The Betel Driver

- **Playable Demo:** https://drive.google.com/drive/folders/1_pmjZ-g_2aUu0bBefhN3aPBVzqaQsihZ?usp=sharing
- **Demo video:** https://yuhao-xue-0410.squarespace.com/betel-driver
- **C# scripts samples:** https://drive.google.com/drive/folders/1oYqQVg8RgFLzQq_fznKXPy653c1hrg2d?usp=sharing

Introduction

This is a split-screen project. The left side features a third-person Traffic-Dodge gameplay where the truck weaves through traffic, while the right side uses a LocoRoco-like rolling and jumping mechanic inside the driver's mouth to control a betel-nut ball and maintain alertness.
I created this project after learning about the serious health risks of betel-nut chewing and the dangers of fatigue driving. My goal is to let players experience these issues directly through gameplay.
The game was fully built in Unity, and I completed all roles independently, including producer, artist, game designer, and programmer.

BackGround

Dangers of Fatigue Driving

- Fatigue driving significantly reduces a driver's alertness and reaction time, greatly increasing the risk of accidents. Studies show that fatigue makes drivers up to four times more likely to be involved in a crash or near-crash.
- When a driver is sleep-deprived or driving monotonously for long periods, their visual search ability and hazard perception drop sharply.
- In severe cases, the effects of fatigue mirror those of alcohol intoxication — a fatigued driver can behave similarly to someone driving under the influence.
- UK data indicates that fatigue may be a contributing factor in up to 20% of all fatal or serious road accidents every year.

Dangers of Chewing Betel Nut (Areca Nut)

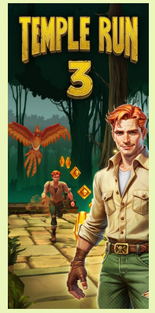
- Even without tobacco, betel nut (areca nut) is classified as a carcinogenic substance, strongly associated with oral submucous fibrosis and oral cancer.
- Long-term chewing can damage multiple body systems, including the endocrine, immune, and reproductive systems. It has been linked to conditions such as hypothyroidism, prostate enlargement, and infertility.
- Betel-nut chewing is addictive and reinforced by social and cultural factors, making cessation difficult and posing serious public-health risks.
- In terms of oral health, betel-nut chewing can cause tooth discoloration, gum irritation, and severe dental damage.

Connection to This Game(Areca Nut)

- The protagonist of this game is a truck driver forced to work overtime under heavy financial pressure. Long-term exhaustion traps him in an extremely dangerous state of fatigue driving. To barely stay awake, he relies on chewing betel nut as stimulation. This creates a vicious cycle in which he continues working while rapidly destroying his own body.

Reference

For the left screen, I looked at games like Subway Surfers, Temple Run, Traffic Rider, Lane Rush, and Lane Splitter. These titles use lane-switching and quick reactions as their core mechanics.
The right screen is inspired by the classic LocoRoco, where tilting the stage moves and bounces a soft-body character.



Innovation Section

Continuous Steering System

Unlike games like Temple Run, which use fixed lane switching and limit movement to 3–5 preset lanes, my driving system operates in a fully continuous space. The player can steer to any position on the road and make fine adjustments, creating a driving experience that feels much closer to controlling a real vehicle.

Jaw-Driven Bounce Physics

On the right screen, I expanded the traditional LocoRoco-style mechanic. The betel-nut ball does not jump through a simple button press; its upward force is determined by jaw lift speed, acceleration, and displacement. I wrote a custom physics formula to produce a bounce motion that better matches real chewing.
I also simulated friction between the ball and different teeth, as well as various tooth interactions—for example, loose teeth that wobble, metal teeth with lower surface friction, and sensitive teeth that react differently on contact.

Dual-Screen Alertness Link

Alertness connects the two gameplay systems. When alertness enters different ranges, the screens begin to influence each other—for instance, steering the truck may tilt the mouth, or the vehicle may become temporarily unresponsive, forcing the player to regain alertness through the right-side gameplay. This system unifies the two screens into a single cohesive experience rather than separate modules.

GameMechanics

Split-Screen Structuret (Areca Nut)

The game runs on two simultaneous screens: driving gameplay on the left and mouth-physics gameplay on the right.

Left Screen: Driving Mechanicst (Areca Nut)

2.1Basic Controls

Players use the Z / X keys to move the vehicle left and right.

2.2Dodging & Collision

Colliding with other vehicles reduces HP; reaching zero results in failure.

2.3 Item System

Items will spawn on the road, and the player can pick them up by driving into them.
Coins
Magnet — pulls nearby coins toward the vehicle
Shield — grants temporary invincibility to the vehicle

2.4 Alertness Effect on Driving

As alertness decreases, input delay increases, steering becomes sluggish, and the vehicle becomes harder to control.

