun viene con 12 Pellets.
- **Cofre de munición**
Aparece en cada sala. Te da munición aleatoria, en este caso de Pellets o Bullets, la cantidad también es aleatoria, da entre 30 y 80 Bullets y entre 6 y 18 Pellets.
- **Cofre de vida**
Puede aparecer hasta uno por nivel (20% de probabilidad), recupera 6 de vida.
- **Cofre de regalo**
Puede aparecer hasta uno por nivel (10% de probabilidad), recupera 4 de vida y te da 64 Bullets y 12 Pellets)
- **Cápsula de Rads**
Aparece en cada sala y te da entre 10 y 16 Rads (puntuación)
- **Cápsula de Rads sorpresa (ver en enemigos)**
Puede aparecer hasta uno por sala pero sustituye a la cápsula normal de rads hay un 25% de probabilidades, no da Rads pero spawna enemigos que si dan Rads.

2.3.4. Acciones

Moverse, ataque normal, habilidad especial, interactuar (coger arma, cofre..).
(Teclas en Controles)

2.3.5. Combate

Tienes que eliminar a todos tus enemigos, algunos te disparan y otros te hacen daño por contacto y algunos ambas. ver en Enemigos.

2.4. Áreas y personajes

2.4.1. Áreas

Las áreas son infinitas y se generan aleatoriamente al comenzar cada nivel, solo hay un tipo de "mundo" y es el desierto, hay bloques donde puedes caminar y bloques donde no, en los bloques donde puedes caminar se spawnear todos los items y enemigos además de cosas de entorno como cactus o plantas (solo cactus en la versión final). Cada área viene con un cofre de armas, un cofre de munición y una cápsula de rads. Además de eso pueden spawnear hasta un cofre de vida y de regalo y hasta una cápsula de rads sorpresa pero esta sustituye a la cápsula de rads normal.

- Hay 7 niveles generados con Tiled y aparecen aleatoriamente pero no hay generación aleatoria infinita de niveles

2.4.3. Personajes

Todos los personajes tienen la misma velocidad y pueden hacer las mismas cosas exceptuando su habilidad especial.



Fish

- Comienza con 8 de vida.
- Habilidad especial de Rodar



Steroids

- Comienza con 10 de vida.
- Habilidad especial de disparar un poco más rápido y hacer a todas las armas automáticas

2.5. Controles

Teclado y mouse:

- Movimiento: W-A-S-D
- Habilidad especial: Espacio/segundo click
- Apuntar: Mouse
- Ataque base: Click izquierdo
- Interactuar: E
- Pausar: TAB/P
- Salir del juego: Escape

Mando:

- Movimiento: Stick izquierdo
- Habilidad especial: A/X
- Apuntar: Stick derecho
- Ataque base: Gatillo derecho
- Interactuar: B/Circulo
- Pausar: Start

- Para hacer que el juego sea compatible con mando tenía que cambiar como funciona el mouse y/o como se cambia de escenas y no me dió tiempo. La versión original no tiene posibilidad de jugar con mando pero sí hay modificaciones que lo permiten así que sería 100% posible pero no me dió tiempo.

2.6. Enemigos



Bandit

Bandidos con un arma básica muy tontos que no te dan un golpe ni aunque quieras que te den, se mueven aleatoriamente y te intentan dar, no te hacen daño por contacto, sueltan 2 Rads cuando les matas. QUITAN 2 de vida y tienen 2 de vida.



Maggot

Gusanos que se mueven aleatoriamente, tienen 2 de vida y quitan 1 de vida, solo hacen daño por contacto. Sueltan 1 Rad.



Rad Maggot

Gusanos radioactivos que se mueven aleatoriamente, tienen 2 de vida y quitan 2 de vida, solo hacen daño por contacto. Sueltan 3 Rads.



Maggot Nest

Tiene 4 de vida. Cuando lo destruyes salen entre 4 y 8 Maggots. No da rads pero los Maggots que spawnea sí. Tiene 2 de daño por contacto.



Big Maggot

Gusano gigante que va hacia tí y cuando muere deja entre 2 y 4 Maggots) hace 2 de daño por contacto. Tiene 4 de vida Suelta 3 Rads.



