CORE

Core Game Engine

One Engine. Infinite Games.

By SNiP

August 4, 2025

What is CoreGameEngine?

- A modular, fully on-chain game engine for autonomous, permissionless games.
- Developers deploy isolated contracts for game logic, assets, player identity, and permissions.
- Clean UI guides devs through deploying 12+ lightweight modules — no custom infra required.
- Built for rapid iteration, deep composability, and full-chain sovereignty.
- Optimized for Core chain — complete game setup costs under $1.

CORE

The Problem

- On-chain game development is fragmented, repetitive, and slow.
- Developers rebuild identity systems, asset logic, and permission layers from scratch.
- There's no modular stack for rapidly launching fully on-chain games.
- Most tooling focuses on assets or frontends — not core logic or infrastructure.

CORE

The CoreGameEngine Solution

- **Factory Pattern:** Deploy isolated contracts per game.
- **Modular System:** Identity (ERC721), Assets (ERC1155), Game Logic.
- **Developer Friendly:** Compose or extend features via interfaces.
- **Infra Focused:** Your game, your rules, maximum speed.

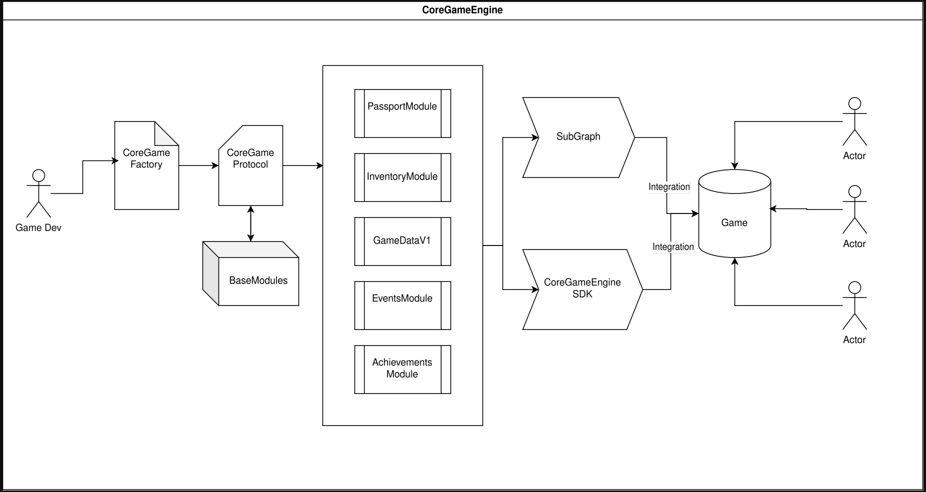
CORE

Where CoreGameEngine Fits

Feature	CoreGameEngine	MUD	Dojo (Starknet)
Infra-first	✓	✓	✓
Modular Assets	✓	✗(via plugins)	✗
Per-game Isolation	✓	✗	✗
EVM Compatible	✓	✓	✗(Cairo)
Dev Experience	CLI + Factory	Hardcoded SoA	Complex ECS
Customizable Logic	✓	● (codegen)	● (complex)

CORE

Architecture Overview



Key Modules

- **CoreGameFactory** – Deploys isolated CoreGameProtocol instances.
- **CoreGameProtocol** – Manages game state, access roles, and connected modules.
- **UserPassport (ERC721)** – Unique game-linked identity and roles.
- **InventoryModule (ERC1155)** – In-game items, currencies, and collectibles.

CORE

Why CoreGameEngine Wins

- Unified infra-first approach – game logic and assets managed together.
- Developer-first – intuitive factory system and modular design.
- Future-ready – plug into existing EVM ecosystems or extend easily.
- Scalable – isolate games, players, and assets per deployment.

CORE

Live Demo

- Game deployed via Factory with single click
- Player mints Passport (ERC721 identity)
- Inventory items minted and consumed on-chain
- Events emitted + subgraph listening (optional)

CORE

Use Cases

- Tap-to-play games (Flappy, runners, idle games)
- Turn-based games (card, tactics, asynchronous PvP)
- Open-world modular MMORPGs (custom assets, roles, quests)
- Arcade + leaderboard templates for events

CORE

Roadmap

- Factory + Protocol Contracts
- Inventory & Passport Modules
- Game templates + CLI for quickstart
- Creator dashboard (deploy + monitor games)
- Player Profiles + Leaderboards
- Optional Subgraph / Off-chain Cache Tools

CORE

Why Now? Why CoreDAO?

- On-chain games are evolving from experiments to ecosystems.
- CoreDAO is early in GameFi — this is the moment to shape standards.
- CoreGameEngine lowers barrier for builders + scales with infra.
- Aligned with CoreDAO's values: composability, speed, decentralization.

CORE

Team & Background

- Builders: Rishabh (SNiP), Krishna (Kling), Kittu (Anon930)
- Anon930: Marketing, ideation and product development
- Kling: Founder of String Metaverse, India's premier and only publicly listed Web3 company.
- SNiP: Chiliz Chain Winner, Mina Protocol Mentions,

CORE

Get Involved

- Try the prototype: `https://coreengine.site`
- GitHub: `https://github.com/XxSNiPxX/CoreGameProtocol`
- Telegram: @heysnip
- Looking for: feedback, testers, game devs, contributors

CORE