2.5 Procedural Generation

The game procedurally generates the road, roadside scenery, traffic vehicles, and item placements.

Right Screen: Mouth Physics Gameplay

3.1 Basic Controls

Left / Right Keys: tilt the mouth to roll the betel-nut ball.
The ball will not roll unless the tilt angle exceeds the minimum tilt angle required for rolling.
Up Key: lift the lower jaw up to the "jaw-lift limit" for that chapter.
The rising jaw pushes the ball upward. After the jaw reaches its upper limit, it begins to fall.During the falling phase (before the jaw fully resets), the player can press Up again to lift it back up to the jaw-lift limit.

3.2 Bounce Physics

The height of the ball's jump depends on the ball's speed at the moment it leaves the jaw.
This speed equals the jaw's speed when it reaches the jaw-lift limit.
The jaw-lifting process is an accelerated movement, so the final speed is determined by:
jaw-lift acceleration, jaw-lift displacement
Summary:
The ball's jump height is positively correlated with jaw-lift acceleration and jaw-lift displacement.

3.3 Betel-Nut Ball Count

When the ball falls out of the mouth, the player loses one ball from the inventory, and a new ball respawns inside the mouth.
When all balls in the inventory are used up, the mouth will no longer spawn new balls.

Oral Element System

4.1 Normal Tooth
Standard friction.

4.2 Metal Tooth
Lower friction, making it easier for the ball to start rolling.

4.3 Sensitive Tooth
High-speed impacts from the betel-nut ball trigger pain effects.

4.4 Loose Tooth
Falls to one side when the mouth tilts past a certain angle, creating a new slope.

4.5 Cancerous Tissue
Hitting it triggers a stronger pain reaction.

4.6 Food Debris
Sticks to the ball → increases surface friction → requires a larger slope before the ball can roll again.

4.7 Bleeding Wound
Blood Jet: When the ball touches the blood jet, it gets pushed upward.
Wound Base: Hitting the wound base causes the left screen to flash red and the mouth model to shake. These effects have a 5-second cooldown and won't trigger repeatedly.
When the player presses the Up key, the wound base shoots out a blood jet.

4.8 Hole
When the betel nut ball falls into the hole, the available betel nut count decreases by one. After that, a new betel nut ball spawns at the center of the mouth.

Stimulation Node System

Stimulation nodes only refresh at several predetermined positions inside the mouth.
Each chapter allows a fixed number of nodes to exist at the same time.

Alertness System (Core Link Between Screens)

Alertness Changes

Decreases over time.
Increases when the betel-nut ball hits stimulation nodes.

Alertness Threshold Effects

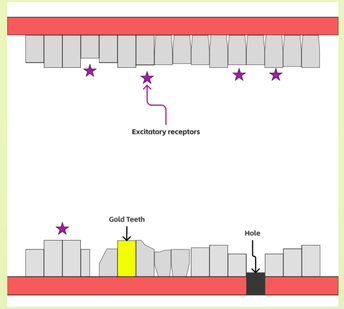
High Alertness
No delay; both screens respond normally.
Medium Alertness
Input delay on driving.
Turning the truck left or right tilts the mouth in the same direction.
The player can simultaneously control the mouth to counteract the tilt caused by the truck's turning.
Low Alertness
Player vehicles cannot turn left or right.
Input delay on tilting.
The mouth continues to tilt for a short distance after the tilt key is released.
Player must restore alertness via the right screen, stimulation nodes.

Difficulty Scaling by Chapters

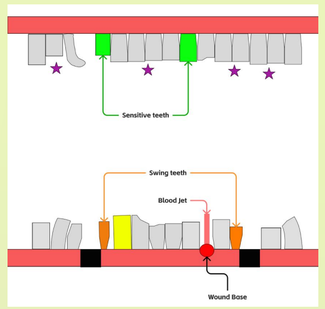
As chapters progress, the oral environment deteriorates and supporting teeth decrease.
More missing teeth, sensitive teeth, cancerous tissue, and debris appear.
Ball control becomes harder → alertness harder to restore → difficulty increases sharply.

