Universidad Nacional de San Agustin de Arequipa

Facultad de Ingeniería de Producción y Servicios

Escuela Profesional de Ingeniería de Sistemas

DOCUMENTACIÓN DEL PROYECTO

Videojuego Educativo de Carreras en Unity

Curso: Tecnología de Objetos

Profesor: Ing. Richart Smith Escobedo Quispe

Integrantes del Equipo

Choquehuanca Zapana Hernan Andy (Rama: hz07s/workspace)
Larico Rodriguez Bryan Fernando (Rama: BryanLarico/workspace)
Portugal Portugal Eduardo Sebastian (Rama: Eduardo-P/workspace)
Mamani Anahua Victor Narciso (Rama: victorma18/workspace)

Semana Documentada: Semana 2

Fecha: 22 de noviembre de 2025

1. Introducción

Este documento describe la arquitectura técnica del sistema de control del vehículo, estados, cámara, sistema de eventos y elementos adicionales del proyecto de videojuego educativo en Unity. Se presenta un desglose formal de cada clase, responsabilidades principales, métodos relevantes y patrones de diseño aplicados.

El formato en dos columnas permite una mejor distribución de contenido técnico y reduce espacios innecesarios, manteniendo un estilo profesional.

2. CarController.cs

2.1. Rol General

Clase principal encargada de gestionar la física, movimiento, aceleración, frenado, dirección y transición entre estados del vehículo.

2.2. Patrón Aplicado

Utiliza el **Patrón State** para cambiar dinámicamente el comportamiento según:

[noitemsep]

- Estado normal
- Estado acelerado (*boost*)
- Estado penalizado

2.3. Responsabilidades

[noitemsep]

- Control de velocidad, frenado, dirección.
- Aplicación de fuerzas físicas.
- Administra estados del vehículo.
- Actualiza posición visual de las ruedas.
- Controla colisiones del vehículo.

2.4. Métodos Principales

InitializeComponents(): Inicializa rigidbody, físicas y fricción.
InitializeState(): Inicia el estado normal por defecto.
Update(): Recibe input y ejecuta lógica del estado.
FixedUpdate(): Aplica fuerzas físicas del estado.
ChangeState(): Cambia entre Normal, Boosted y Slow.
ApplyAcceleration(): Aplica torque a ruedas traseras.
ApplySteering(): Controla ruedas delanteras.
ApplyBraking(): Aplica freno a todas las ruedas.
ApplyNormal/Boost/SlowPhysics(): Ajusta drag y física.
HandleCollision(): Reduce velocidad en colisiones fuertes.

3. CameraController.cs

3.1. Rol General

Sigue al vehículo suavemente usando interpolación y ajustes de rotación.

3.2. Características

[noitemsep]

- Sistema de *follow camera*.
- Movimiento suave con Lerp.
- Rotación progresiva con Slerp.

3.3. Métodos Principales

LateUpdate(): Calcula posición deseada y aplica suavizado.
SetTarget(): Asigna un nuevo transform objetivo.

4. IVehicleState.cs

4.1. Rol General

Interfaz que define las funciones mínimas para cualquier estado del vehículo.

4.2. Métodos Definidos

[noitemsep]

- EnterState()
- UpdateState()
- FixedUpdateState()
- ExitState()
- HandleInput()

- OnCollisionEnter()

Sirve como contrato para que todos los estados mantengan un comportamiento consistente.

5. NormalState.cs

5.1. Descripción

Estado base del vehículo. Usa la aceleración y velocidad normales.

5.2. Características

[noitemsep]

- Física estándar.
- Dirección sensible.
- Frenado normal.

5.3. Funciones Relevantes

EnterState(): Activa física normal.
HandleInput(): Procesa input de aceleración, giro y freno.
OnCollisionEnter(): Reduce velocidad si el impacto es fuerte.

6. BoostedState.cs

6.1. Descripción

Estado activado al responder correctamente una pregunta.

6.2. Efectos

[noitemsep]

- Aumenta aceleración.
- Reduce drag.
- Multiplica input del jugador.

6.3. Funciones Relevantes

UpdateState(): Controla duración del boost.
ExitState(): Retorna la física a estado normal.

7. SlowState.cs

7.1. Descripción

Estado penalizado al fallar una pregunta.

7.2. Efectos

[noitemsep]

- Aceleración reducida.
- Mayor drag.
- Control más pesado.

7.3. Funciones Relevantes

UpdateState(): Maneja temporizador de penalización.
ExitState(): Restaura física normal.

8. EventBus.cs

8.1. Rol General

Sistema de mensajería global bajo patrón **Observer**. Permite comunicación desacoplada entre sistemas.

8.2. Funciones Principales

Subscribe(): Registrar escuchadores.
Unsubscribe(): Remover escuchadores.
Publish(): Distribuye eventos a los suscriptores.

8.3. Eventos Implementados

[noitemsep]

- **VehicleBoostEvent**
- **VehiclePenaltyEvent**

9. MenuPrincipal.cs

9.1. Rol General

Controla el menú de inicio del videojuego.

9.2. Funciones

IniciarJuego(): Carga la escena principal.
SalirJuego(): Cierra la aplicación.