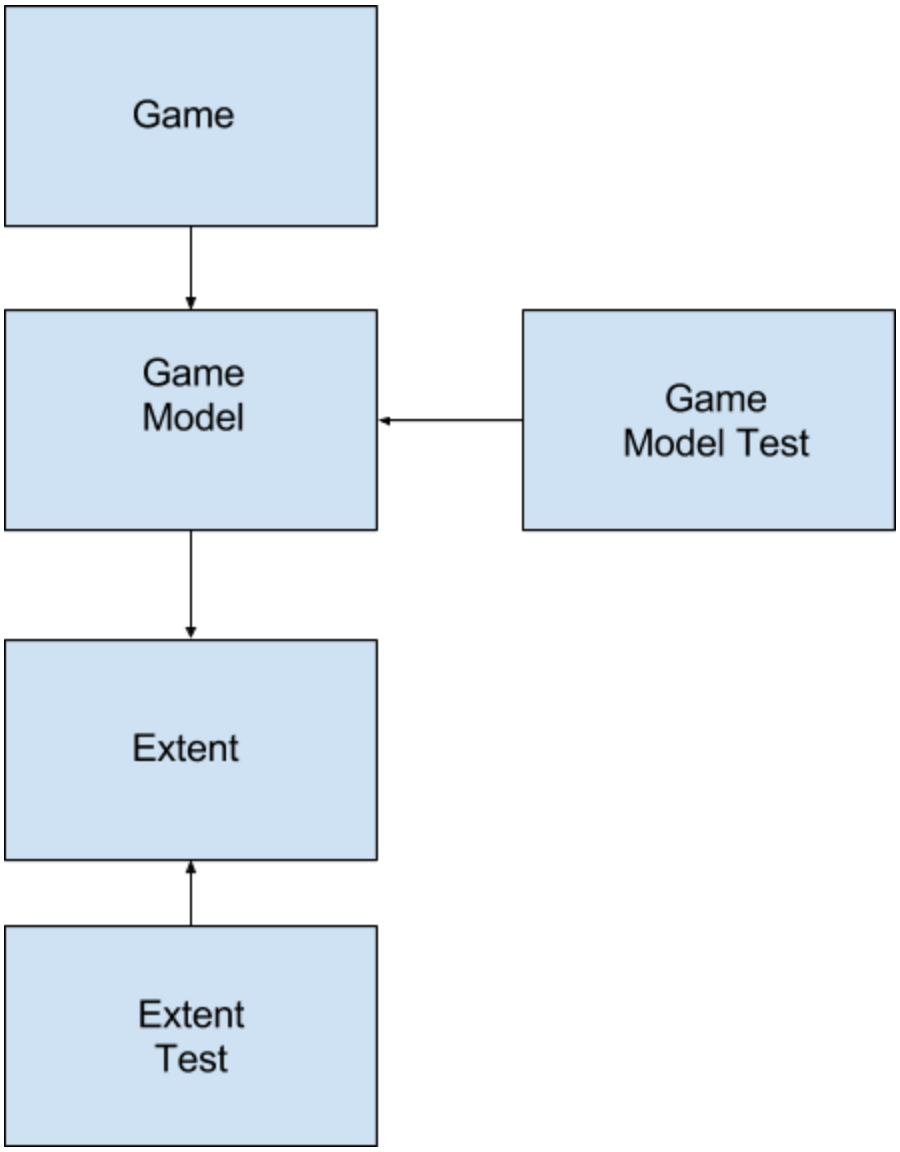
Team: Anna Schall, Jodi Rush, Emma Moorhead, Leon Rogge
Project: This is the Only Level

UML Diagram:



Classes:

1. *Game* is the GUI for the game. It's methods include:

➢ Draw —draws the state of the board and calls upon other draw methods
➢ Draw Platform —draws each of the platforms by getting values from GameModel
➢ Run —runs the game and checks for player movement, jumping, hitting spikes, and handleKeyPress
➢ Draw Player —draws the character, uses extent to get player coordinates

- ➢ handleKeyPress —calls on move to shift player coordinates left or right
- ➢ Jump —checks for current floor, causes jumping movement, and sets gravity of environment
- ➢ Draw Spike —draws triangles to represent spikes of doom
- ➢ WinScreen and TitleScreen - splash screens that display game start and game end

2. *GameModel* is the logical model of the game, it contains all getters, counts death, checks if button has been pressed or exit reached.

- ➢ Game Model —sets starting values of player, platforms, and spikes
- ➢ Getters —fifteen getters for platforms, player, button, gate, and deaths
- ➢ Button Press —checks if player overlaps button, returns true if so, and gets rid of gate when button has been reached
- ➢ Exit Reached —checks if player overlaps with exit, returns true if so
- ➢ Spike Hit —checks if player overlaps with spikes, returns true if player hits spikes, resets board to initial state, and adds to death count

3. *Extent* gets values for the objects we create and includes methods that check for movement, current floor, and overlap.

- ➢ Extent methods —two extent constructors, one for player (circle) and one for platforms, spikes, and hit boxes (rectangles)
- ➢ Getters —five getters for length, height, width, x, and y
- ➢ Setters —four setters for height, width, x, and y
- ➢ Move —allows for horizontal player movement
- ➢ Current Floor —returns current floor of player dependent on platform they are on or above using the player's coordinates
- ➢ Overlaps —detects overlaps between circular player and rectangular objects

The remaining classes are test classes. We have tests for GameModel which include:
These tests are not currently functional.

- ➢ Is the level complete when the player overlaps with the exit and button is true?
- ➢ Is the button pressed? If so, is the gate set to null?
- ➢ Is the player deaths counted and is the player respawned at the start?

We will also write tests to determine:

- ➢ Does ExitReached end the level? And does it set up the next level?
- ➢ Does beating five levels end the game?

These tests are tests in ExtentTest, that test methods in Extent.

They currently pass.

- DetectsOverlap- three tests (detectsOverlap, detectsLackOfOverlap, and detectsDiagonalOverlap) that asses is the overlaps method is working
- StoresX and StoresY - checks to see if Extent stores X and Y values
- Move - checks if horizontal movement changes and stores a new X for player

Next steps:

Player will not be able to go through the bottom and side of platforms, will not vibrate on platforms, will not cling to the ceiling, and will not be able to jump indefinitely. We wish to condense display methods into arrays that store locations of objects to be drawn and have deaths and level titles be displayed on the screen.