Tower Defense

✅ **Published**   ✏️ **Edit**   ⋮

Introduction

The first projects have given you an opportunity to develop skills in architecting and coding a game, along with learning (probably) a new language and development platform (web). This project's purpose is to put all of that to work in a sophisticated game; this is intended to be the biggest project of the semester.

Warning: The assignment description is open for (small) changes throughout development as you work on the project and discover changes that need to take place. The description is substantially complete as of March 12th, but we may make discoveries along the way that necessitate changes to the design. That sounds an awful lot like real software development, now that I think about it!

Assignment

Write a tower defense style game, detailed below. The following wiki link provides background information on these style of games, and surprisingly, that "Tower Defense" is copyrighted (who knew):

- **https://en.wikipedia.org/wiki/Tower_defense**   **(https://en.wikipedia.org/wiki/Tower_defense)**

Here are links to two desktop tower defense style games you can play online (yuck, these need Adobe flash installed):

- **http://www.kongregate.com/games/preecep/desktop-tower-defense (https://www.kongregate.com/games/preecep/desktop-tower-defense)**
- **http://www.kongregate.com/games/casualcollective/desktop-td-pro (http://www.kongregate.com/games/casualcollective/desktop-td-pro)**

I've decided to lead the assignment based on the technical requirements, rather than the game design. The focus of this course is the technical implementation of games, not game design, therefore the requirements and grading are centered around that. With that said, there are essential game design elements described. These design elements are required, but I want to leave some details of design up to some level of interpretation, as long as a legitimate tower defense game shows up. *If you have any questions about the game design elements, please ask, so we are all in agreement.*

Technical Requirements

- Must be a Node.js server-based solution.

- Game configuration options.
  - Display or not display the tower placement grid.
  - Display or not display the placed tower weapon coverage.

    These configuration settings must be (at least) persisted to the brower's local storage.

- User configurable controls. Most everything is going to be mouse-based, which isn't going to be configurable. However, we can provide keyboard shortcuts, which will allow the user to customize.
  - Upgrade selected tower: u
  - Sell selected tower: s
  - Start the next level: g

  The interface for this must present a screen where the name of the game function is displayed and to the right of it, the key combination. Using the mouse, the user can select the action, then some visual will change to indicate it is possible to now select a new key combination, then the user presses the new key combination (doesn't have to be a combination, could be a single key) and that immediately becomes the new keyboard shortcut for that game function.

  The configured controls must be persisted to the browser's local storage.

- All turrets and creeps must be animated.
  - Turrets must rotate to shoot in the desired direction.
  - Creep movement must be animated, and must point (turning if necessary) in the direction they are moving.

  If you look at the graphics for the first link I provided there are enough graphics to do this. All turrets sit on a base graphic, and then use one of the three level graphics for the actual turret. The base is stationary, while the turret rotates on top of the base.

- Health bars displayed above each creep. Full health is a complete green bar, with the bar progressing to red as health is lost.
- When a creep dies, the score value for that creep is displayed and floats towards the top of the screen from the location the creep died. The floating score should persist for a period of 1 second.
- During tower placement:
  - Draw the radius for its weapon type.
  - The game must check to see if it would block a path for the creeps from an entrance to the exit. If the tower placement would block the path, do not allow the tower to be built; *before* placement, provide a visual cue for the player that indicates this.
- You must create a particle system that allows the game code to start an effect and have that system provide the complete management of all active effects. In other words, your game code will have a single particle system "manager", for the lack of a better term. Then, during gameplay, when an effect is needed, the game code invokes an effect, using the particle system, and it goes from there. Code for invoking an effect might look like...

```
ParticleSystem.createFireEffect({
  position: { x: 100, y: 100 },
```

```
  duration: 1000, // milliseconds
  ... // ...won't be a parameter, that is filler for whatever else is needed for the effect
});
//
// This particle system will also have .update(elapsedTime) and .render functions.  This way
// your gameplay code doesn't have to do anything other than create an effect and let the
// particle "engine" take care of everything.
//
```

(*tasteful*) Particle effects are expected for the following items:

- When a creep dies
- Trail on a bomb as it moves
- When a bomb hits
- Trail behind a guided missile
- Explosion when a guided missile hits
- When a tower is sold
- Creep AI
  - Creeps enter their entrance at random locations and times, at their respective entrance.
  - Creeps must use a shortest path algorithm to move from their current position to their exit.
- Sound
  - Sound effects for firing of weapons, explosions, creep death, tower placement/sale, etc.
  - (optional) Music playing in the background.
- High scores must be persisted to the server.