Upgrade System

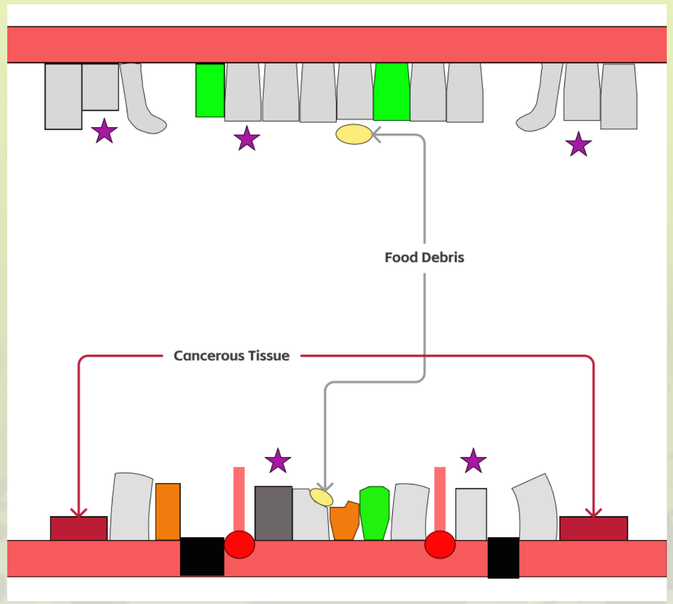
Players can spend the coins collected in each level to increase the truck's maximum HP and the maximum number of betel-nut balls they can carry.



LEVEL1

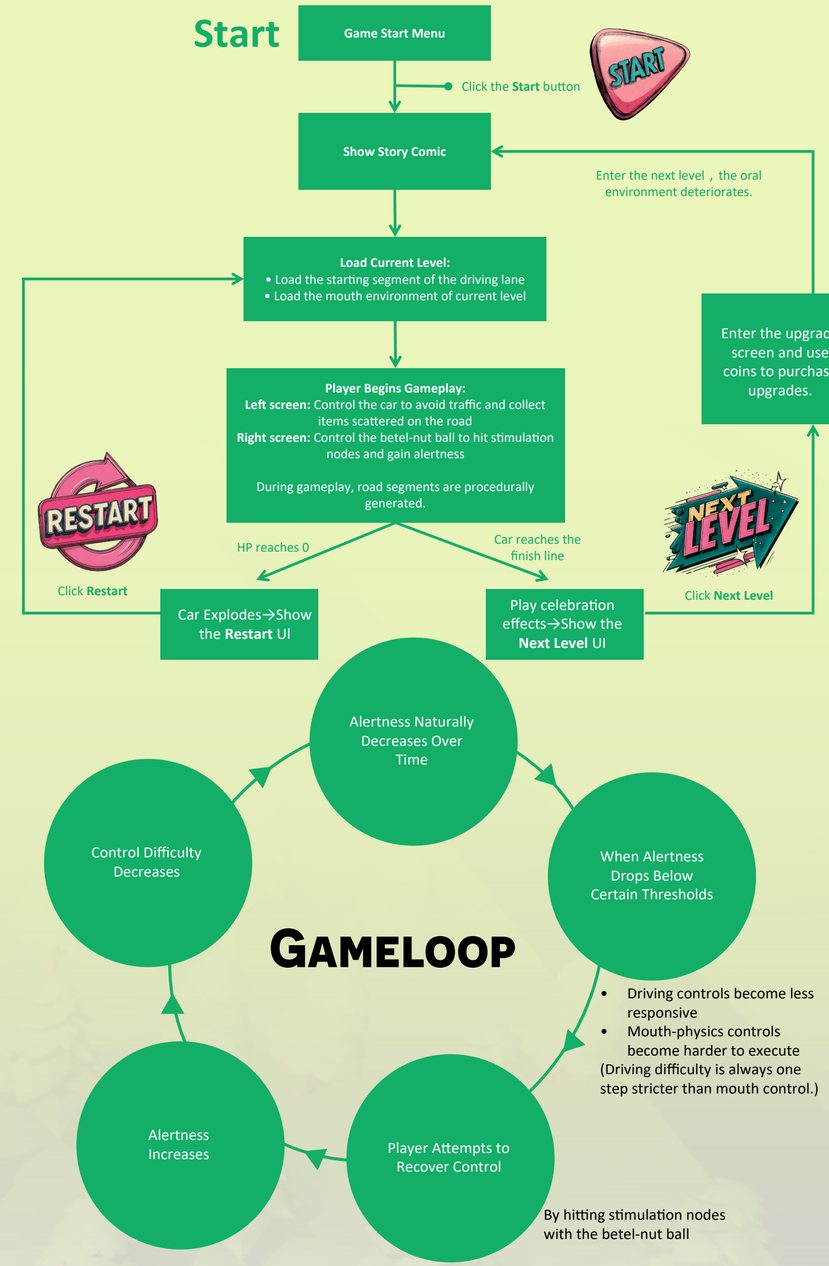


LEVEL2



LEVEL3

GameFlow



The game forces the player into the same cycle as a real fatigued driver: relying on stimulation to stay in control, only for the effect to fade and the struggle to restart.

