**Intro:**

I have enjoyed reading the code you wrote, it is functional, but I would like to leave you a trail of crumbs for improving it. We will begin with making a new Javascript file called game.js. This file will serve as a way of outsourcing the low-level complexity, as such we can focus on the high-level implementation of the game in the moth2.htm file.

**Step One:**

Let’s begin with the concept that our code should have an intermediary to manage the game state. This will be our game.js file. Here we should begin with replacing our start() and stop() methods from the moth1.html file. We should basically copy paste them for now in to the game.js file.

**Step Two:**

In the game.js file we should probably provide some state information about the state of the game (aka what state the ticker is in). This should take the form of a single global variable in the file. Additionally, here we should make a list of constants, the list should include simple indicators for the running or stoppage of the game. This list will represent the state the ticker is in, as such it should be changed from running to stopped when the ticker’s state changes.

**Step Three:**

With our new game states in place it will be important to update and change game states when the ticker is updated. This can be implemented efficiently with a switch statement, or in a longer format with if/else if statements.

**Step Four:**

As a small intermediary step, I would recommend making a game\_init method, this is integral to setting up the game state as stopped initially in addition to ensuring that the ticker is paused. This will essentially remove the line of code at line 62 in moth1.htm.

**Step Five:**

I think it would be prudent to remove the anonymous function in the tick event listener to outsource it to the game.js file. I would make a new method for this code, and instead place this method name where the old anonymous function previously was. Now in the game.js method you should implement a way of checking the state of the ticker in this method (this is where our list of game state constants come in). If you have been attending to/updating your game state constants this will enable you to only update the screen when the ticker is un-paused, and will obey the command to pause the game. I would recommend you implement this in a switch statement.

**Step Six:**

Finally, in the moth2.js file I feel it would be a good idea to provide an enumerated constant list of states that the moth can be in, as such that object is “cognizant” of the state in which it resides. These states would provide “linkage” and a way of tackling what exactly the moth is doing while the game is running. These states would greatly simplify the ending of the game, by knowing with certainty what state the moth is in. I would use this information to add conditional checks in the dies() and update() methods.