At Debt's Door Game Design Document

Edison Cai - 20241135

Sergiu Mereacre - 20238029

Bayan Nezamabad - 20251971

Jack O’Brien - 20245777

Table of Contents

Introduction/ Overview

At Debt's Door (or A.D.D.) is a fast-paced shooter/strategy game for the PC platform. The goal of the game is to sneak in and infiltrate various locations such as banks, jewelry stores, convenient stores etc., and loot as much as you can. This takes place from a top-down perspective where the camera follows the player.

Tools and weapons can be acquired throughout the game to help the player's pursuits and improve their experience.

While stealth is the opportune approach, it isn't necessary. If the player desires it, they can choose an offensive strategy instead. Weapons such as pistols and knives help you carry out this risky task.

NOTE: All code snippets shown is written in GDscript.

Story Summarised

A.D.D. is set in the modern American era, where money rules and you, “The Player”, live your life desperately trying to gather it.

After some poor choices and even worse gambles, you find yourself in debt to the infamous Starman (our main antagonist), America’s most renowned mob boss. Left with a limited time frame, “The Player” has no choice but to resort to a life of crime and life-threatening activities.

Every day, “The Player” will perform actions that puts his life and freedom in danger for the ability to pay the money owed back.

Story In-Depth

- Starting from a simple robbery of a gas station convenience store, “The Player” quickly moves up the ladder and with the right equipment, high-end banks are ripe for the picking. All of this is essential of course, as the time frame in-order-to pay “Starman” creeps ever closer.
- These robberies are not easy tasks of course, as the cops that occupy and guard these banks are no pushovers. Once discovered, “The Player” has limited time to escape with the loot before being knocked down by police.
- While the story is quite simple, an opening cutscene helps set the dark, grimy tone of the game and the urgency of acquiring the money. Through the game’s arms supplier, “The Dealer”, small snippets of Starman’s lore are discovered, helping you understand how a simple man became the monster that he is today.
- While there is a definitive ending for the game, there is also an ending where the player is unable to pay back the money owed in time. This helps show players there is consequences and punishment for failing the task at hand.

Features That Make The Game Compelling And Unique(Importance of Money)

The core theme of the game is money. This means a lot of the game's features will revolve around money and how it can be used.

Money motivates the player through Starman's threats but players will find it difficult to collect the amount owed without spending a bit of it on useful items.

Features That Make The Game Compelling And Unique(Scarcity and Strategy)

This creates an interesting conflict in the player's mind. Should they play it safe and leave themselves vulnerable or should they invest some of it in equipment to defend themselves?

Players should think ahead when buying weapons as they have durability and will be destroyed if they are used too many times. So, it is important to use items appropriately.

Features That Make The Game Compelling And Unique(Encumberment)

As you gather more money in a stage, you will get slowed down by the weight of the money. If you do not want to get slowed down, you might have to throw some of it away to flee from a dangerous situation which adds another layer of strategy.

We really wanted to incorporate the notion of “High Risk, High Reward” into our game.

```
#Function for throwing money away in chunks
func throw_money():
    #Checks if the player has enough money to throw
    if stats.money >= 50:
        #Determine the thrown money's velocity using the mouse's position
        var thrownMoneyVelocity = mouse_position.normalized() * 700
        #Allow the world to generate the money using the velocity that was calculated
        emit_signal("throw_money", thrownMoneyVelocity)
        #Decrease the amount of money owned
        change_money(-50)
    #Handling the case where the player doesn't have enough money
    else:
        print("You don't have 50 bucks to throw away!")
```

Controls During Play

- WASD/Arrow Keys – Walking
- Mouse position – Decide player's direction and focus on what object to interact with
- Left click – Attack
- Right click – Throw money
- Spacebar – Roll
- E – Interact with object and people
- Q – Use heal item
- Escape Key - Pause

Game Mechanics

- Can move around in 2D space.
- Can't run but can roll to gain extra speed for a while.
- You slow down as you gather more money, but you can throw some away.
- When approaching an item/person that can be interacted with, an icon will pop up, signifying that they are interactive.

- Can steal from people and objects such as safes.
- There are collectibles in some stages that gravitate towards the player once they are close enough to it.
- Can kill people and destroy certain objects such as security cameras.
- Can use melee and ranged weapons.
- Can cycle through weapons.
- Weapons have durability.
- Shops to buy items from.

Game Mechanics - Summary

In summary, A.D.D. exhibits many features found in other top-down 2D shooters but incorporates some that are unique to itself.

It is important for us to express our creativity and make the game even more enjoyable than a typical 2D top-down shooter.

Game AI

All the AI in A.D.D. can be broken down into 2 different types: Passive and Aggressive AI. Both types can be killed by the player.

Passive AI

When the player is being a model citizen and not causing trouble, such as attacking or stealing, passive NPCs will randomly switch between their ”IDLE” and ”WANDER” states.

Using a timer with intervals of several seconds, every time it counts down to zero, one of the two states will be chosen, and the timer will be reset.

Shuffling states

```
#Creates a timer that picks a random state every couple of seconds
timer.set_wait_time(3.0)
timer.set_one_shot(false)
timer.connect("timeout", self, "update_wander")
add_child(timer)
timer.start()
state = pick_random_state([IDLE, WANDER])
```

Randomise target position to walk to

```
#Updates the target position but now there's no more state shuffling
func update_target_position():
    target_position = global_position + Vector2(rand_range(-wander_range, wander_range), rand_range(-wander_range, wander_range))
```

Passive AI

However, if the player is in the NPC’s area of detection while committing crime, their state will flip to ”PANIC”, which will alert the stage that the player has been detected, which will consequently spawn in enemy cops.

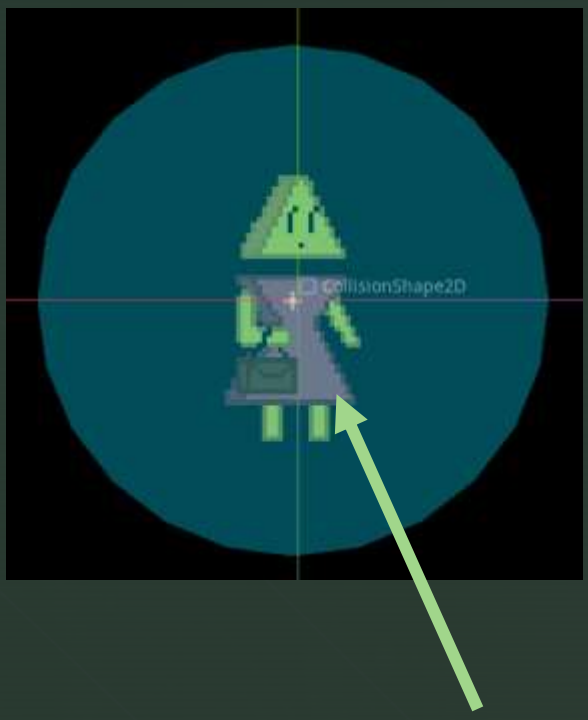
If they detect other NPCs panicking, their state will also be set to "PANIC".

For detecting other NPCs

For detecting players. By giving it a direction, players can perform stealth actions while passive NPCs are not "looking".

