UNIVERSITÀ DELLA CALABRIA
DIPARTIMENTO DI MATEMATICA E INFORMATICA

Game Design Document

Beat the Dungeon

By DanteScot's Production

Academic Year 2023 – 2024

Professor

Carmelo Macrì

Team Members

Kevin Corasaniti (230673)

INDEX

Game Overview.....4
General Information.....4
External Influences.....5
Story and Setting.....6
Summary.....6
Characters.....7
The adventurer.....7
808.....7
Spider.....7
Stephen.....7
Jerry.....7
Gameplay.....8
Introduction.....8
Controls.....9
Game System Mechanics.....10
Entity's Mechanics.....11
Player.....11
808.....11
Spider.....11
Stephen.....12
Jerry.....12
Spikes.....12
UI Design.....13
Main Menu.....13
In game.....14
Level Design.....16
Rooms.....16
Tutorial.....16
Lobby.....16

Levels.....16
Lighting.....17
Art Assets.....18
Sprites Sheets.....18
Single Sprites.....19
Tilemaps.....19
Animations.....20
Example code based animation.....20
Example animator base animation.....20
VFX.....21
Fireflies.....21
Screen Shake.....21
SFX and Music.....22
Implementation..... 23
Player.....23
PlayerManager.....23
PlayerController.....25
Global.....26
BeatManager.....26
RythmedObject.....27
Future perspectives.....28

Game Overview

General Information

- Single player
- Genre: Roguelike, Rhythm Action RPG, Picchiaduro, Sparatutto, Adventure
- Platform: PC

Title meaning: The title plays on the fact that beat can both refer to defeating the dungeon and to the musical term “beat”

It doesn’t have an official logo yet.

External Influences

The three major influences in the game are:

- **Hi-Fi Rush** - Rhythmic mechanic
- **The Binding of Isaac** - Style and general gameplay
- **Hades** - Some mechanics like the permanent upgrade

Story and Setting

Summary

The adventurer finds himself inside this laboratory without knowing how he got there. As he approaches the strange object in the center of the room, the laboratory seems to come to life and 808 makes its appearance.

808 is a service robot in the facility that will alert the adventurer to the fact that all the other robots in the facility have gone mad after the humans disappeared and that everything there is synced with the music.

The adventurer's only chance to save himself is to become strong enough to overcome all the adversities that will arise during the search for the exit

Characters

The adventurer

He doesn’t remember how he got there. The last thing that he remembers is that he was running in the forest. For some reason he has a sword with him.

808

It is a service robot of the facility. He is the last peaceful robot and wants to help the adventurer to get out. He even knows how to cure any wound somehow.

Spider

It was once used to keep everything clean and in order, now it seems that he finds you as the thing to “clean”.

Stephen

At one time he was also a service robot, now as soon as he sees a human he would run full speed at him without stopping.

Jerry

It was a humanoid robot created to defend the facility, it just changed what he perceived as danger.

Gameplay

Introduction

The game has a 2D top-down view where you can move, attack and shoot along the X and Y axis.

The movement can be done even diagonally in opposites to the attack/shoot that is fixed to the axis (top-down left-right), to make the bullet go diagonally you have to give them momentum with the movement at the time of the shoot.

The facility will change every game (not implemented yet).

The gameplay moves around the music, every enemy and the player attacks based on the beat of the music. Some enemies even got the animations fixed with the music. The only way for the player to shoot is making a perfect attack on the beat.

Controls

KEY	ACTION
W	FORWARD
A	LEFT
S	BACKWARD
D	RIGHT
E	INTERACT
SPACE	TALK / NEXT SENTENCE
ESC	BACK / PAUSE

Game System Mechanics

- **BEAT:** It represents the BPM of the current song. Everything that must be synced with the music will use this mechanic
- **LOAD/SAVE:** The game will automatically save/load the game data every time that you come back to the lobby
- **SCENE MANAGEMENT:** Every macro-area is a different scene in unity ad will be resetted every time you change it
- **DIALOGUE SYSTEM:** It allows the player to talk with the NPCs and will trigger some events too
- **UPGRADE SYSTEM:** The player can upgrade the character in order to make it stronger. Every upgrade will cost some “gear” that will be found around the level or when you clear a room
- **POWER-UPS SYSTEM:** Around the level you’ll find objects that will give you special power, they’ll last only for the current run and will be lost when you’ll come back to the lobby