MDA Framework (Dynamics)

Split Attention

Players must divide attention between driving and controlling the mouth, rapidly switching focus between two simultaneous tasks.

Alertness Tug-of-War

Alertness continuously decreases over time, forcing players to periodically focus on the right screen to hit stimulation nodes. This creates a constant tension between “driving safely” and “sacrificing attention to recover alertness.”

Risk–Reward Decisions

To hit stimulation nodes, players must move the betel-nut ball into riskier areas (missing teeth, loose teeth, wound zones, or active blood jets). Each attempt may restore alertness—or cause the ball to fall, trigger pain feedback, or make control harder.

Chapter-Based Escalation

As the game progresses, safe teeth become fewer, diseased areas increase, and stimulation nodes become harder to reach, gradually raising the difficulty and pressure.

Item-Driven Behavior Shift

Players are incentivized to chase spawned items on the road. These items provide temporary invincibility, auto-collection benefits, or currency that can later be used to upgrade max HP and ball capacity—affecting both routing and decision-making.

Aesthetics

Tension & Stress

Constantly juggling two tasks creates continuous mental load, simulating the exhaustion and pressure of working under fatigue.

Horror & Challenge

The oral environment progressively deteriorates across chapters—loose teeth, sensitivity, cancerous tissue, and bleeding wounds—making the player feel the character’ s physical decline alongside rising difficulty.

Hope

The upgrade system provides a small but meaningful sense of accomplishment:
Higher HP capacity
Increased ball inventory
Players feel: “I’ m struggling, but I’ m improving.”

Relief & Temporary Safety

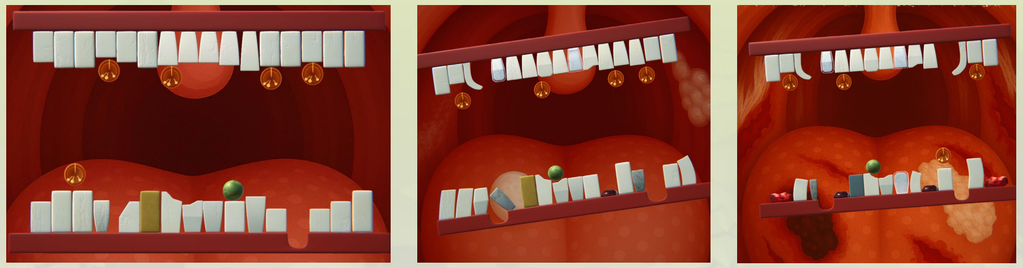
The shields and magnets provided players with a brief respite and a small surge in gold rewards.

Story Desgin

A long-haul truck driver works beyond his physical limits due to financial stress and becomes dependent on chewing betel nut to avoid falling asleep at the wheel. Over time, the habit severely damages his health, leading to early-stage oral cancer — yet despite the diagnosis, life pressures force him to keep driving.