Passive AI

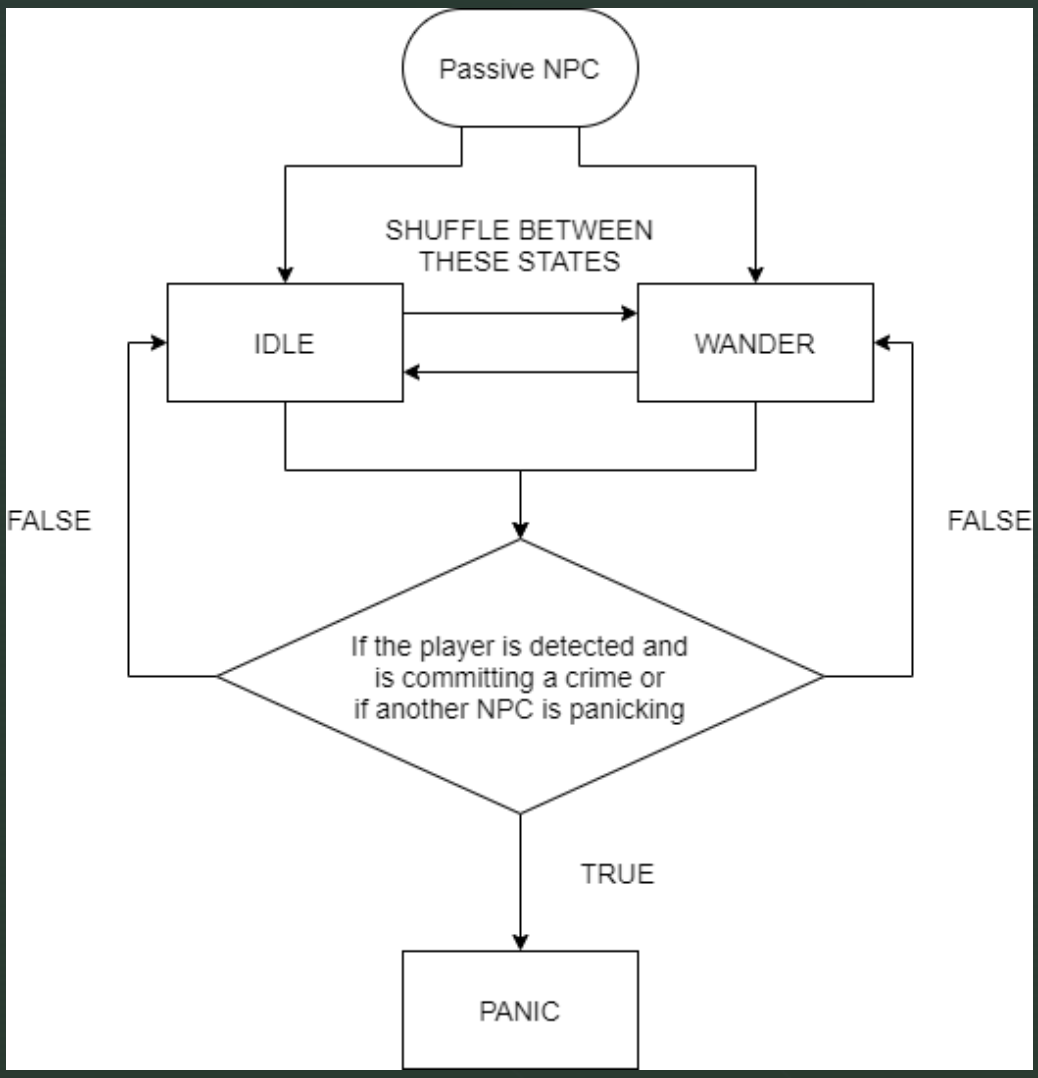
Some NPCs can also be pickpocketed by holding down the "interact" button while in their interactive range.



Separate area that once the player enters, they can start pickpocketing.

Make their panic speeds easily editable in the game engine. That way, both NPCs that will run away(panicSpeed > 0) and NPCs that will cower(panicSpeed = 0) when panicking can be implemented.

Passive AI


Passive NPC
SHUFFLE BETWEEN
THESE STATES
IDLE
WANDER
FALSE
FALSE
If the player is detected and
is committing a crime or
if another NPC is panicking
TRUE
PANIC

Aggressive AI

Once the world has been alerted of the player’s crimes, cops will spawn and target the player.

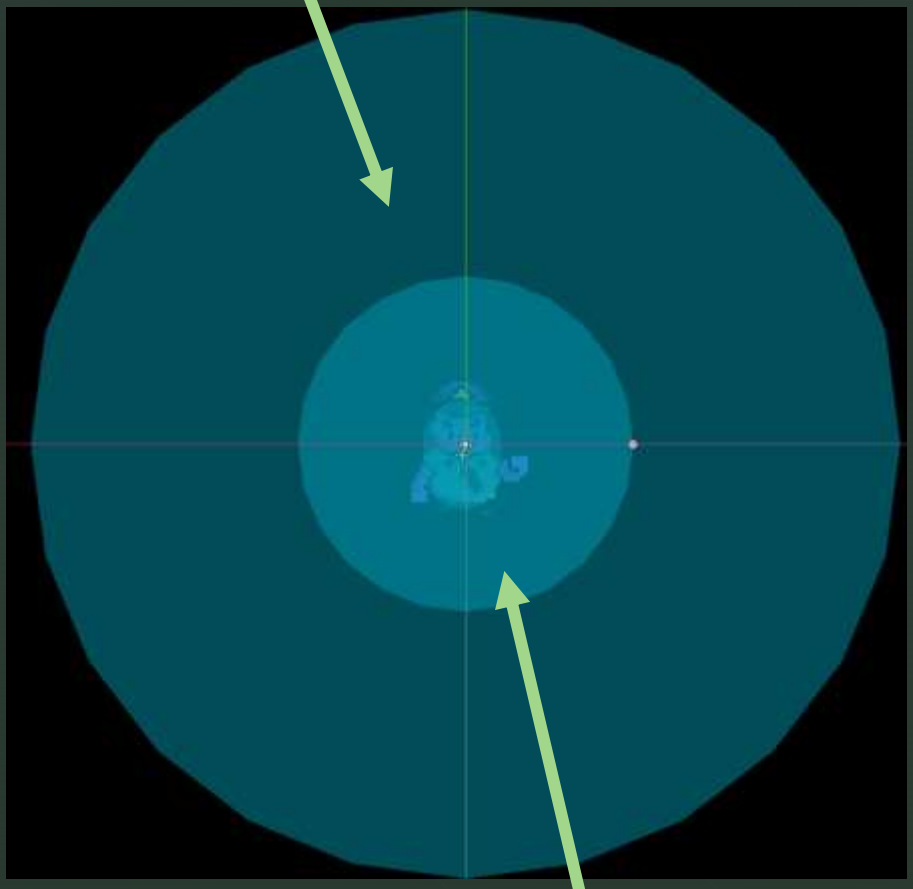
They also have an area of detection and if the player is within this area, they will be chased down mercilessly.

If they get close enough to the player, they will start attacking with whatever weapon is in their possession.

If the player manages to escape from their area of detection, the cops’ states will also flip between ”IDLE” and ”WANDER”, looking for the player.