Scorpion

Escorpión que escupe radiación, tiene 10* de vida y escupe radiación que quita 4 de vida y también quita 4 de vida por contacto. Suelta 6 Rads.



Golden Scorpion

Igual que el Scorpion pero con 16* de vida, 6 de daño y suelta 10 Rads.



Cápsula de Rads

Es como un cofre de rads pero con otro sprite y en vez de soltar rads directamente, spawnea entre 6 y 8 Rad Maggots.

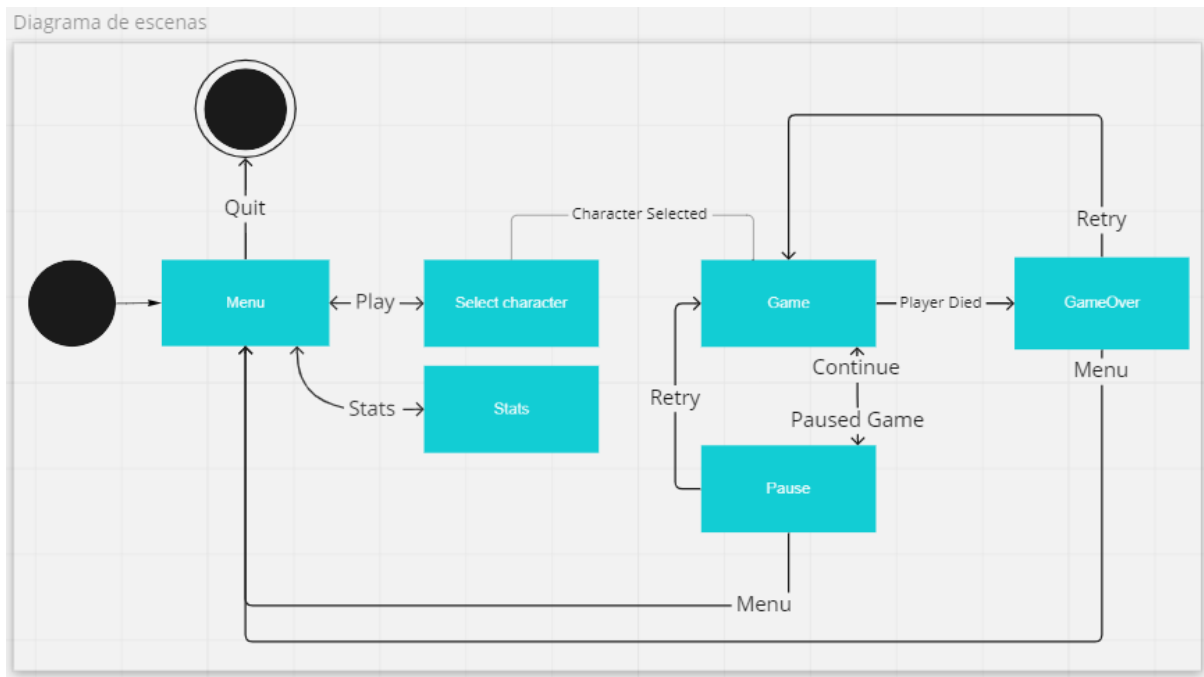
*Le subí la vida porque me pareció muy poco para los enemigos más fuertes que tengo hasta ahora.

2.7. Extra

Ya que no me dió el tiempo ignoré completamente los extras

3. Esquemas

3.1 Esquema de escenas:



3.2 Estados de arquitectura:

QGISElement
QGISElement

- Fields
 - test
 - joinedID
- Methods
 - getQGISelement
 - getCollection
 - getDistance
 - getGeometryID
 - getID
 - getType
 - getW
 - getX
 - getY
 - getZElement
 - init
 - moveFile
 - rename
 - setGeometryID
 - setID
 - setType
 - setW
 - update
- Named Types

Symbol	Meaning
LA	Left Arrow
LS	Left Square
LD	Left Double
LW	Left Word
LSpace	Left Space
LE	Left Edge
LSide	Left Side
LSideW	Left Side Width
LP	Left Parenthesis

```
Kerneldump
Dump:
MEMID
SELECT_CHARACTER
STATS
INIT_LOAD
GMMT_OPR
GMMT
PAUSE
_LASTSCENE
```