Entity’s Mechanics

Player

- **HEALTH:** Represents the number of hits the player can receive before dying, some hits from stronger enemies are worth double
- **MOVING:** You can use WASD to move around
- **ATTACK:** You can attack with the “Directional Arrow”, if the attack is perfectly timed you’ll shoot too. You can attack only on time, so, if you attack out of time you’ll have to wait the beat and will not shoot

808

- **GLOW:** It will glow every beat
- **MOVING:** It will always move towards the player using the NavMesh

Spider

- **HEALTH:** Represents the life remaining, if it goes below 0 the enemy dies
- **MOVING:** If the player is too far it will move randomly around his position otherwise will move towards the player using the NavMesh
- **ATTACK:** It only make contact damage for every beat that it’s touching the player

Stephen

- **HEALTH:** Represents the life remaining, if it goes below 0 the enemy dies

- **MOVING:** It will constantly move towards the player the NavMesh
- **ATTACK:** It only make contact damage for every beat that it's touching the player

Jerry

- **HEALTH:** Represents the life remaining, if it goes below 0 the enemy dies
- **MOVING:** It will move towards one of the pre-chosen positions and will emit a "earthquake" every step the NavMesh
- **ATTACK:** It makes both contact damage for every beat that it's touching the player as also ranged attack. The ranged attack of one bullet towards the player if in first phase or three bullet in second phase
- **PHASE:** If it has less than half of the starting life it will enter the second phase, it will, besides the extra attack, double his damage, grow bigger and make a stronger earthquake

Spikes

- **RETRACTED/PREPARATION:** It will not make any damage
- **EXTENDED:** It will make damage to the player
- **PHASE:** It will change phase based on the beat

UI Design

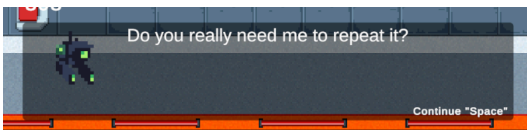
Main Menu

- **PLAY:** Starts the game, it will automatically detect any previous saves
- **SETTINGS:** Settings for the game
- **CREDITS:** Credits for artists whose works were used in the game
- **QUIT:** Quit the game

The menu WILL CHANGE based on the real time of the day (between 6-18 is day otherwise is night)

In game

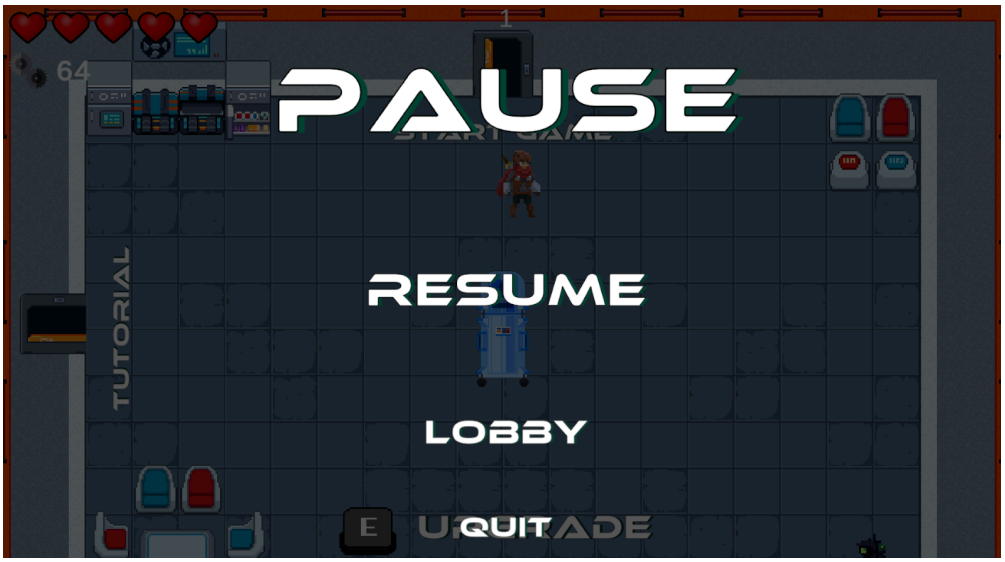
- Near NPCs that you can talk with will appear the notification to talk and if the button is pressed will actually start the dialogue