Game 2D Art

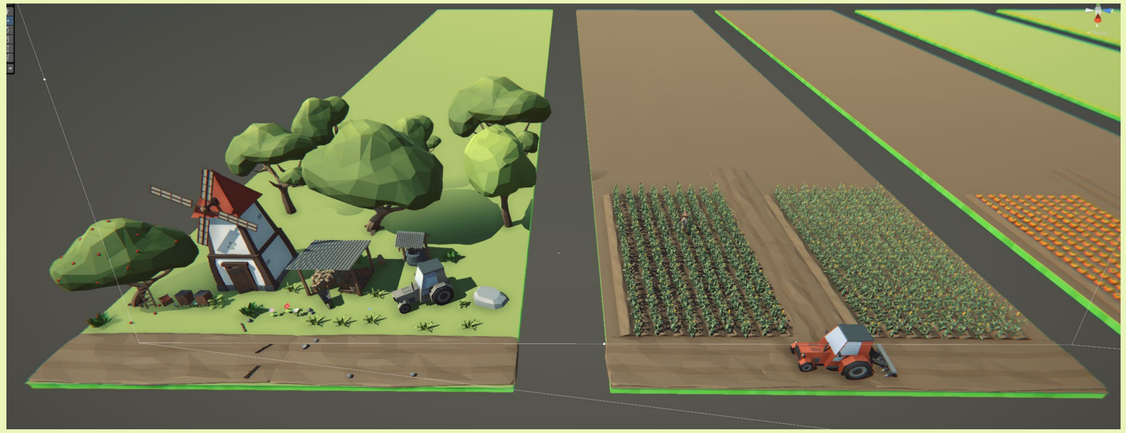
WAKEFULNESS ROUTE

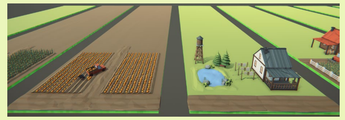


Level1 Level2 Level3

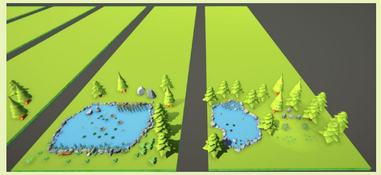
game 2D Art

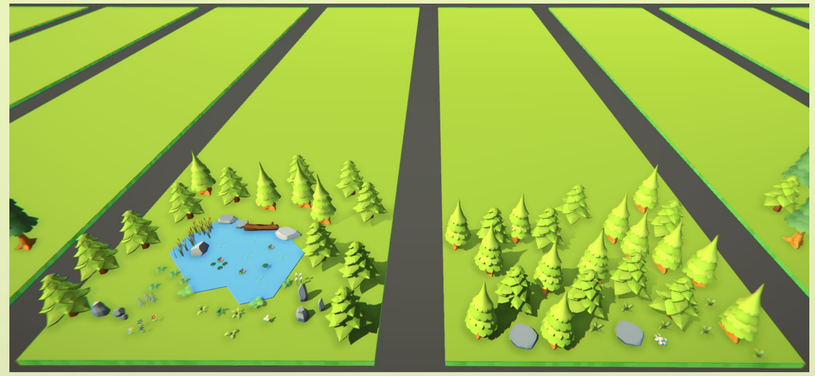
Farmprefabs

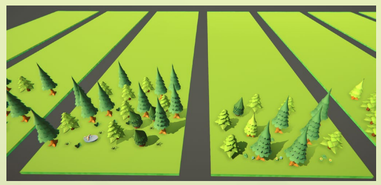


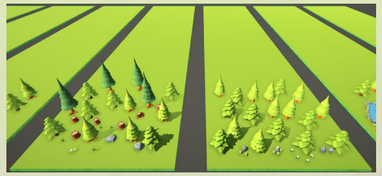


ForestPrefabs

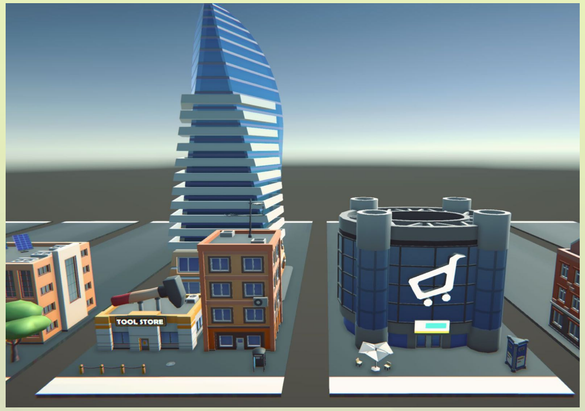


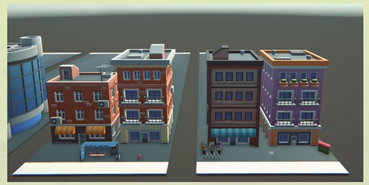


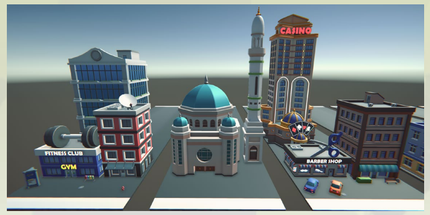


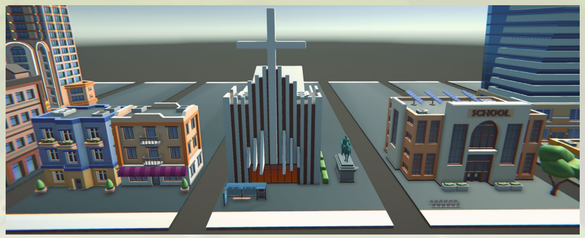


CityPrefab

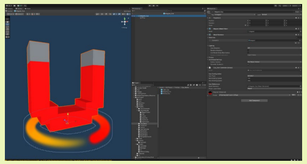




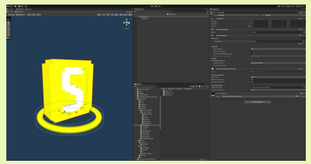




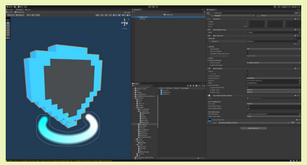
Unity Inspector



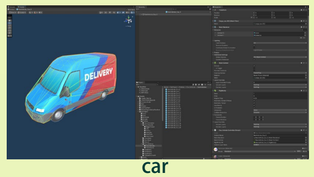
