Building an HTML5 Game Engine

Andrew Brough & Bethany Dunfield

Basic Engine Concepts

- Overview of a basic engine
- Event Handling, Animation, and Gameplay
- Asset Loading
- Advanced Gameplay & Collision Detection
- Time Based Logic

Basic Engine Overview

Basic Engine

Components of an Engine

- Variables and Game State
- Game Loop
  - Update
  - Render

Basic Engine

HTML and JS requirements

- Canvas tag
- JS
  - need the canvas ‘context’ for rendering
  - Prototypes

Basic Engine

HTML and JS requirements

- Game Loop - window.requestAnimationFrame()
- FPS indicator for debugging

Event Handling & Animation

& Basic Gameplay

Event Handling

- Bind to ‘window’ object in Engine constructor
- Call handler functions within the Engine scope so we can modify its variables

Animation / Rendering

- Keep the position of objects as a variable in the engine
- Update this variable through gameplay logic
- Render object at this position

Asset Loading

Asset Loading

- We need to load all our images and assets before we begin the game loop
- Process
  - Queue up all the images we want to load
  - Continually call the loading function. It will work its way through the queue and return the % of images it has already loaded each time it is called
  - Once we reach 100% we can start the game loop

Advanced Gameplay & Collision Detection

Advanced Gameplay

- Adding enemies, score and score objects, player health, and a lose condition
- Randomly spawn both enemies and score objects
- Use basic collision detection for enemies and score objects with player
- Mouse click handler to restart game

Time Based Logic

Time Based Logic

The game so far works pretty well.

BUT if the framerate drops or speeds up, so does everything in it.

We need to modify the game to make it frame-rate independent. Instead, we will base our motion on time.

Time Based Motion

- We can figure out how far to move objects each frame by:
  - Defining their speed in px/s
  - Calculating how long it’s been since the last frame (in s)
  - distance(px) = speed(px/s)*elapsedTime(s)

Time Intervals

- Instead of spawning objects each frame, let’s spawn them at periodic intervals:
  - Define how often the event should occur
  - Each frame, check if it’s been long enough
  - If so, perform the event, and reset the counter

Time Intervals

- During the game, you can also change these variables to increase or decrease the frequency of the event
- For example, in the game, enemies will spawn more frequently over time

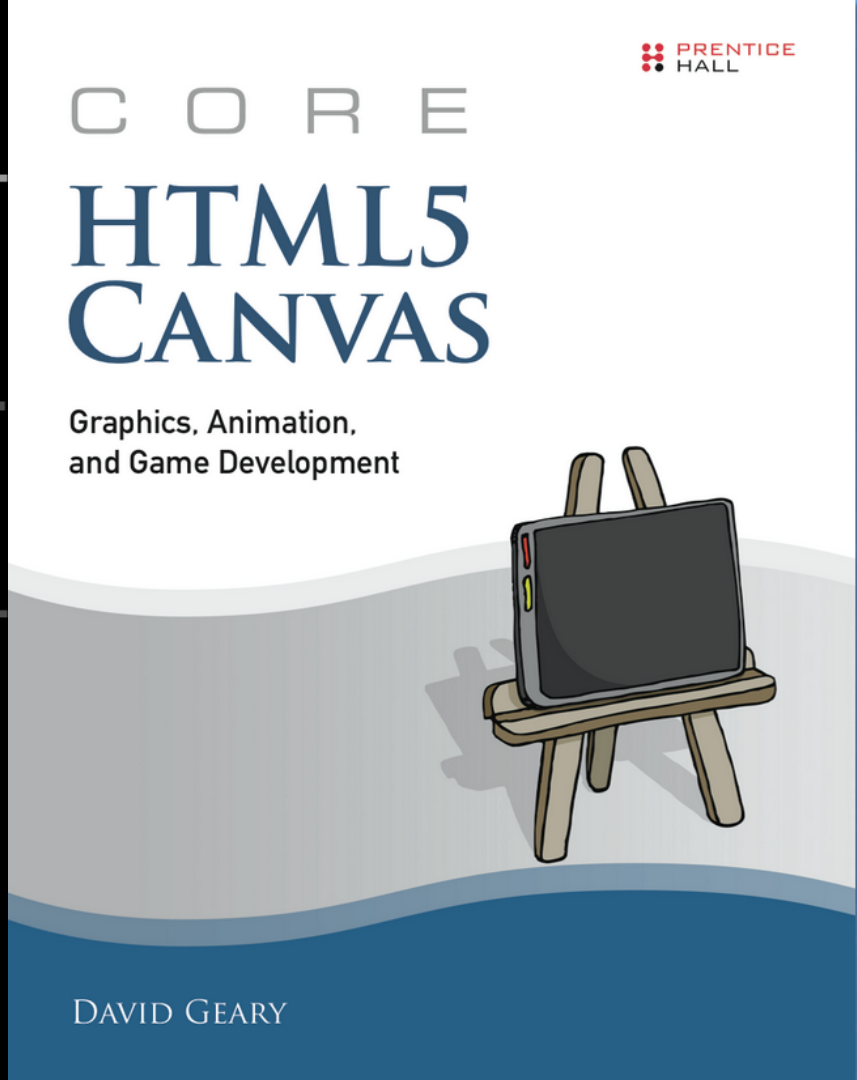
Next Steps

Next Steps

- Spritesheets for characters / objects
- Persistent High Scores

Further Reading

- Core HTML5 Canvas
  - Free ebook!



Thank you