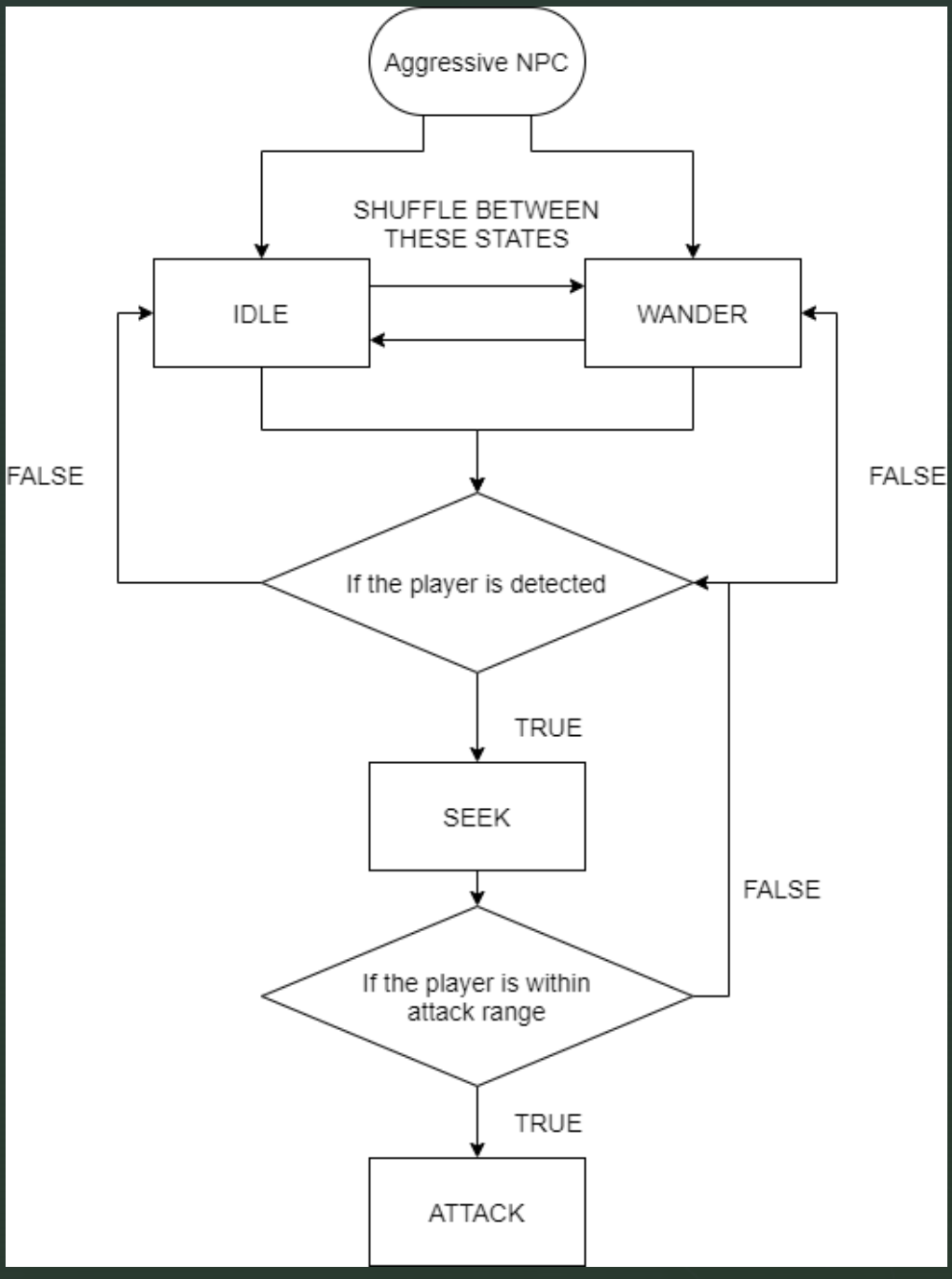
Aggressive AI

Player detection area



When the player enters this area, start attacking

Aggressive AI

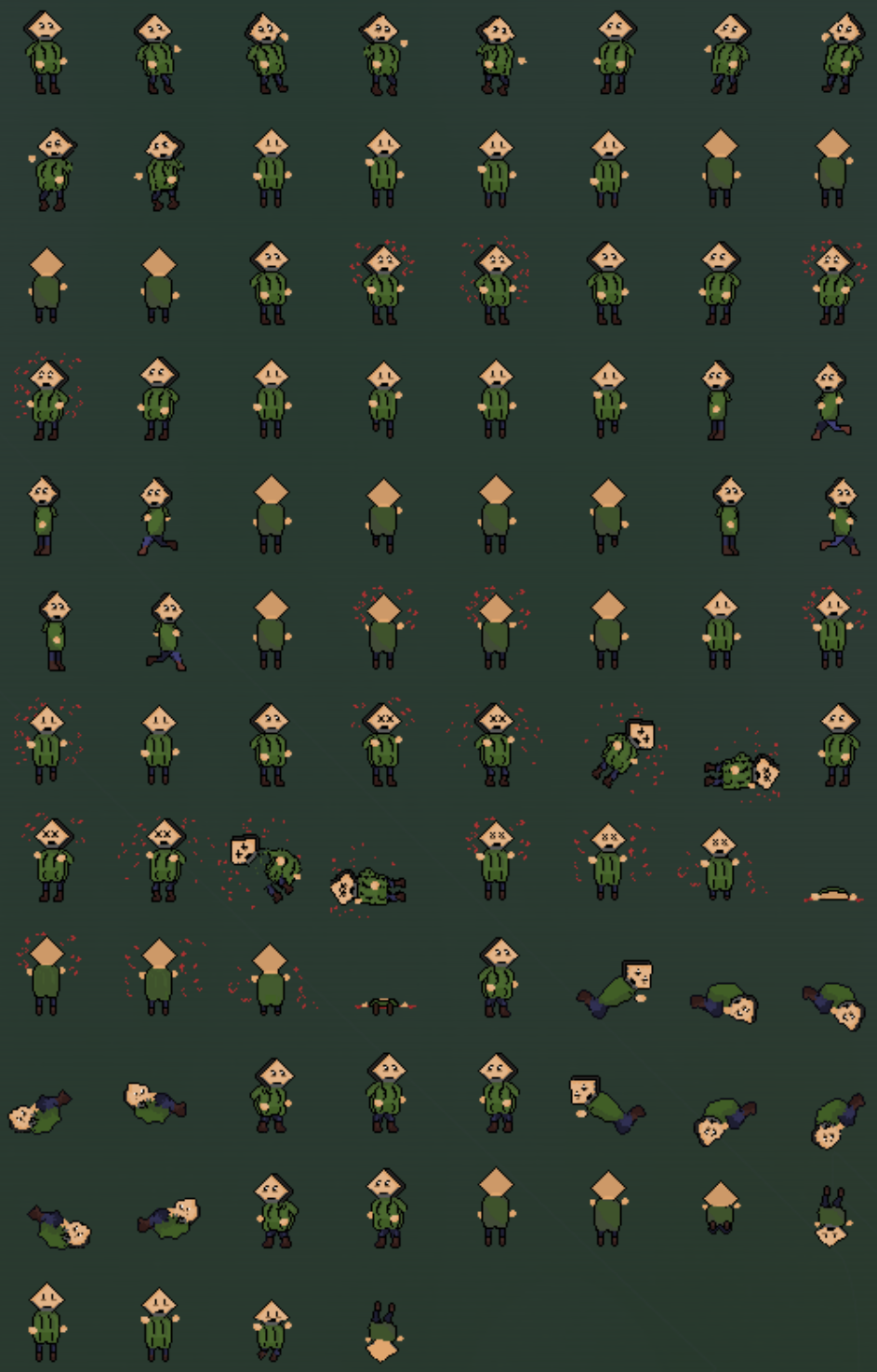


Game Elements

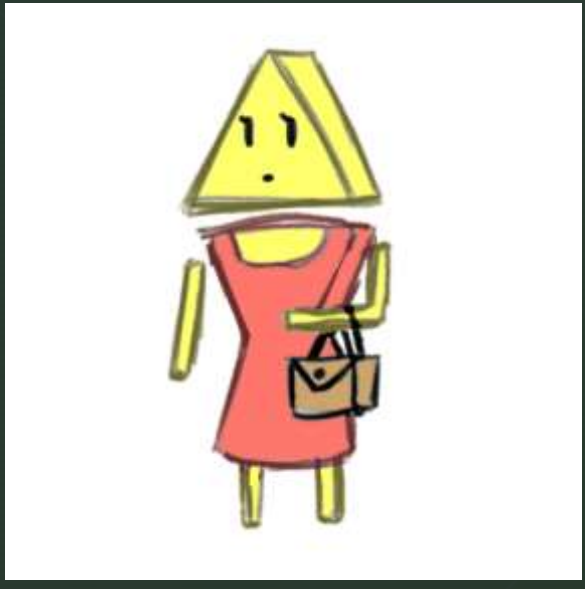
Game Elements - Characters

Here are the characters that the player can encounter during their time playing.