Magnet



Coin

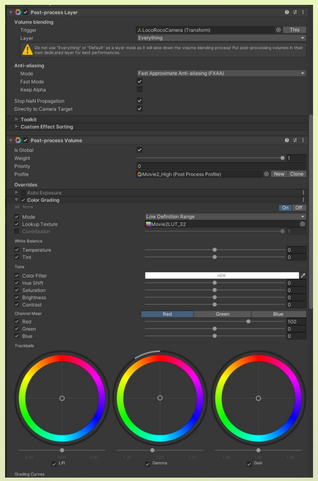


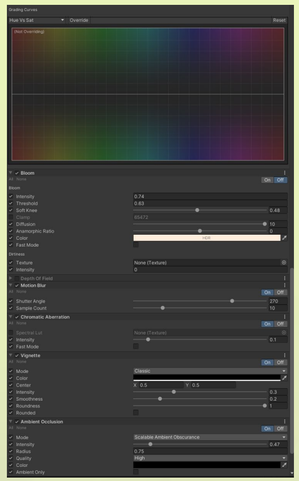
Shield



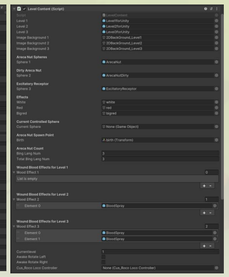
car

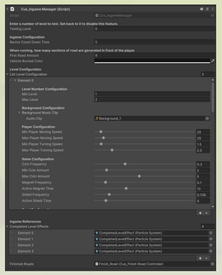
POST PROCESS



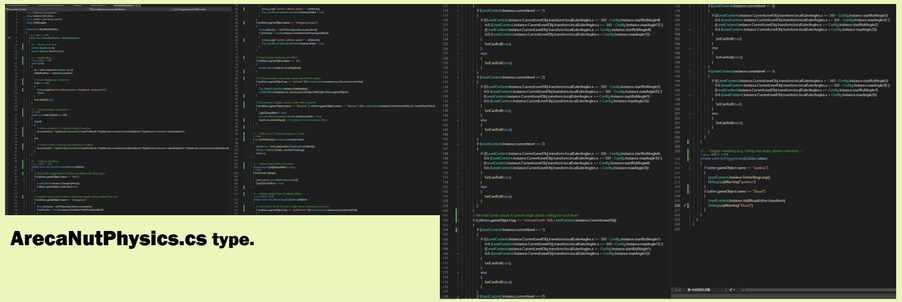


Unity Inspector（Scripts Inspector）





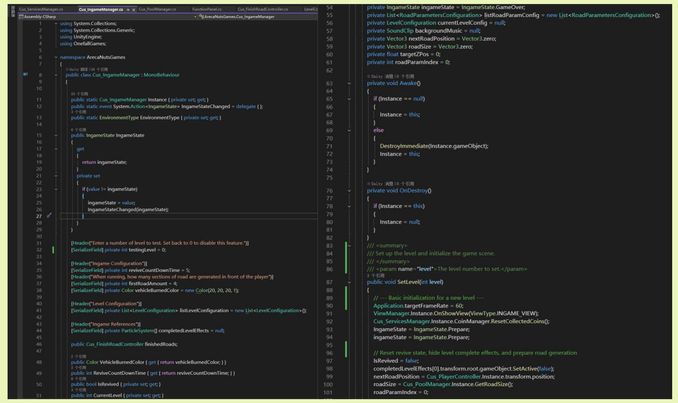
Game code



ArecaNutPhysics.cs type.

Betel Nut Core Physics Script

Handles all collision logic with teeth, wounds, cancer tissue, and excitatory items. Dynamically controls betel-nut rolling constraints based on mouth tilt angle and tooth type.

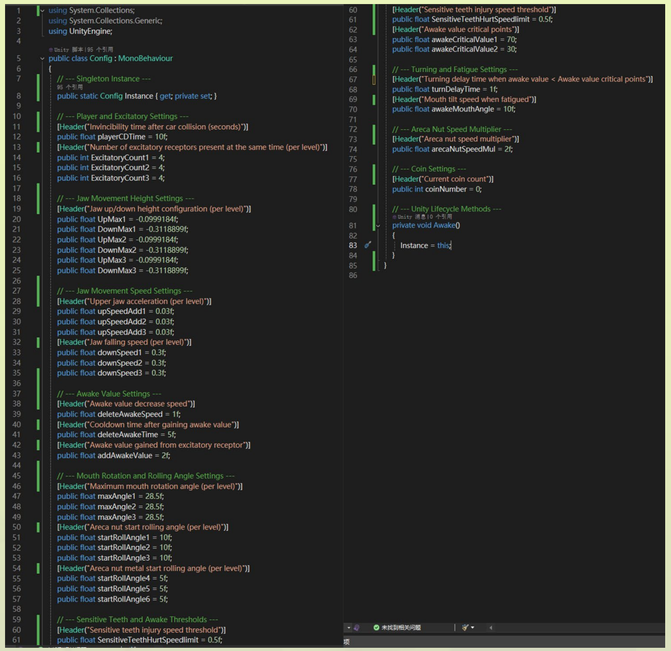


Cus_IngameManager.cs

Racing Game Core Manager (Singleton)

