1. Grid Hack
   1. Game design
      1. Teams

* There are at least two teams in each level.
* The players all belong to a single team.
* Teams can be either hostile or friendly to other teams.
  + 1. Turns
* The players, and their team, go first
* All other agents follow, by team
* Turn order is not defined in the level data, although there might be potential for this at a later stage
  + 1. Movement
* Agents can move through friendly agents.
* Agents can’t attack friendly agents.
* Agents can attack hostile agents.
* Agents cant’ move through hostile agents.
* Agents have either fixed or random (d6) movement
  + 1. Victory condition
* The players win if they complete the level objective.
* The players loose if they all die.
* The players loose if they can’t complete the objective.
  + 1. Map Visibility
* Each team has map visibility
* Initial visibility is set to false for each team
* For the initial implementation only the players can activate board visibility
* For simplicity, the players update visibility for both teams. This could be changed in the future
* Players activate monsters upon viewing them
  + 1. Actions
* Agents can move.
* Agents can use a weapon (attack).
* Agents can use a spell (varied effect).
* Agents can use a potion (varied effect).
* Agents can jump.
* Agents can search (random effect).
* Agents can use an object.
  1. To Do
* Add a hud to the display to allow the user to define actions.
* The hud should not scale with the map: it has a fixed location.
* Add objects to the game.
* Determine a way of handling objects/agents that are not a single cell in size.
* Graphics required.
* Generate a path for display when planning a move.
* Consider an inventory interface in the hud for selecting weapons. Maybe simply a form of drop down box.
  1. stateSetup.js
     1. To Do
* ~~Initialize the board visibility~~
* Set board starting visibility with triggers
* Setup turn order prior to starting the game
* Set visible monsters to active
* Set invisible monsters to inactive
  + 1. Private variables
       1. Placed
    2. Global variables
       1. msgPump
    3. Methods
       1. initialize = function()
       2. run = function()
       3. render = function()
       4. update = function()
       5. addPlayer = function(args)
       6. setMouseFocus = function(cellX, cellY, mouseX, mouseY)
  1. stateGame.js
     1. To Do
* Set up a game turn loop
* Handle the current players turn
  + 1. Private variables
    2. Global variables
    3. Methods
       1. initialize = function()
       2. run = function()
       3. render = function
       4. update = function
  1. Doors.js
     1. To Do
        1. onUse

The args given to Door may require knowledge of the actual user.

* + 1. Methods
       1. Door(cell1, x1, y1, cell2, x2, y2, open, images)
       2. onUse = function(args)
       3. onMouseOver = function(args)
       4. draw = function(context, x, y, size, scale, sprites)