- Scene
 - init
 - Methods
 - ~Scene
 - init
 - updateFrame
 - split
 - render
 - reset
 - Scene
 - setFrame
 - update

[illegible][illegible]

```
public
  ThemeLogo
  class
  * important
  # fields
  # contact
  # name
  # methods
  # ThemeLogo
  # init
  # render
  # ThemeLogo
  # update
```

- public
 - Mouse
 - class
 - is abstract
 - fields
 - instance
 - instance
 - methods
 - mouse
 - getinstance
 - set
 - Mouse
 - sender
 - update

```
public
Camera
class
+ constructor
+ Fields
  + _glrenderer
  + _glMap
  + _glinstance
+ Methods
  + ~Camera
  + Camera
  + getinstance
  + set
```

- public
- PickableObject
- class
- ↳ Inheritance
- ↳ Fields
 - ↳ `parent`
 - ↳ `childNodes`
 - ↳ `name`
 - ↳ `objName`
 - ↳ `isFrame`
 - ↳ `rotation`
 - ↳ `type`
 - ↳ `path`
- ↳ Methods
 - ↳ `~PickableObject`
 - ↳ `getType`
 - ↳ `init`
 - ↳ `moveTo`
 - ↳ `PickableObject`
 - ↳ `render`
 - ↳ `update`
- ↳ Nested Types

public class Background {
 // ...
 // Fields
 private int _id;
 private string _name;
 // Methods
 public void Background() {
 // ...
 }
 public void Update() {
 // ...
 }
}

[illegible]

```
public class Lingo {  
    // ...  
    public static void main(String[] args) {  
        // ...  
    }  
}
```

Bullet

Class
@Serializable

Fields

- `_angle`
- `_bulletType`
- `_collided`
- `_collisionCounter`
- `_damage`
- `_direction`
- `_selfFrame`
- `_speed`

Methods

- `-Bullet`
- `Bullet`
- `getCollisionCounter`
- `getDamage`
- `getDirection`
- `getSpeed`
- `hit`
- `isCollidedFromPlayer`
- `render`
- `setCollision`
- `update`

Serialized Types

Audio
Cass

• Faith
• **gondance**

• Materials
• **audio**
• **audio**
• **class**
• **class**
• **gondance**
• **gondance**
• **playing**
• **practicing**
• **playing**
• **practicing**
• **practicing**
• **practicing**

Discs
Cass

• Faith
• **Start/Stop**
• **Start/Stop**

- **Fields**
 - `startTicks`
 - `start`
 - `ticks`
- **Methods**
 - `tick`
 - `getTicks`
 - `start`
 - `stop`
 - `times`

```
InputManager
Class
+ Fields
    - _defaultInput
    - _device
    - _gamepad
    - _joystick
+ Methods
    - InputManager()
    - anyKeyPressed()
    - getDevice()
    - getJoyPressed()
    - getJoystick()
    - InputManager()
    - IsClickPressed()
    - setClickAtThe()
    - update()
```

- Fields
 - g/frameSize
 - g/frameRate
 - h
 - gl/frameSize
 - isHit
 - m
- Methods
 - Video
 - closeGLES
 - close
 - closeGL
 - getFrameIn
 - getFrameOut
 - renderGLES
 - renderGL
 - renderGLS
 - renderGLSL
 - testClose

...

```
public
ScoreGameOver
{
    Date
    # Score

    # Fields
    .._gameOver
    .._ODuration
    .._goCardinal

    # Methods
    .. = ScoreGame
    .. init
    .. split
}
```

Scored
Date
Items

Scenario

Ello

W Summ

a Field

b Mole

[illegible]

```
public class ScorePause {
    // Fields
    private int _score;
    private int _position;
    private int _countdown;

    // Methods
    public void Start() {
        // ...
    }
}
```

```
public
SceneObject3D
{
    // ...
    // Fields
    // ...
    // Methods
    // ...
}
```