Acts as the main gameplay controller, handling level initialization, state machine management (Prepare / Playing / Revive / CompletedLevel / GameOver), and broadcasting state events.

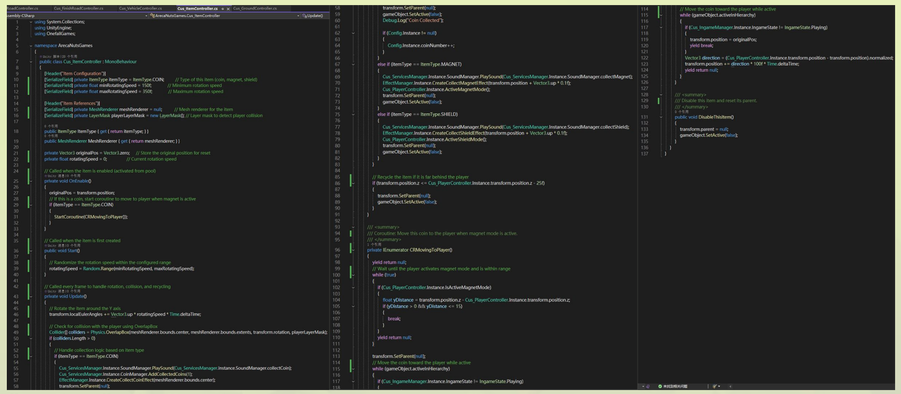
SetLevel() fetches configuration values, randomizes road count, player speed, and vehicle density, and generates road segments and finish lines via the object pool. It also controls BGM, completion effects, and revive flow.



Config.cs

Global Configuration Singleton

Centralized manager for all gameplay parameters (excitatory item count, mouth rotation angle, wakefulness thresholds, betel-nut speed multipliers, etc.). All values are exposed in the Inspector for convenient difficulty tuning across levels.



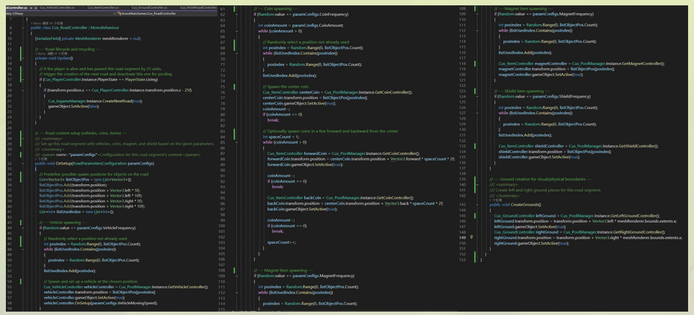
Cus_ItemController.cs

Item Controller (Coins / Magnet / Shield)

Handles item rotation and collision with the player. Upon collection, plays SFX/VFX and triggers logic based on item type—updating Config.coinNumber, activating magnet mode, or enabling shield mode. In magnet mode, coins automatically fly toward the player via a coroutine. Items are recycled when falling 25 units behind the player.

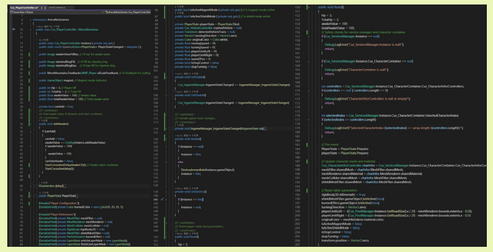
Cus_LocoRocoCameraController.cs

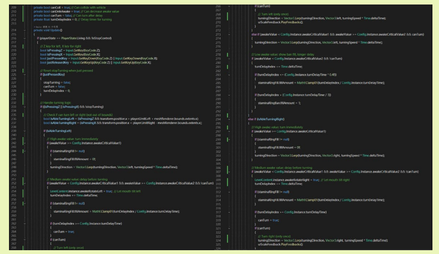
LocoRoco Camera Controller (Singleton)
Controls right-screen camera shake effects. When shake is enabled, the camera performs random positional jitter for 2 seconds (Random.insideUnitSphere * swing) while triggering flashing visual feedback via LevelContent. After shaking ends, the camera resets to its original position.