Characters - Player



Characters – Classy Woman



Characters - Timmy

Characters -Shopkeeper

Characters – Police(Default)

Characters – Police(Beard)

Characters – Police(Smoker)

Characters – Police(Sunglasses)

Characters - Dealer

Game Elements - Equipment

Modular, portable

Every player and NPC should be able to equip any weapon. All they must do is to call its individual “attack” function to activate it.

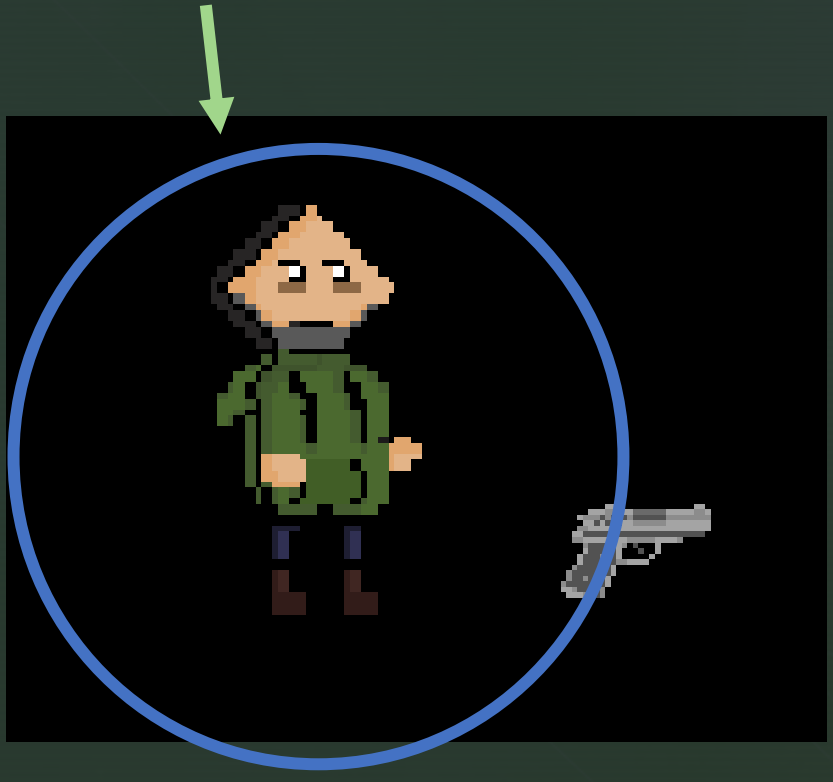
Position will be constrained to a radius around the user, allowing 360 degrees of freedom.

Every weapon will have animations that play when they are used.

Weapons can either have 4 variations of sprites it can flip between based on whether it’s facing up, down, left or right or 2 variations by rotating the sprite within the game engine and flipping horizontally/vertically as needed.

Game Elements - Equipment

All weapons will be constrained to a radius around the character.



For the player, the weapon can be moved using the mouse's position.

Equipment – Melee Weapons

- Variables
  - Durability, max_durability
- Attack function
  - Play the attack animation and appropriate sound effects.
- Durability function
  - Call decrease_durability if the player is using the weapon on an NPC.
  - If the durability goes to 0 or below, remove it and play a broken effect to signify to the player that it is broken.

Equipment – Ranged Weapons

- Variables
  - AmmoCount, bulletSpeed

- Attack function
  - Same as the melee weapons except this time, if there is ammo left, generate an instance of a bullet and give it a velocity that's determined by the ranged weapon's bulletSpeed variable.

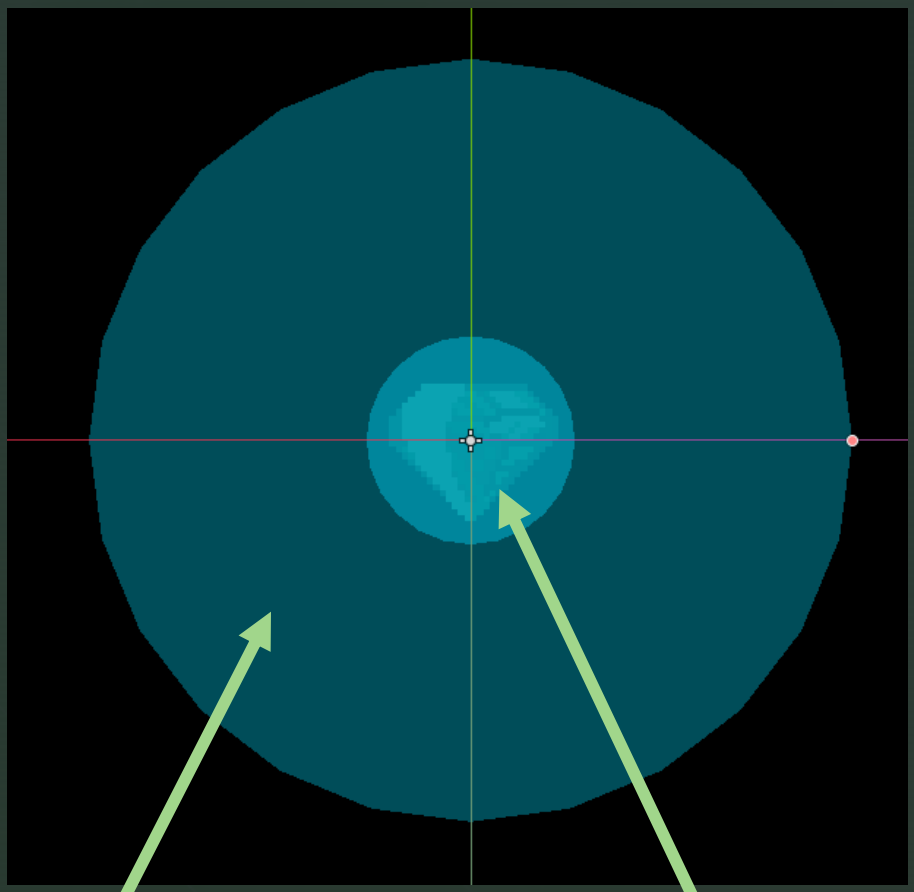
- Ammo
  - Instead of durability, call a decrease_ammo_count function to decrement ammoCount. Unlike melee weapons, ranged weapons won't get destroyed. They will just become unusable once they run out of ammo.
  - For the ammo, it contains a hitbox area that will deal damage once it enters a hurtbox. When it hits any collision body(including walls), it will be deleted.

Game Elements - Objects

Here are the items and objects that the player can encounter during their time playing.

