Mini Project

Base Paper

GameScript: A Simplified Scripting Language for Video Game Development

---

1. What the Base Paper Is About (GameScript)

The base paper introduces GameScript, a simplified scripting language used to define behavior in games.
The main idea of GameScript is not power, but simplicity and control.

Key ideas from the paper:

- Everything is treated as an agent
- Each agent has state
- Behavior is defined using simple if–then rules
- No loops inside the language (logic runs every frame)
- Mostly numeric variables and simple conditions

The paper shows that even with these restrictions, complex behaviors can still be created.
The focus is on clear behavior definition rather than flexibility or performance.

---

2. Problem We Are Solving

In web development, animations are usually written using CSS keyframes.

CSS animations are powerful, but they come with practical issues:

- Hard to visualize timing when many elements are involved
- Managing delays manually is messy
- Coordinating multiple elements is error-prone
- Debugging animation behavior is difficult
- Animation logic and timing are tightly coupled with CSS code

For simple animations, CSS works well.
For complex animation sequences, it becomes difficult to manage and maintain.

---

3. What Our Project Does

Our project is a web-based animation tool.

It allows developers to:

- Visually create animations using a timeline
- Preview animations in real time
- Export clean CSS animation code

Instead of writing CSS keyframes manually, developers use a visual interface.

The final output is still standard CSS that can be used directly in any website.

---

4. Where GameScript Fits In

GameScript is not used instead of CSS.

Users do not write GameScript code.

GameScript ideas are used internally to manage animation logic.

Inside the tool:

- Each animated element is treated as an agent
- Each agent has states such as:
    - waiting
    - running
    - completed
- Every frame, simple rules decide how the animation progresses

Conceptually, the logic works like this:

- If the current time is before the start time, nothing happens
- If the animation is active, properties are updated
- If the animation is finished, it is marked as complete

This approach is directly inspired by the GameScript execution model described in the base paper.

---

5. Why We Use GameScript

We did not need GameScript to build this tool.

We chose GameScript because:

- It is the base paper for the project
- Its agent-based, state-driven rule model fits animation sequencing naturally
- Animations are inherently time-based and state-based

The goal of the project is not to prove that GameScript is better than CSS, but to apply the paper's model in a different domain.

---

6. Technologies Used

Languages:

- HTML for structure
- CSS for final animation output
- JavaScript for logic, timeline handling, and preview

Concepts:

- Frame-based updates using requestAnimationFrame
- State machines
- Rule-based logic inspired by GameScript

No backend is required.

---

7. Novelty of the Project

- GameScript was originally proposed for video games
- This project applies its core ideas to web animations
- Uses agent-based state logic to manage animation sequences
- Generates real, usable CSS animation code

This makes the project a practical extension of the base paper rather than a direct replication.

---

8. Final Summary

We are building a visual tool to create animations that exports standard CSS code.
GameScript is used only as an internal logic model to manage animation timing and behaviour.