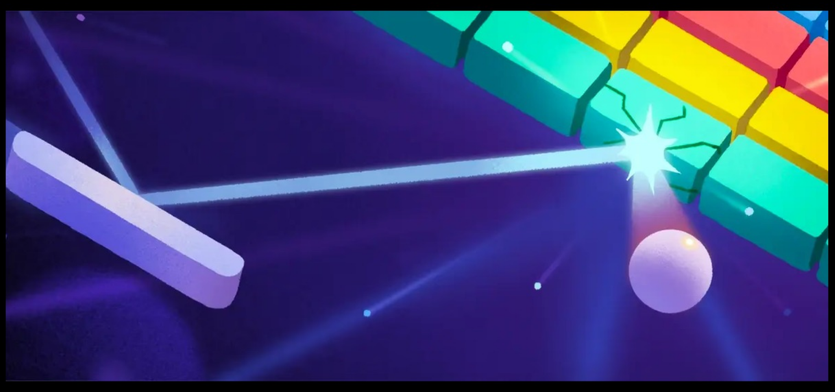
# BrickBreaker



**A retro arcade game, reinvented.**
**Built from scratch in x86 Assembly with a modern, minimalist approach.**

### Overview.

Classic brick-breaking fun powered by direct memory manipulation and custom VGA rendering.

### Objective.

Demonstrate system-level mastery by hand-coding a full game engine in Assembly.

### Tools.

MASM assembler, DOSBox, and custom SVG assets for visual flair.

### System architecture.

Direct VGA mode access, keyboard ISR, and low-level memory optimization.

### Methodology.

Stepwise programming: bootstrapping, draw loop, input handling, and rendering.

#### Challenges & Process

Implementing precise collision detection and frame-perfect movement involved creative bitwise logic and careful testing for stability.

Overcoming assembly's verbosity required macros, modular design, and relentless debugging in a noisy, low-level environment.

Custom keyboard interrupt handling allowed hyper-responsive input. Each step advanced both understanding and execution speed.

#### Results, Learnings, and Impact

Achieved a smooth-playing, nostalgic arcade experience in pure Assembly. Frame rate and precision matched commercial retro games.

Gained new insight into how graphics, input, and physics interact at the machine level. Direct manipulation of memory and hardware was both empowering and humbling.

Project strengthened practical x86 and debugging ability, and validated self-taught, incremental low-level programming methodologies.