Objects – Loot That Gravitates Towards The Player

Objects – Loot That Gravitates Towards The Player



Trigger collect area

Player detection area

How the item can move towards the player:

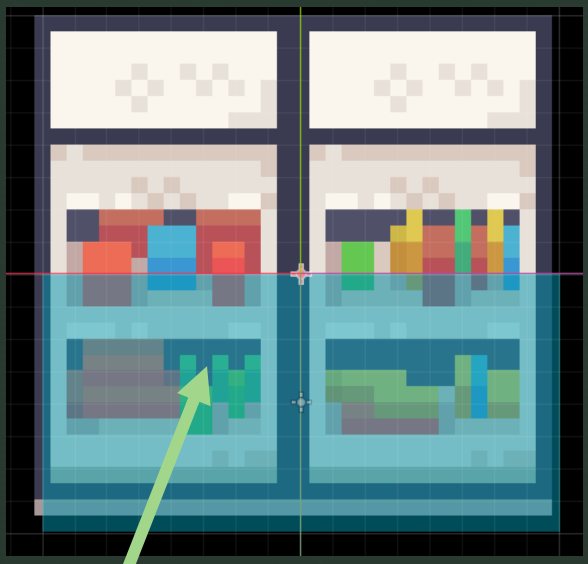
```gdscript
#Accelerates towards player once he enters the playerDetection area
func _physics_process(delta):
    for body in playerDetection.get_overlapping_bodies():
        if body.is_in_group("player"):
            var vector = get_tree().get_nodes_in_group("player")[0].global_position - global_position
            #Gradually gain speed until the max velocity is reached
            velocity = velocity.move_toward(vector * MAX_SPEED, ACCELERATION * delta)
    velocity = velocity.move_toward(Vector2.ZERO, FRICTION * delta)
    move()
```

Objects – Decorative Objects

Objects – Decorative Objects

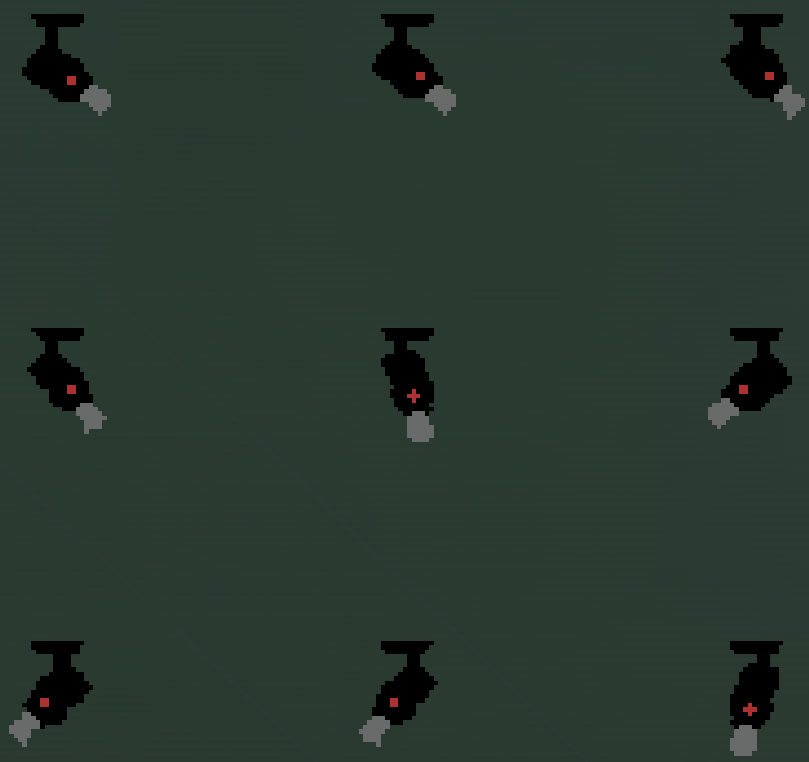
Since most of these objects are just for decoration, it’s fine to just give them collision shapes as necessary so that the player can't go through them.

However, some might have extra functionalities such as doors and safes containing money that can be robbed.



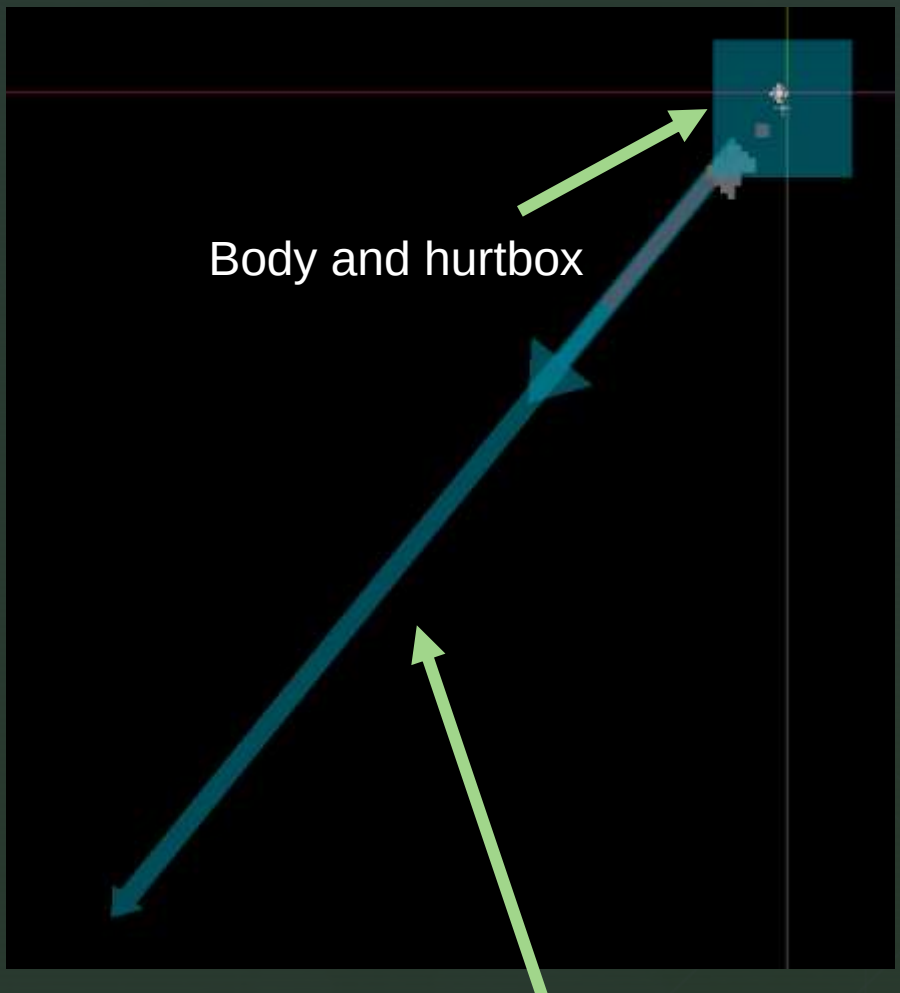
Collision

Objects – Security Cameras



Objects – Security Cameras

- Like NPCs, cameras also have health and will be destroyed once its HP reaches 0.
- The laser coming out of the camera should be blocked by bodies such as walls.
- For some cameras, if the player is detected while committing crimes, start spawning cops.
- Other cameras will be more sensitive and alert the cops as soon as the player is detected at all. Place these in places with more loot and allow players to distinguish between the 2 camera types using different colored lasers.



Game Elements - Game Economy and Statistics

- This link will send you to our Github repository, where you will find an Excel sheet documenting stats of levels, characters and items.
- Developers should maintain a good balance to ensure fair and fun gameplay.

Game Elements - Audio

Sound effects needed:

- Walking, hurt and death sounds for every character.
- Rolling sound for the player.
- Attacking and breaking sounds for every weapon.
- Whirring sound for the security camera.
- Sounds for UI elements such as buttons and selecting items in the loadout screen.
- An alert sound when the world is aware of the player's crime, warning them at cops are coming.
- “Kaching” sounds for any money related actions.

