Adam Duvin

Changes:

My game changed significantly, from a gameplay standpoint, from what I originally envisioned. Originally, I wanted the player to have to shoot bullets at the enemies, but I couldn’t figure out how to make the physics work with shooting. Instead, I opted to have the player collide with enemies to kill them.

Requirements:

I have a fittingly dark and spooky song looping throughout the game phase. Enemies attacking, killing enemies, and shooting fireballs all have sound effects. All sound files are mp3. I made all gameplay image assets. They are in an NES-like style. All files are PNG or JPEG. A custom font from google was used for all text. The entire game, except for the start button, are in the HTML5 canvas element. All sprites load image files. The player uses a 4x4 sprite sheet so their character faces the direction they are moving and is fully animated. A particle system is used for the fireball. Animations are smoothed with requestAnimationFrame and use delta time to prevent movement from being processor dependent. During pauses and resumes, cancelAnimationFrame is used to prevent the scene from continuing. The canvas was used to its full effect, with many of the background and text elements being canvas operations.

Interaction:

The player controls a dragon using WASD to move and the mouse to shoot fireballs. Controls resemble those of similar games and are responsive and intuitive. A key daemon was used for the multidirectional movement of the player character, allowing them to move diagonally.

Usability:

The game pauses on blur and on focus. The controls and objective are stated at the bottom of the title screen. Score is visible in the top right as well as wave information. It is clear which state the player is in. Except for the game state, all other states are labeled. The game starts off easy, but the waves become larger and the enemies move faster with each wave. An absolutely positioned button was used for the start button. There is a title screen, a game screen, and a game over screen.

Game Design:

My final product differed a bit from my original plan but mostly fulfilled it. The game is single player, and controls, rules, and the win/lose condition are listed on the title screen. The player’s decisions and strategies impact how long they last. The game doesn’t have the most depth, but gets more difficult at later levels. Players develop better strategies and understanding of the game mechanics as they play. With some refinement and more content, this game would be entertaining to other people. The player sprite is animated and able to face in the four cardinal directions.

Coding:

No frameworks or libraries were used. Images and sound files are preloaded when the page loads. Object literals were used. Modules were used to encapsulate the main game logic and also as pseudo-classes for certain objects. There are many function constructors, like the ones for the player and the castle. The main game and several other modules were put in the same file because I had trouble when trying to use them together. Otherwise the keyboard handler and utility functions are in separate files. Functions were used to prevent repetitive code. At the top of each file, “use strict” was used. Constructors are capitalized. All function names begin with a lowercase letter. All statements end with semicolons. The var keyword is used for variables. Each function has a comment describing what it does. All console.log calls have been removed.

Above and Beyond:

I used JavaScript’s built-in local storage functionality to save players’ games to their browsers. This allowed me to make a high score chart on the game over screen. I made all of the in-game image assets.

What Went Right:

Code and art wise, the game is decent. I like how solid and simple the controls were. I thought my art skills would not be up to the task of creating custom sprites, but I was able to make cohesive assets that fit the theme.

What Went Wrong:

I should have started this project long before I did. I wound up procrastinating and turned the project in a day late. I would have also liked to have separated each module into different files, but I had too much trouble with it when I tried.

Credits (can also be found in the credits.txt document):

Music:

"Dragon and Toast" Kevin MacLeod (incompetech.com)

Licensed under Creative Commons: By Attribution 3.0 License

http://creativecommons.org/licenses/by/3.0/

05224 fireball whoosh

FxProSound

https://freesound.org/people/Robinhood76/sounds/248116/

Converted to mp3

Licensed under Creative Commons: By Attribution 3.0 NonCommercial License

https://creativecommons.org/licenses/by-nc/3.0/

Metal hit with metal bar resonance

jorickhoofd

https://freesound.org/people/jorickhoofd/sounds/160045/

Converted to mp3

Licensed under Creative Commons: By Attribution 3.0 License

https://creativecommons.org/licenses/by/3.0/

Eating Sound Effect

"Crunching, Wooden Fence, A.wav" by InspectorJ of Freesound.org'

https://freesound.org/people/InspectorJ/sounds/352207/

Cut and converted to mp3

Images:

In-Game Assets - Adam Duvin

Title Screen - https://en.wikipedia.org/wiki/Markree\_Castle#/media/File:Markree-castle-by-night-2.jpeg

Game Over Screen - <https://www.flickr.com/photos/holidayonnet/3890512859>

Grade: 90%. The game came out very well and went above and beyond. However, points should be docked for the late submission.