General Game Design

These are the basic required game design elements. A visual reference from a previous edition of this course can be found at the following link: **here**

- An open arena style gameplay area; rather than a single path maze style.
- The arena is square.
- Creeps may enter from four different openings, each opening centered on each of the arena walls.
- Towers
  - Two types of ground
    - Projectile
    - Bomb : area effect
  - Two types of air
    - Projectile
    - Guided projectile (missile)
  - Three different upgrade levels for each
- Creeps
  - Two ground
  - One air (note that flying creeps aren't blocked by towers, they can, well, fly right over them)
- Levels

- Levels are manually started by the player. When a level is complete, the game pauses, giving the player a chance to regroup and prepare for the next wave. Both between and during a level, the player can add/upgrade/sell towers.
- At least three levels must be part of the game
  - Level 1
    - Ground only creeps start on left and exit on the right: ? waves
  - Level 2
    - Ground only creeps start of the left and exit on the right: ? waves
    - Ground only creeps start on the top and exit on the bottom: ? waves
  - Level 3
    - Ground and air creeps start of the left and exit on the right: ? waves
    - Ground and air creeps start on the top and exit on the bottom: ? waves
- Scoring
  - Resources earned from destroying creeps
  - Total value of towers
  - Number of levels and waves before end of game
- End Game
  - Start with some max number of creeps that can escape, when that number hits 0, game is over.

User Interface

- Menu system
  - New Game
  - Options
    - Game Options
    - Keyboard Controls
  - High Scores
  - Credits

Art & Sound Assets

You may use any art and sound assets that meet the requirements of the game. I've found some art assets that will cover most of the artwork necessary at the following links:

- **http://www.hirefreelanceartist.com/free-tower-defense-graphics.html (http://www.hirefreelanceartist.com/free-tower-defense-graphics.html)**
- **http://opengameart.org/content/tower-defense-art**   **(http://opengameart.org/content/tower-defense-art)**

There are quite a lot of good art assets available at **opengameart.org**   **(https://opengameart.org/)**   If others in the class find good resources, send me an email and I'll add them to the list.

	Points	160
Submitting		a file upload
File Types		zip

Due	For	Available from	Until
Apr 23	Everyone else	Mar 2 at 12pm	Apr 23 at 11:59pm
Apr 27	1 student	-	Apr 27 at 11:59pm
May 4	MangoHabanero	-	May 4 at 11:59pm

Tower Defense

You've already rated students with this rubric. Any major changes could affect their assessment results.

Criteria	Ratings		Pts
Node.js server-based solution All files served to the browser from a server.	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
User configurable controls (persisted to local storage) and game options * Ability to change keyboard controls * Ability to toggle on/off placement grid * Ability to toggle on/off tower weapon coverage	15.0 pts Full Marks	0.0 pts No Marks	15.0 pts
Animated turrets & creeps	15.0 pts Full Marks	0.0 pts No Marks	15.0 pts
Health bars on creeps	5.0 pts Full Marks	0.0 pts No Marks	5.0 pts
Floating score on creep death	5.0 pts Full Marks	0.0 pts No Marks	5.0 pts
Tower placement radius and blocking * Visual indication a tower can/can't be placed at the location * Visual indication of a tower's weapon reach	15.0 pts Full Marks	0.0 pts No Marks	15.0 pts
Particle system & effects	20.0 pts Full Marks	0.0 pts No Marks	20.0 pts
Creep shortest path travel	10.0 pts Full Marks	0.0 pts No Marks	10.0 pts
Sound	5.0 pts Full Marks	0.0 pts No Marks	5.0 pts
Game scoring, with high scores persisted to server	5.0 pts	0.0 pts	5.0 pts

Criteria	Ratings		Pts
	Full Marks	**No Marks**	
Game Design - Three Levels Including the paced entry and randomized starting position at each entrance.	**12.0 pts Full Marks**	**0.0 pts No Marks**	12.0 pts
Four tower types This also includes the three upgrade levels for each.	**16.0 pts Full Marks**	**0.0 pts No Marks**	16.0 pts
Three creep types Two ground, one air	**12.0 pts Full Marks**	**0.0 pts No Marks**	12.0 pts
Overall game and menu interface/presentation	**15.0 pts Full Marks**	**0.0 pts No Marks**	15.0 pts
			Total Points: 160.0