Music needed:

- To fit the theme of the game, music containing rap beats would be appropriate as it adds intensity.
- Incorporate some calmer music for moments such as accessing the loadout screen. This helps the player focus as they choose their weapons.

Game Elements – Levels(1)

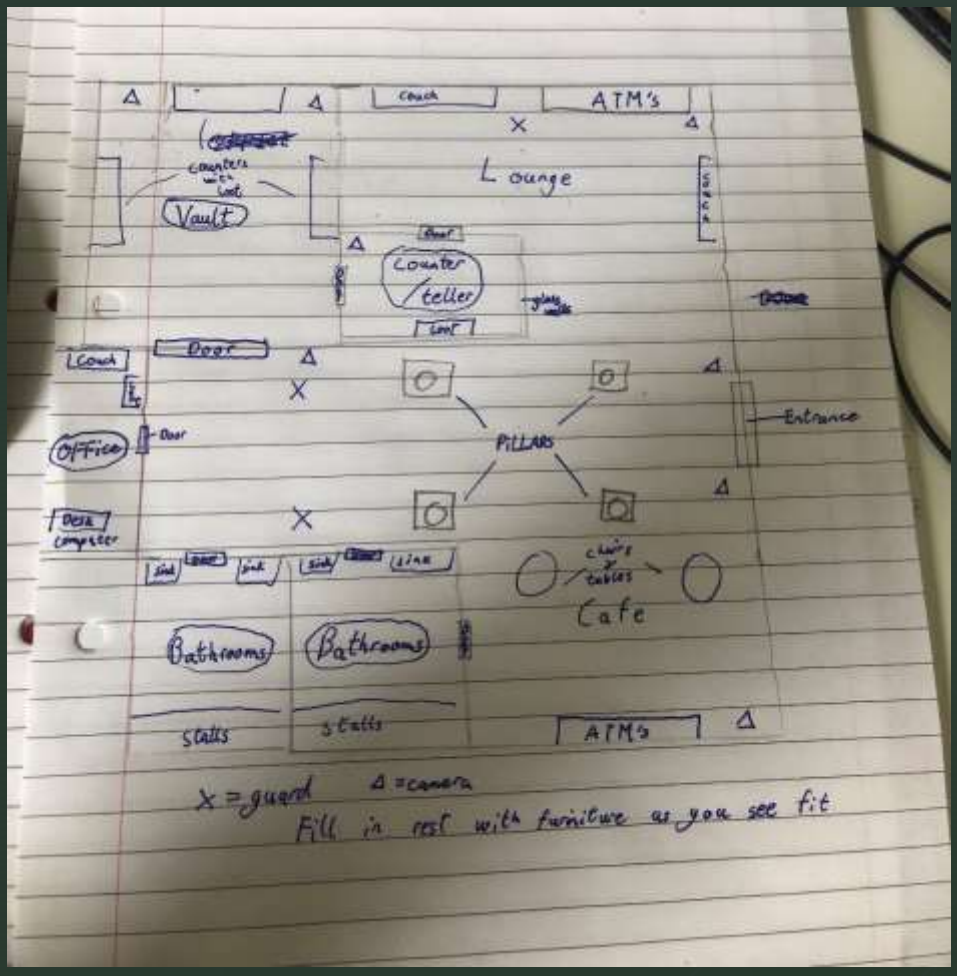
- From the main menu, players can choose to play a tutorial stage that will teach the features of the game.
- This is a more immersive way of understanding the features over just reading a page of instructions.
- Introduce the basic features such as movement first, before teaching the other features such as stealing and throwing money.

Game Elements – Levels(2)

- At the minimum, there should be a convenience store level. This will act as an introductory level for the player to get comfortable with the mechanics.
- After making the convenience store level, developers can move on to bank and jewelry store levels.
- Existing floor plans can be useful as inspiration.

Game Elements – Levels(3)

- A plan for a bank stage. Contains more loot than the convenience store but the security is also greatly increased.

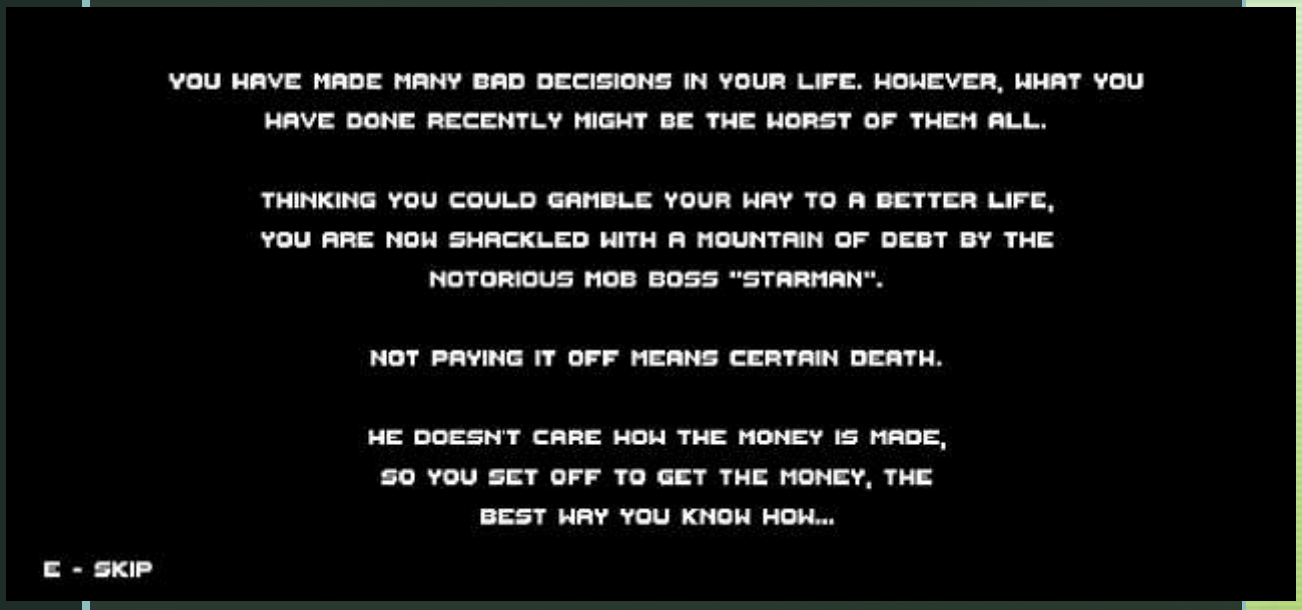


Game Progression(1)

After clicking “Start”, give a brief intro of the story. Returning players can skip this to get right into the action.

Initially the player starts off with just his fists, which do minimal damage to enemies.

This helps encourage stealth as a full-frontal assault is not recommended with no equipment.



Game Progression(2)

After the first heist, “The Player” unlocks the shop, a sort of hub where items and weapons can be purchased from “The Dealer”. These weapons include guns and knives.