Case	Se
1	0.85
2	0.85
3	0.85
4	0.85
5	0.85
6	0.85
7	0.85
8	0.85
9	0.85
10	0.85
11	0.85
12	0.85
13	0.85
14	0.85
15	0.85
16	0.85
17	0.85
18	0.85
19	0.85
20	0.85
21	0.85
22	0.85
23	0.85
24	0.85
25	0.85
26	0.85
27	0.85
28	0.85
29	0.85
30	0.85
31	0.85
32	0.85
33	0.85
34	0.85
35	0.85
36	0.85
37	0.85
38	0.85
39	0.85
40	0.85
41	0.85
42	0.85
43	0.85
44	0.85
45	0.85
46	0.85
47	0.85
48	0.85
49	0.85
50	0.85
51	0.85
52	0.85
53	0.85
54	0.85
55	0.85
56	0.85
57	0.85
58	0.85
59	0.85
60	0.85
61	0.85
62	0.85
63	0.85
64	0.85
65	0.85
66	0.85
67	0.85
68	0.85
69	0.85
70	0.85
71	0.85
72	0.85
73	0.85
74	0.85
75	0.85
76	0.85
77	0.85
78	0.85
79	0.85
80	0.85
81	0.85
82	0.85
83	0.85
84	0.85
85	0.85
86	0.85
87	0.85
88	0.85
89	0.85
90	0.85
91	0.85
92	0.85
93	0.85
94	0.85
95	0.85
96	0.85
97	0.85
98	0.85
99	0.85
100	0.85

```

public
class State {
    // ...
    public
    void
    ...
}

```

[illegible]

```
public
Weapon
{
    # Fields
    private string _nameType;
    private int _id;
    private string _damage;
    private int _rotation;
    private float _approachAngle;
    private float _chase;
    private WeaponType _weaponType;

    # Methods
    public Weapon()
    {
        GetId();
        GetType();
        GetWeaponName();
        GetWeaponDamage();
        GetWeaponRotation();
        GetWeaponApproachAngle();
        GetWeaponChase();
        GetWeaponType();
    }
    public void GetId()
    {
        GetWeaponName();
        GetWeaponDamage();
        GetWeaponRotation();
        GetWeaponApproachAngle();
        GetWeaponChase();
        GetWeaponType();
    }
    public void GetWeaponName()
    {
        GetWeaponDamage();
        GetWeaponRotation();
        GetWeaponApproachAngle();
        GetWeaponChase();
        GetWeaponType();
    }
    public void GetWeaponDamage()
    {
        GetWeaponRotation();
        GetWeaponApproachAngle();
        GetWeaponChase();
        GetWeaponType();
    }
    public void GetWeaponRotation()
    {
        GetWeaponApproachAngle();
        GetWeaponChase();
        GetWeaponType();
    }
    public void GetWeaponApproachAngle()
    {
        GetWeaponChase();
        GetWeaponType();
    }
    public void GetWeaponChase()
    {
        GetWeaponType();
    }
    public void GetWeaponType()
    {
    }
}
```

public
Cache
class
100% Documentation

Fields

- boolean
- float
- int
- int

Methods

- Cache
- Cache
- getCache
- set
- setValueCache
- unset

Nested Types

```
public
  Counter
  class
  {
  public
  {
    # Fields
    private
    {
      count: int;
      name: string;
      type: string;
    }
    # Methods
    public
    {
      ~Counter()
      {
        count = 0;
      }
      get: int
      {
        return count;
      }
      set: int
      {
        count = value;
      }
      increment()
      {
        count++;
      }
      reset()
      {
        count = 0;
      }
    }
  }
}
```

[illegible][illegible]

```
public
class
{
    // Fields
    private
    {
        _annoType
        _chartStyle
        _color
        _font
        _type
        _year
        _regions
    }
    // Methods
    public
    {
        ~Chart()
        Chart()
        GetAnnoType()
        GetType()
        Init()
        IsOpen()
        Open()
        Render()
        SetHttpResponse()
        Update()
    }
    // Nested Types
}
```