- Near objects that you can interact with will appear the notification to use it

- After interacting with the incubator you'll access the upgrade interface with the icon of the ability (a tooltip will appear hovering the image), some dot representing the level (+1 level for every colored dot), the price for the upgrade and the upgrade button, you can exit it pressing "Esc"

- Pressing “Esc” outside of an interaction will pause the game. Three button will appear
    - **Resume:** Will resume the game
    - **Lobby:** Will bring you to the lobby
    - **Quit:** Will bring you to the main menu



- When you pick a powerup in game you’ll get the name of it and the description

Level Design

Rooms

Tutorial

It’s an empty room with various trigger in order to make the player try every mechanics

Lobby

It contains the access to the tutorial room as well as the access to the levels. It also contains the incubator for the upgrades and a modified version of 808 that you can talk with.
There even is all the thing for the starting “animation” as lights and sound sources and some decorations

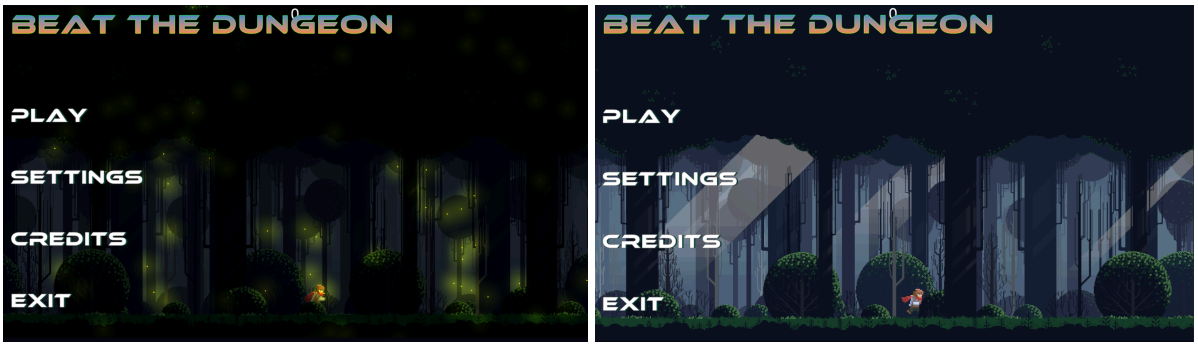
Levels

The level will be procedurally generated (not yet) making every run unique. Will always be present only a starting empty room, various general rooms (that could be with just traps, with enemies, or gears/powerups), and a boss room possibly containing some boss-specific characteristics.

Lighting

In order to make lights work in a 2D ambient an external package with extra adjustment was necessary. Things such as Shadow or the “Fireflies” required manual adjustment.
When all the lights are disabled it automatically goes back to the standard unity illumination and vice versa.

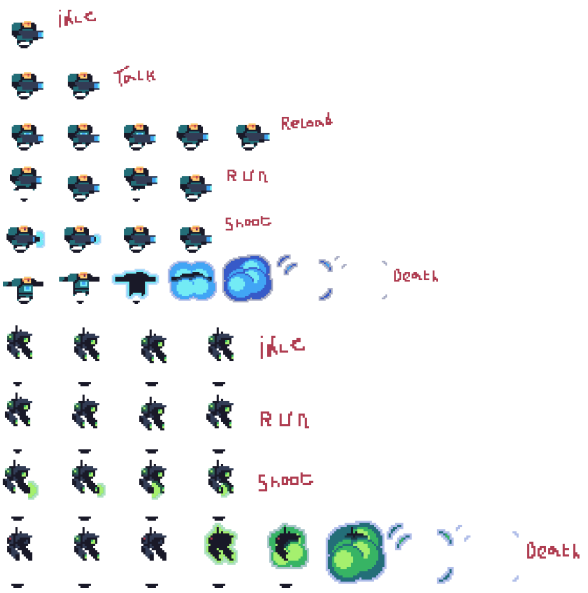
Same scene, different real time hour and different lighting system