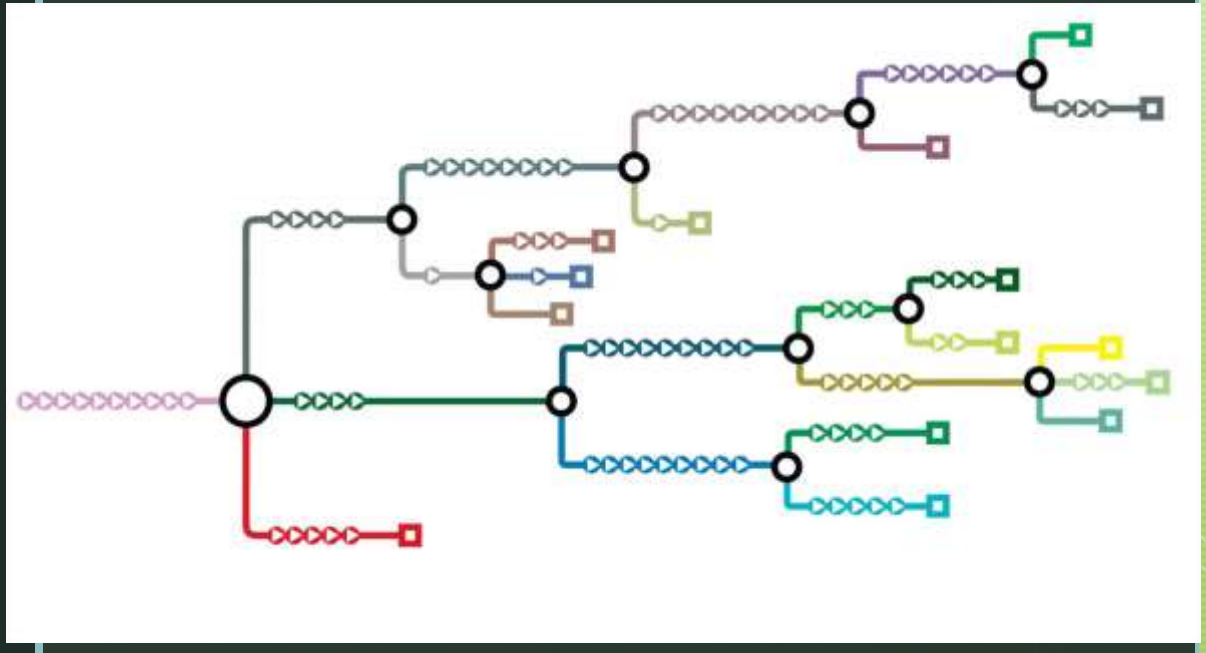
Other items include med-packs that can be used mid-session to restore some health.

Inspirations can be taken from games such as "Enter the Gungeon". Players can buy an item after walking up to it and pressing “E”.

Game Progression(3)

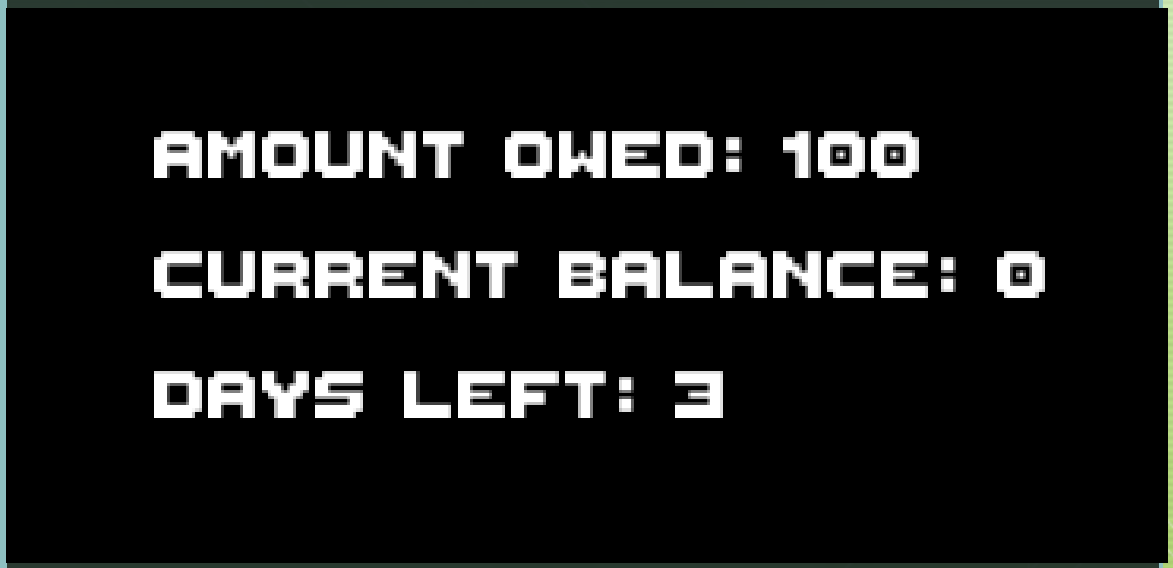
All of these are purchased with the money and collectibles stolen from the heists. These collectibles come in the form of gems, gold bars, coins, and cash that the player collects by walking over them during the heist.

These items and weapons make gameplay more varied as the number of approaches multiply exponentially. “The Player” must carefully balance buying upgrades to help him earn more money with the task of saving his money in-order-to pay back Starman.



Win Loss Conditions

- From day 1, the player is shown a dramatic screen showing how much money they have, how much is owed and how many days is left to pay it off.
- Every time they exit a stage or get defeated by the police; days will pass.
- Once the deadline comes, it checks if the player has enough money.
- If they have enough, they win! If they don't, it's game over.



Death/Victory Screens

Provide restart and exit buttons so players can decide if they want to play again or quit.

System Menus

While designing the various system menus in A.D.D., we do not want to distract the player with copious amounts of unnecessary menus.

Every menu must be as snappy and quick as possible.

We want the focus to be on the gameplay, to minimize the overall time spent in menus as much as possible so that our player gets as much entertainment by playing the game.

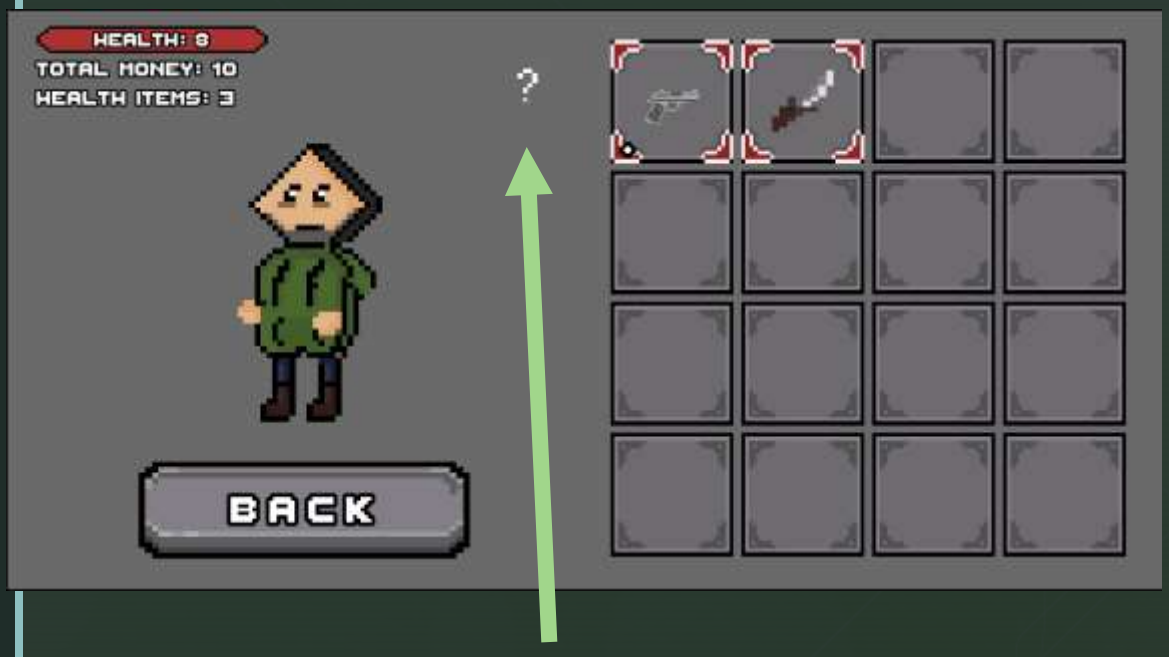
System Menus – Main Menu

In the main menu, buttons for “START”, “TUTORIAL”, “CREDITS” and “EXIT” are present against a cinematic illustration. We want to make this visually pleasing.