- public
 - HighFogget
 - init
 - springboot
- Private
 - promotion
 - postaddoArticle
 - articles
 - name
 - player
 - state
 - newDistance
- Methods
 - HighFogget
 - HighFogget
 - getRate
 - init
 - receiveDamage...
 - render
 - setNameOfPon...
 - setPlayerFor...
 - update
- Virtual Types

```
public
  Scorpion() {}
  Scorpion(int)
  Scorpion(int)

  Fields
  * _foundation
  * _contactorName
  * _tip
  * _name
  * _player
  * _shootOff
  * _spreadAngle
  * _rate
  * _newChance

  Methods
  * Scorpion
  * getRate
  * set
  * noEvilDamage
  * render
  * Scorpion
  * setPlayerOffCenter
  * shoot
  * update

  Nested Types
```

public

File

Class
↳ Overview

Fields

- File
- JanDev
- JanDev
- JanDev
- JanDev

Methods

- File
- File
- int
- recursiveDelete
- render
- update

public

Stream

Class
↳ Invariant

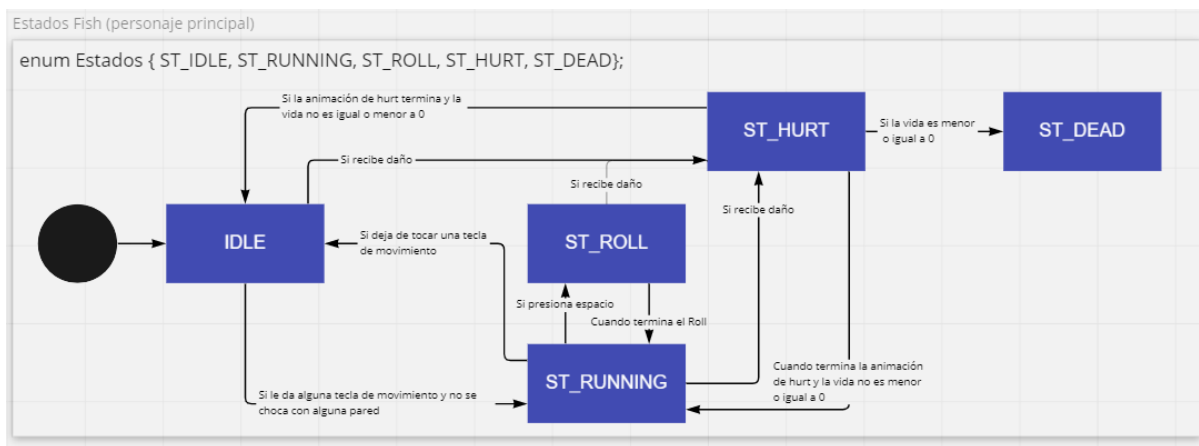
Fields

- `_zip`
- `_lastDir`
- `_lastDir`
- `_path`
- `_rotation`

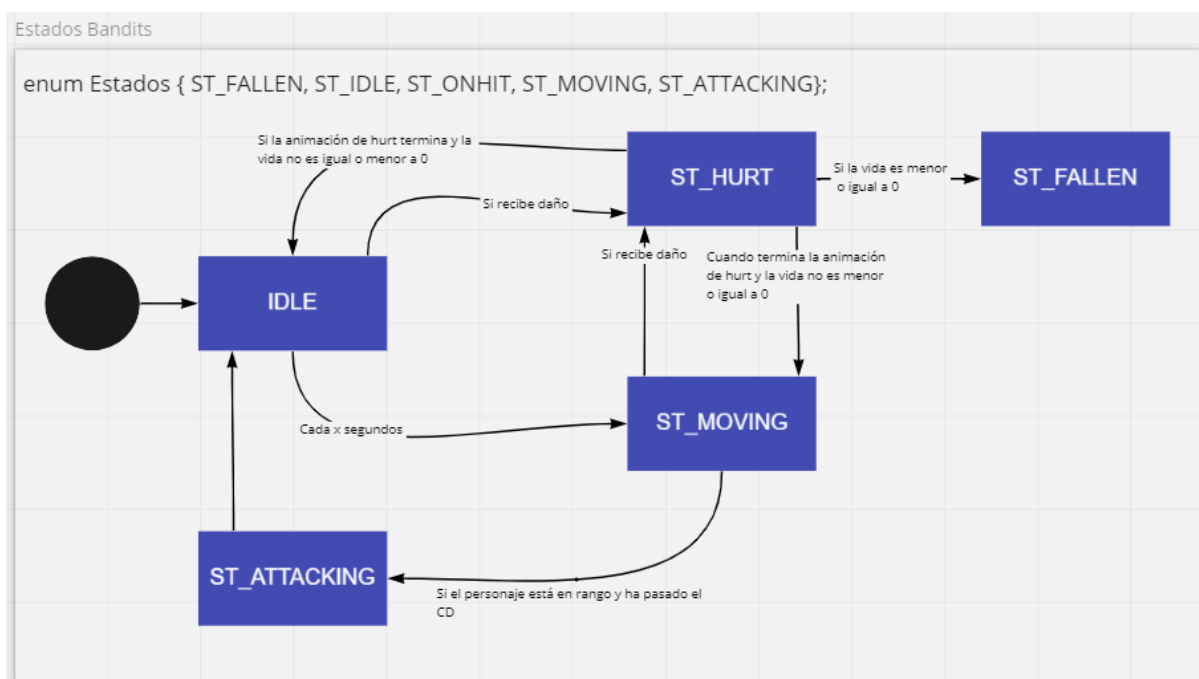
Methods

- `<>Stream`
- `init`
- `nextEndOfRange`
- `number`
- `streams`
- `update`

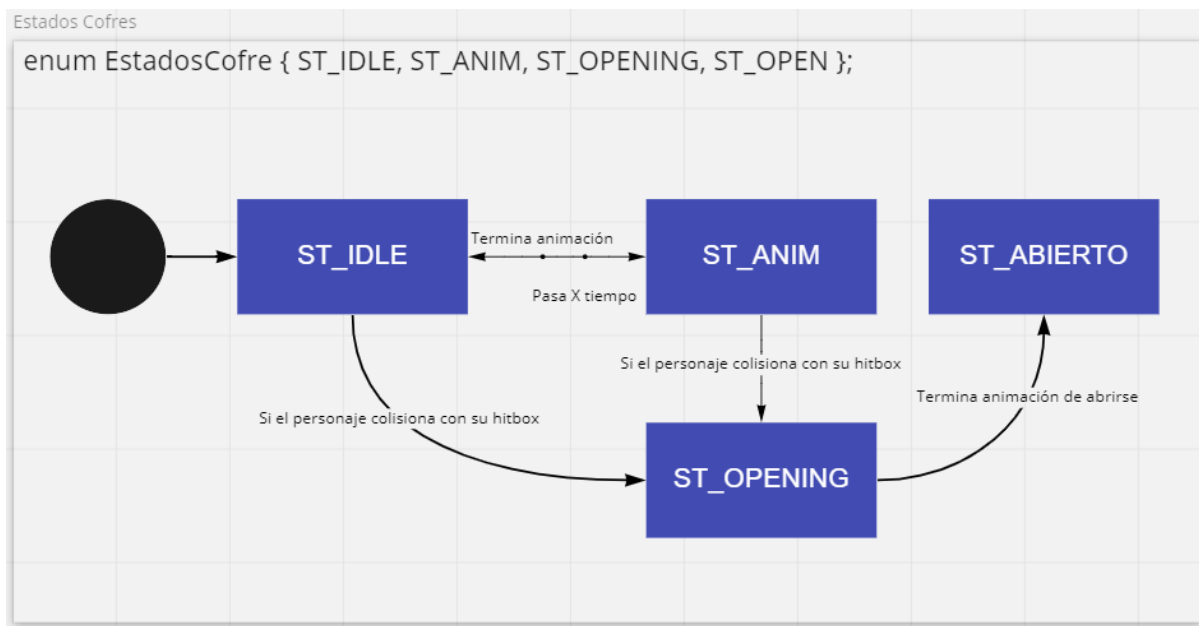
3.3 Estados personaje principal:



3.4 Estados un enemigo:



3.5 Estados objeto del juego:



4. Explicación

El flujo del juego lo conseguí con ayuda del Scene Director, en las escenas de Menu, Stats, Pausa y Game Over, cambia de escena usando objetos de clase button, con el mouse (que capturo con el InputManager) puedo detectar si está en su hitbox y si ha dado click mientras estaba en su hitbox. La escena de Game sólo cambia de escena si el personaje muere o si le da a la P (pausa).