Art Assets

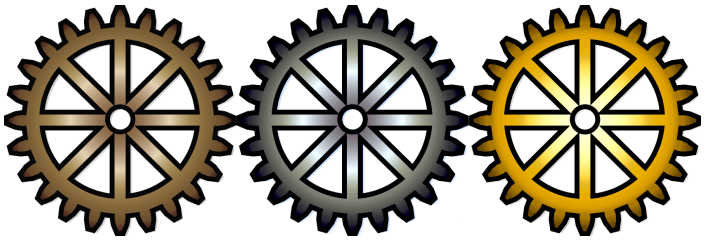
Sprites Sheets

The majority of the sprites are taken from things called “Sprite Sheets” which are multiple sprites in one big image, you can access the single image using the tool “Sprite Editor” in unity. Unfortunately not always the sheets are well made, in that case you must manually adjust them.
Here is some of the sheets used in the game:



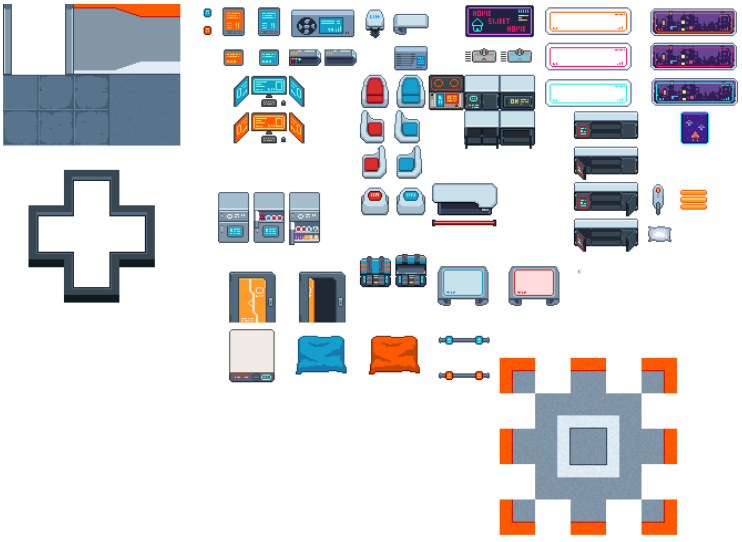
Single Sprites

Some of the sprites were taken from single images and edited in a software called “Gimp” that is similar to “Photoshop”. Sometimes the image was just manually removed from the background (ex. incubator) while other times they were actually edited (ex. Gears was recolored and was created the relative sprite sheet)



Tilemaps

To make easier the creation of the ambients tilemaps were used, edited as well using gimp to reorganize some object and make like the corners for the walls

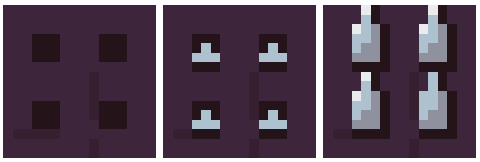


Animations

Animations in a 2D environment are just the fast change of the image, some of them were managed by script while others were managed by the animator of unity