System Menus – Loadout Screen

There is a loadout menu in which you can equip/unequip your weapons. A 4 x 4 grid inventory is shown on the right with the player's avatar standing on the left. The "BACK" button is placed below the player and when it is pressed, it will send you back to the main game.

The inventory shows every weapon that you have in your arsenal and currently equipped weapons will be visually marked with a border. To equip/unequip an item, players simply click on them. Provide a help button in case any players don't know how to operate this screen.



Help button

System Menus – Map Selection Screen

In the map selection screen, players will see icons of various stages and shops that are available to them. By hovering over each icon, text will show up at the bottom of the screen, giving info about the stage such as how much loot is potentially available and potentially how dangerous it might be.

It is important to us that the player is aware of the challenge ahead. We want our players to know what they are getting themselves into when starting a new level.

Game Design Plan – Roles Needed

- Programmer
- Game tester
- Art Director
- Audio Director
- UI Director
- Level designer
- Marketing staff

Game Design Plan – Production(1)

- Total production time: 180 – 200 hours.
- Operations can be pipelined and carried out in parallel.
- 100-110 hours go into coding and the programmer can start straight away, using default textures and bodies to add fundamental features.
- Quickly make a test level to test various features.
- While the programmer is at work, art and music directors can start on getting assets. One by one, the programmer can drop in the textures/audio and set up animation states and audio managers.

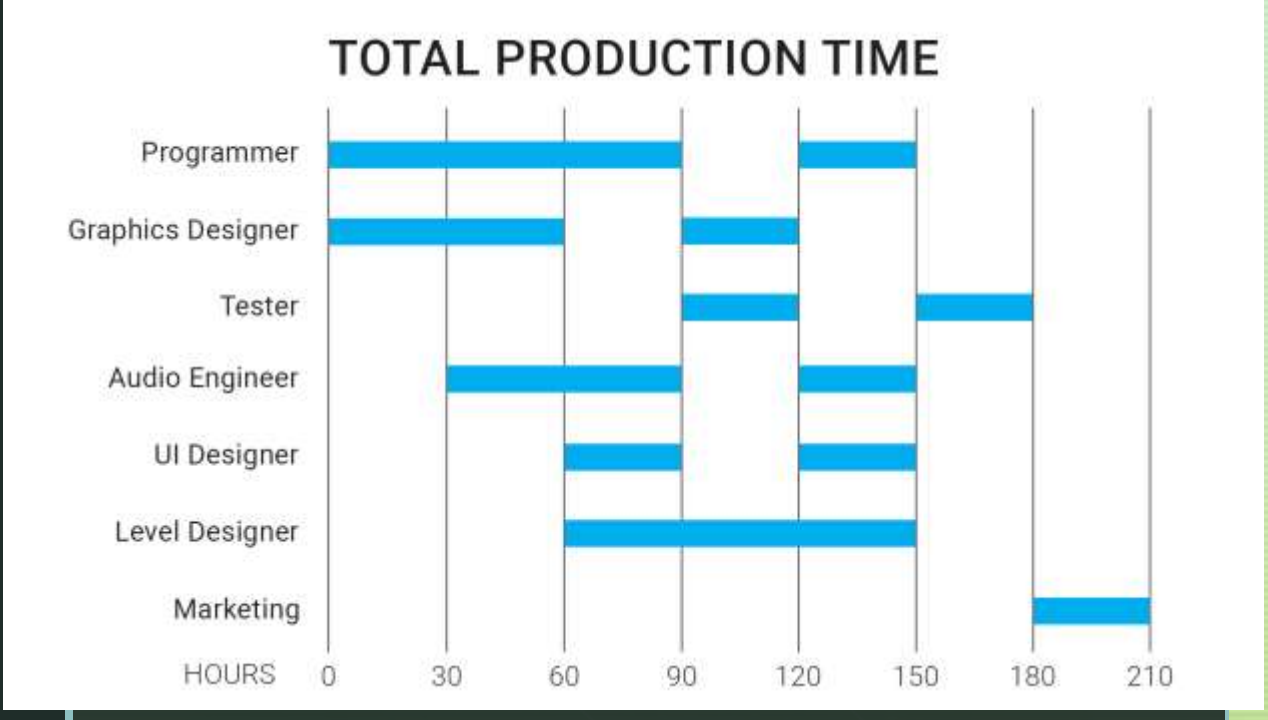
Game Design Plan – Production(2)

- For every feature that the programmer adds, they will rigorously put it through many simulations and find as many bugs as they can.
- For example, when creating characters with AI, they should be tested to their limits. Is the game stable no matter where they are placed? How about when 50 of them are loaded in a single level?
- UI Director can also start the design of every screen and icon. Communicate with the art and music director for assets. Talk to the programmer about how to link different screens together.
- Meanwhile, the level designer can plan out levels knowing what objects and game elements will be available.(Characters, loot, security camera)

Game Design Plan – Production(3)

- Once most of the features are done at around the 90-100 hour mark, level designer can work with the programmer in assembling levels.
- Towards the end, the marketing team will start to advertise and promote the game, knowing that the features are mostly finished. They will set a realistic release date for the development team to target.
- Afterwards, everyone will test, fix bugs, provide feedback and tweak various parameters in the game until the release date arrives.
- If working in the Godot game engine, export the game using the Windows Desktop preset. Fill in the necessary fields in the “Application” section including “Icon”, “Product Name”, etc.

Game Design Plan – Production(Visual Representation)



Postproduction

Postproduction is just as important as the production of the game as it is what will help spread awareness about our game.

Postproduction - Marketing(Trailers)

Firstly, a trailer should be created in order to raise excitement for the game. Time and effort should be taken in crafting the artwork for this short trailer as it is the first chance for audiences to be interested.

Postproduction - Marketing(Sponsorships)

In order to promote the completed A.D.D, free codes and a sponsorship would be provided to popular Twitch Streamers and Youtubers in order to raise awareness of the game.

The exposure it would create would really boost sales of the game and be an overall benefit to the A.D.D community. A large community improves the lifespan of the game and helps the possibility of a sequel.

Conclusion

- Completing A.D.D. is a vigorous process. It takes a lot of time and dedication to create all the assets, mechanics, ideas and everything else that is needed for a fully playable version of A.D.D.
- We believe that this idea when done correctly, can be popular, very fun to play and successful. Building this project from the ground up can be challenging but once all the mechanics of the game that are essential for the player are fully stable, developers can focus on level design and polish.
- This game is theoretically infinitely expandable as newer levels can be created whenever the ability to do so is there. This keeps the game fresh and keeps bringing the players that love the game back for more and potentially new players.
- Every step outlined in this document is essential to be completed correctly. For example, if our ideas for postproduction are not applied correctly this whole project has a bigger chance of never taking off and essentially failing.
- We believe that if this project is done properly, it can be profitable. For example, we can offer the ability to purchase our currency for money thus giving the player the ability to buy powerful items in the store faster.

References

The Github repository: https://github.com/TheCaister/At-Debts-Door-Document