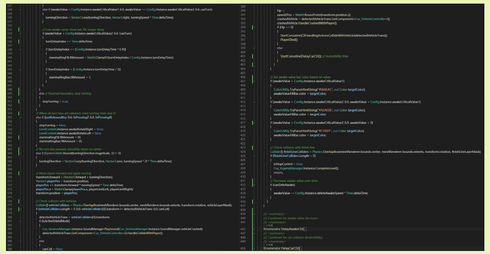


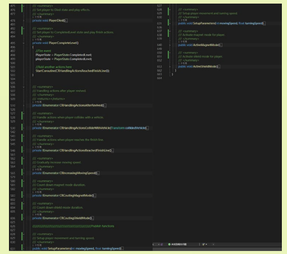
Cus_RoadController.cs

Road Segment Controller
During OnSetup(), uses RoadParametersConfiguration to spawn random objects (vehicles, coin chains, magnets, shields) across five road positions (Center, Left-5/10, Right-5/10). A position index list prevents overlaps. Once the player passes the segment, it triggers Cus_IngameManager.CreateNextRoad(true) to spawn the next road, and recycles itself and its side decorations.









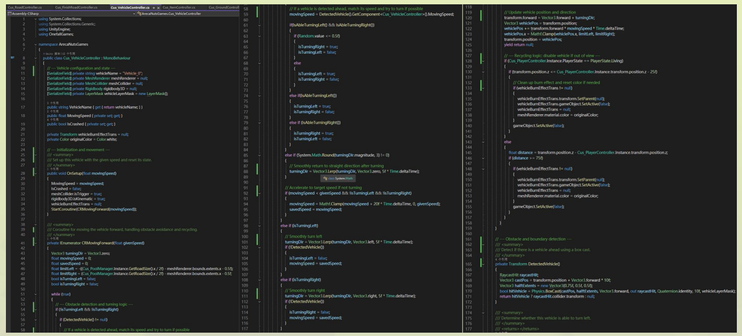
Cus_PlayerController.cs

Racing Player Core Controller (Singleton)
Manages the player state machine (Prepare / Living / Died / CompletedLevel), steering with Z/X keys, HP system, and magnet/shield behavior. Implements an innovative Wakefulness System:

When wakefulness drops below a threshold, steering input is delayed (turnDelayTime), and the right-side LocoRoco mouth tilts (awakeRotateLeft/Right) to simulate fatigue.

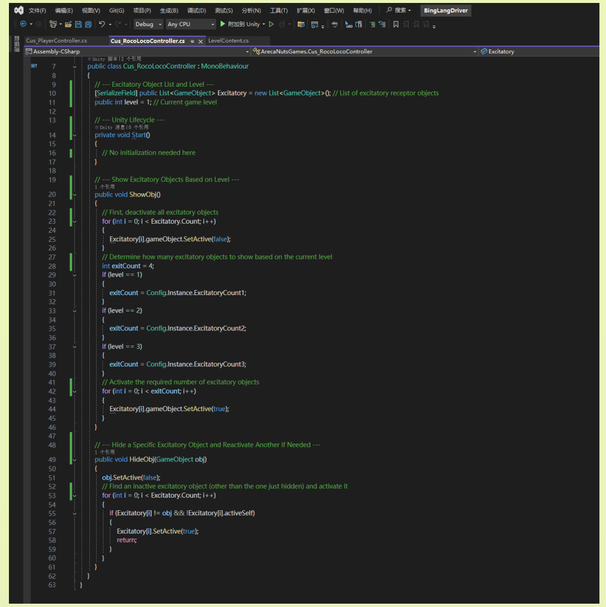
Collecting excitatory items restores wakefulness; it gradually decays over time.

Also handles collision reactions, invincibility frames, revive logic, and updates UI elements including progress bar color (Green/Yellow/Purple) and stamina ring.



Cus_VehicleController.cs

Vehicle Controller
Automates vehicle movement, avoidance logic, and collision handling. CRMovingForward() uses BoxCast to detect traffic and randomly steer left/right when blocked, matching the speed of the forward vehicle until safe. Collision response depends on player state:
With Shield → HandleCollideWithPlayer() applies stronger impact (15f)
Without Shield → HandleCrashedWithPlayer() applies smaller force (7f)
Vehicles then activate physics, play fire effects, change material tint, and stop movement.



Cus_RocoLocoController.cs
Excitatory Item Manager (LocoRoco Side)
Attached to three level objects. Based on the current level (1/2/3), it activates a configured number of excitatory items from Config. When an item is collected, HideObj() disables it and randomly re-activates another, ensuring the total number remains constant.