La animación del inicio fue muy sencilla de hacer, solo contadores y input del usuario. La animación del menú también fue sencilla gracias a este for que me ahorró mucho tiempo

```
for (size_t i = 0; i < 97; i++)  
{  
    if (i < 10) {  
        _bSpriteID[i] = ResourceManager::getInstance()->loadAndGetGraphicID(Video::getInstance()->getRenderer(), ("Assets/UI/BackgroundMenu/background_00" + std::to_string(i) + ".jpg").c_str());  
    }  
    else {  
        _bSpriteID[i] = ResourceManager::getInstance()->loadAndGetGraphicID(Video::getInstance()->getRenderer(), ("Assets/UI/BackgroundMenu/background_0" + std::to_string(i) + ".jpg").c_str());  
    }  
}
```

Ya dentro del juego la progresión es así: Empiezas partida y no termina hasta que mueres o cierras el juego. El objetivo es tener la puntuación más alta lo que es cada vez más difícil porque cada vez salen más enemigos y más fuertes. La puntuación se consigue matando esos enemigos en forma de "rads" que se traducen a nivel. En el menú de Stats puedes ver tus mejores puntuaciones junto a una captura de tu última muerte. (que también se muestra en game over y en pause se muestra difuminada la partida actual)

La dificultad aumenta de esta manera: hay una variable que se llama dificultad que va aumentando mientras más niveles pasas y esa variable se usa para aumentar el porcentaje de aparición de los enemigos, por ejemplo la probabilidad de que salga un gusano gigante en el nivel 1 es 0, pero en el segundo nivel es un 50%, luego un 66% de que salga al menos uno y un 33% de que salgan 2 y así.

Los rads son unos objetos recogibles que tienen un valor de 1 de puntuación los pequeños y 3 de puntuación los grandes. Estos rads cuando se inician aparecen en una rotación aleatoria que se queda así hasta ser recogidos.

Un escorpión especial da 10 rads, estos 10 pueden terminar siendo 3 rads grandes y uno pequeño, todos pequeños, uno grande y 7 pequeños, etc.

Estos rads atraviesan paredes y se acercan a tí cuando estás lo suficientemente cerca con una función que se llama getdistance donde envías un rect y se compara con el rect que lo llama, así entonces llamo esa función en el jugador y cuando están lo suficientemente cerca uso una función de los rads que es moveTo, simplemente hace que los rads se acerquen al jugador.

Otras cosas que dejan los enemigos son vida y munición, funcionan casi igual a los rads pero estos no se mueven para que el jugador pueda elegir si quiere recogerlos o no por si tiene máxima munición o vida que no se desperdicie.

Estas puntuaciones se guardan en una clase singleton llamada Highscore donde también se calcula si la última puntuación obtenida se guarda en el top 10 o se descarta. En este menú se guarda una captura de pantalla que se hace a través de una función de "Video" que lo que hace es leer los píxeles con la función SDL_RenderReadPixels y luego los guarda en una imagen con la función SDL_SaveBMP. Al iniciar el programa se crea una imagen vacía y cuando juegas y pausas o pierdes la partida se crea una nueva captura que luego se usará en Stats, GameOver y Pause.

El personaje se mueve a todas direcciones con ayuda del InputManager y las colisiones de la función getID del mapa. La posición del mouse crea la rotación del personaje de esta manera:

Si el mouse está del centro del personaje a la derecha, el personaje está mirando a la derecha y si el mouse está del centro del personaje a la izquierda, el personaje mira a la izquierda, esa sería su "rotación" y las armas apuntan a donde esté el mouse en todo momento, si el mouse está desde el centro del personaje para arriba, las armas se pintan abajo del personaje y sino, se pintan arriba.

