### Technical Design Document Template

1.0 Revision History

<As you revise the document, list what was changed and when it was changed>

|  |  |
| --- | --- |
| Version | Description |
| 1.0 | Initial document |

2.0 Development Environment

2.1 Game Engine

Proprietary

2.2 IDE

Windows - Visual Studio 2022 Preview

Mac OS – Jetbrains CLion

2.3 Source Control procedures

All source is available on [GitHub](https://github.com/lcomstive/CoffeEngine)

2.4 Third Party Libraries

[*Box2D*](https://box2d.org/)

[*Raylib*](https://raylib.com/)

2.5 Other Software

3.0 Game Overview

3.1 Technical Goals

- 2D animated graphics

- Stable 60 FPS on moderate hardware  
(recent-ish CPU, dedicated GPU)

- Consistent AI

3.2 Game Objects and Logic

- Animals (child of animated sprite object)

- Behaviour tree determines movement and actions

Subclasses:

- Predatory Animal (either carnivore or omnivore)

Will hunt other animals for food

- Passive Animal (either omnivorore or herbivore)

Will avoid predatory animals and look for non-animal food sources

- Trees, rocks, and miscellaneous obstacles

- Pond (water source)

- Plants (food source for herbivores and omnivores)

- Carcasses (food source for carnivores. Replaces animals that have died)

3.3 Game Flow

User cannot interact with simulation.

4.0 Mechanics

<A list of the core game mechanics. I.e., what the player can do and how they achieve this, and what this triggers in the game. For example, shooting enemies is a core mechanic in an FPS>

5.0 Graphics

<Describe graphics features here. I.e., is your game top-down 2D? >

6.0 Artificial Intelligence

<Describe how AI works, i.e. state machine, fuzzy logic, GOAP. Describe the various behaviours and how they change behaviour, how do the ‘creatures’ in the game evaluate the world> <include diagrams/flowcharts showing decision making processes>

7.0 Physics

<if needed>

<What engine are you using, what features from it (spring? Colliders?) how will physics be handled for objects? (box or sphere collider for objects, capsule for player) need to record specific locations for any reason? Potential slowdowns and how to mitigate.>

8.0 Items

<List of items you can pick up that can affect the player, and what they will affect, like ‘picking up the hammer (refer collisions above) adds 5 to the players attack attribute’. Include details on how items influence gameplay or AI logic.>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Item | Parameter | Parameter | Parameter | Description |
| Default | 6 | 6 | 6 |  |
| Weapon | 5 | 7 | 7 |  |
| Weapon | 8 | 5 | 5 |  |
| Weapon | 5 | Possible 10 | NA | text |
| Weapon |  |  |  |  |

9.0 Game Flow

9.1 ‘Mission’ / ‘Level’ structure

<Are all levels stored in memory? what data is saved across levels, are levels loaded synchronously to prevent pauses?>

9.2 Objectives

<What does the player try to accomplish on each level/mission? How is the players progress evaluated?>

10.0 Levels

<If any of the Levels require specific behaviours, describe those here>

11.0 Interface

11.1 Menu

<What are the menu options and what do they do?>

11.2 Camera

<Describe the camera, how it moves, perspective/orthographic, can it switch? How? Does it need to render-to-texture? does it prevent itself going through walls, use flowcharts to document behaviour>

11.3 Controls

<Keyboard, tablet touch/swipe/tilt, joystick, mouse etc. record double taps, multi touch, use mouse smoothing/ scale mouse for aiming etc.>

14.0 Asset List

<List all files needed, along with known attributes >

16.0 Technical Risks

<if you want your game to be a 1000 player pvp battle royale with 4k 120fps graphics, you need to say if this is doable and how you intend to do it>