Example code based animation

The spike were animate by code changing sprite every beat



Example animator base animation

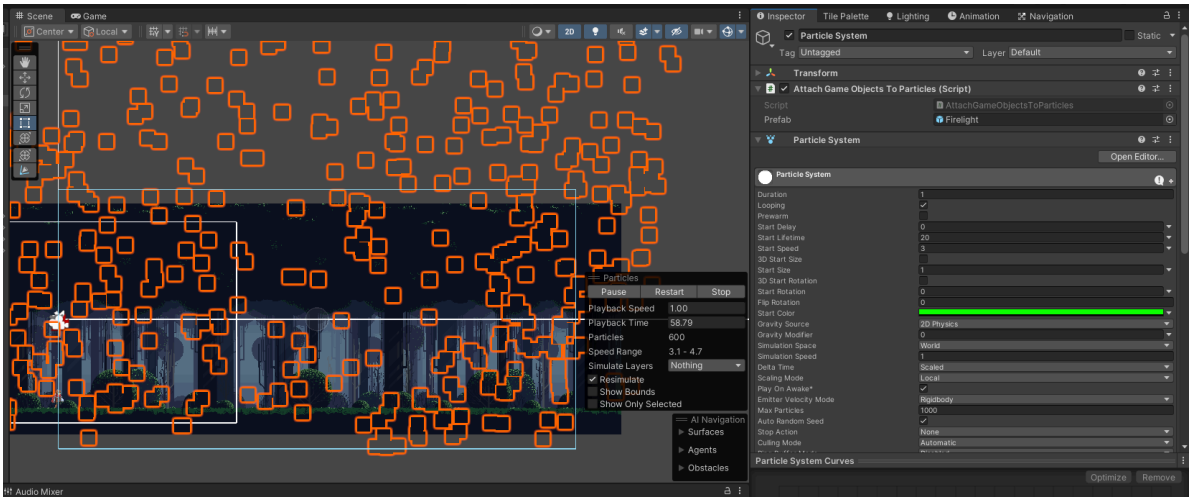
The player is one of the objects with the animation made using the animator, the following animation were created

- **Player_idle:** when the player is idle
- **Player_run:** when the player is moving
- **Player_hurt:** when the player has been hitted
- **Player_Pickup:** when the player picks up something
- **Player_attack:** when the player attack
- **Player_die:** when the player dies

VFX

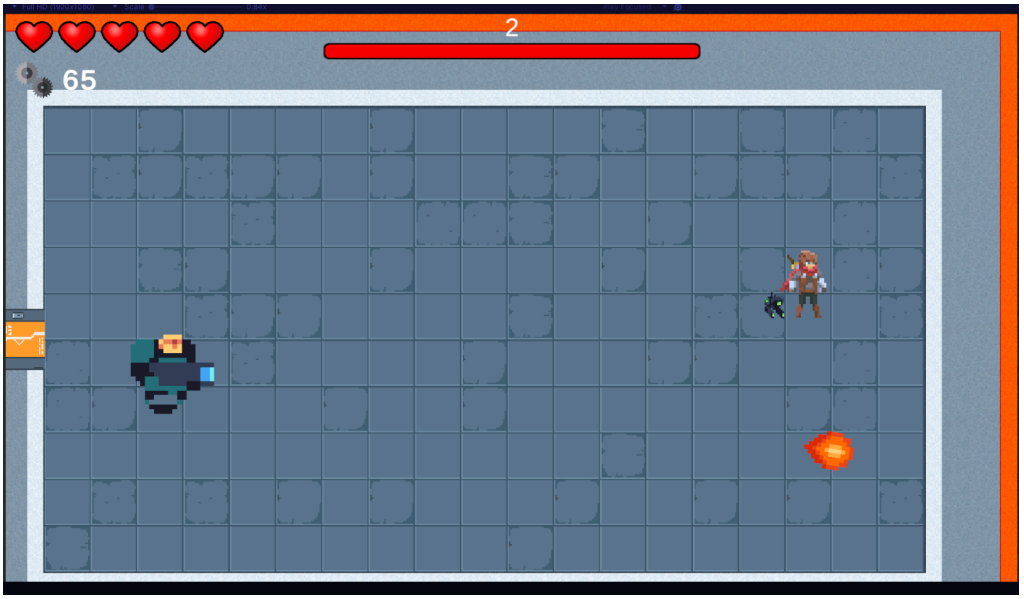
Fireflies

It’s used in the main menu during night time, the Unity’s Particles System spawns transparent particles and after that a gameobject with the light is benign attached to it



Screen Shake

It’s used by Jerry to make feel the weight of his steps



SFX and Music

Simple sfx was made using a program called “SFXR” while more complex sounds were taken from soundcloud. The music was taken from various places but except for the loop given to the professor they were edited in some way to create loops or distortions.