Para hacer que el arma apunte a el mouse lo hice de esta manera:

```
int weaponX = _Rect.x + _Rect.w / 2 - sCamera->getX();
int weaponY = _Rect.y + _Rect.h - sCamera->getY();
int delta_x = weaponX - sMouse->getX();
int delta_y = weaponY - sMouse->getY();
double angulo = (atan2(delta_y, delta_x) * 180) / 3.1416;
```

Saco el ángulo restando la posición del mouse a la posición del arma para obtener el vector y convirtiéndolo a grados.

Algo parecido hice para que el jugador pueda disparar. En este caso en la inicialización de la clase Bullet (init) para que pueda ser compartido tanto por el jugador como los enemigos.

```
_direction = finalPos - initialPos;
_direction = glm::normalize(_direction);

float angulo = ((float)((rand() % (spreadAngle * 2 + 1)) - spreadAngle) * 3.1415f / 180);
_direction.x = _direction.x * cos(angulo) - _direction.y * sin(angulo);
_direction.y = _direction.x * sin(angulo) + _direction.y * cos(angulo);

_angle = ((float)(atan2(_direction.x, _direction.y) * 180.0f / 3.1415f) - 90.0f) * -1.0f;
```

Igual que antes hago la posición final menos la posición inicial pero en este caso lo normalizo con ayuda de la librería GLM y le agrego un spreadangle que es para que la bala no salga completamente recta. Lo normalizo para poder multiplicarlo por la velocidad y así tener un movimiento de la bala, también tomo los grados para la rotación de la bala.

Las armas salen de los cofres (menos la inicial con la cual apareces) y puedes recogerlas estando encima de ellas y dándole a la E. Así cambia de estado de "en suelo" a "en mano". Para "tener" las armas, el jugador tiene un inventario que son dos punteros de arma para el arma en mano y el arma en inventario pero no en mano.

Los cofres se abren cuando pasas por encima de ellos y dejan cosas como vida o munición quitando las cápsulas de radiación que funcionan diferente. Las cápsulas de radiación solo se abren cuando las rompes con el arma y dejan radiación o enemigos que dejan radiación al morir.

Los mapas los creé con referencia de mapas reales en Tiled y los implementé como en las prácticas anteriores.

Los enemigos son todos bastante sencillos y comparten mucho comportamiento, Todos menos la cápsula de rads y el maggot nest caminan igual y algunos además de caminar normal siguen al jugador si están a cierta distancia y les atacan disparando como el Bandit o el Scorpion, el Gusano gigante simplemente se acerca a tí y te ataca a melee (daño de contacto) Reutilicé tanto la clase bullet como el vector de bullet mismo agregando un bool de `_isBulletFromPlayer` por lo que no tuve que agregar ningún comportamiento nuevo a los enemigos para que disparen.

Aprovechando el vector de bullet los enemigos son empujados por las balas cuando mueren (así como en el juego original). Dependiendo de lo grande que sea el enemigo y la distancia que ha recorrido la bala se moverán más o menos distancia. Esto lo hice reutilizando el vector de movimiento de la bala y multiplicandolo con la X-Y del enemigo y creando una "fuerza de empuje" que se calcula reutilizando la función `getdistance`. Este empuje dura el tiempo que el enemigo esté en animación de morir.

5. Fuente de información

Sprites:

https://www.sprisers-resource.com/pc_computer/nuclearthrone/

De aquí pude sacar algunos sprites aunque los tuve que recortar.

https://www.youtube.com/watch?v=p0Q_MoMsKII

De aquí saqué la animación del fondo del menú principal, la descargué y pasé a imágenes para luego hacer la animación en C++

<https://yal.cc/r/17/yytextureview/>

Con esta página pude sacar la mayoría de los sprites, arrastras el ejecutable y te muestra los sprites que tiene pero tienes que recortar todo tú mismo y no están en orden

Controles:

<https://nuclear-throne.fandom.com/wiki/Controls>

Música:

https://www.reddit.com/r/NuclearThrone/comments/dt2bdp/nuclear_throne_high_quality_music_mod/

+Info del juego:

https://nuclear-throne.fandom.com/wiki/Nuclear_Throne_Wiki

Botones-Mouse SDL2:

<https://www.youtube.com/watch?v=DKx6DxGY5ZA>