An important consideration should be made:
Not all songs can be used in the game, as everything is synchronized with the music, we need to know the exact BPM of the song, which for some has been calculated manually, but it is not possible for all songs. Once we have the BPM we have to figure out if it is good for that situation, too low and the game will be boring and slow, too high and it will become difficult to handle the whole thing and it will be annoying.

Implementation

Here we'll discuss about some of the most important classes

Player

PlayerManager

The Player Manager is responsible for managing every aspect of the player and keeping the data between levels. It's Singleton and will not be destroyed during the scene changing.

Every skill has it's relative getter, setter and leveled getter

- **Getter:** simply returns the skill value
- **Setter:** sets the skill value and notify the observers
- **Leveled Getter:** returns the skill value applying the relative upgrade based on it's level

This class uses the pattern *Observer*.

InstantiatePrefab(string prefabPath) is a support function for the *Power* class.

EndGame() will reset all the starting stats and saves the gears obtained.

moveSpeed	Starting speed of the player
moveSpeedLevel	Relative skill level
maxHealth	Starting max health of the player
maxHealthLevel	Relative skill level
currentHealth	Remaining life of the player
luck	Starting luck of the player
luckLevel	Relative skill level
baseAttackDamage	Starting attack dmg of the player
baseAttackDamageLevel	Relative skill level
attackSpeed	Starting attack speed of the player
attackSpeedLevel	Relative skill level
attackRange	Starting attack range of the player
attackRangeLevel	Relative skill level
gears	Gears collected by the player
startingStats	Keep all the starting stats
observers	List of observer
player	Transform of the player
currentRoom	Room where the player currently is
minions	Transform parent of all minions

PlayerController

The class Player Controller uses the data from the Player Manager in order to make a faster playable character.

Function as *PrepareAttack()* and *Attack()* assures that the attack is always in time.

critWindow() will manage when the player is able to attack.

CheckMelee() check for melee attack.

SetCurrentRoom() manages the camera boundaries.

critTimeWindow	Grace time for crits
attackSpeed	Speed for the bullet
baseAttackDamage	Melee/Ranged attack damage
finalDamage	“Normalize” the damage
attackRange	Range of ranged attack
moveSpeed	Player movement speed
canAttack	Boolean to check if the player can attack
isCrit	Boolean to check if it’s a critical attack
isAttacking	Boolean to check if the player is attacking
attackDirection	Direction of the attack
powers	List of power of the player
movement	Check the movement of the player

Global

BeatManager

This class is responsible for the management of the main audio and the relative synchronization of all the other objects. It's Singleton and will not be destroyed during the scene changing. It uses a subclass called **Interval**.

Interval.CheckForNewInterval() will be called every *Update()* of the BeatManager to check if the correct interval for the beat has passed, in case, sends the event.

bpm	BPM of the current song
interval	Interval class to manage the beat
Interval.steps	Steps between intervals
Interval.LastInterval	Check if it's a new interval

RythmedObject

Almost every class that wants to go on time extends this class. It implements all the basic things to manage the timed call.

OnBeat() it's called every beat and assures that the extending class is called only on the right one.

Trigger() it's the actual function called on the right beat.

ResetCounter() auxiliary function for *OnBeat()*

index	Beat to wait every time
waitBeat	Beat to wait the first time
beatToWait	Beat to wait before calling *Trigger()*

Future perspectives

There’s a long list of the things that I wanted to have already implemented and an even longer list of things I want to implement in the future.
Here I’ll share some of those things in no particular order.

- Make the level generation procedural.
- Add quick time events.
- Add more powers.
- Rewrite some classes in order to make them cleaner and more optimized.
- Variety to the levels with more objects in every room.
- Add More enemies/bosses.
- Implement the story.
- Make the game “play” with players’s computer files