# Sprint 1: Learning outcomes journal

James Browne

Contents

[Sprint 1: Learning outcomes journal 1](#_Toc343269570)

[LO 1: Cloning models in Three.js 3](#_Toc343269571)

[The goal: 3](#_Toc343269572)

[Problem: 3](#_Toc343269573)

[Solution: 3](#_Toc343269574)

[LO 2: Creating intervals 5](#_Toc343269575)

[The goal: 5](#_Toc343269576)

[Problem: 5](#_Toc343269577)

[Solution: 5](#_Toc343269578)

[LO 3: Stopping intervals 6](#_Toc343269579)

[The goal: 6](#_Toc343269580)

[Problem: 6](#_Toc343269581)

[Solution: 6](#_Toc343269582)

[LO 4: Fading a canvas 7](#_Toc343269583)

[The goal: 7](#_Toc343269584)

[Problem: 7](#_Toc343269585)

[Solution: 7](#_Toc343269586)

[LO 5: Drawing videos using the canvas 8](#_Toc343269587)

[The goal: 8](#_Toc343269588)

[Problem: 8](#_Toc343269589)

[Solution: 8](#_Toc343269590)

## LO 1: Cloning models in Three.js

### The goal:

Rather than go through the process of re-loading a model from file I would like to clone an existing model. This will save time, effort and hardware resources. Ideally I would like it to be in the mesh loader object that I created to control the model loading in the previous sprint.

### Problem:

The main worry is that there isn’t a function in the collada loader to clone a mesh. That would mean coding something out of my skill set, animation, which will take a lot of time and effort.

### Solution:

Research links:

<https://github.com/mrdoob/three.js/issues/764>

The Three.js script comes with a scene utility class that has a “cloneObject” function. When the “ColladaLoader.js” loads a collada file it stores it as a Three.js mesh. The “cloneObject” basically creates a new mesh object and does a deep copy before returning the new cloned mesh.

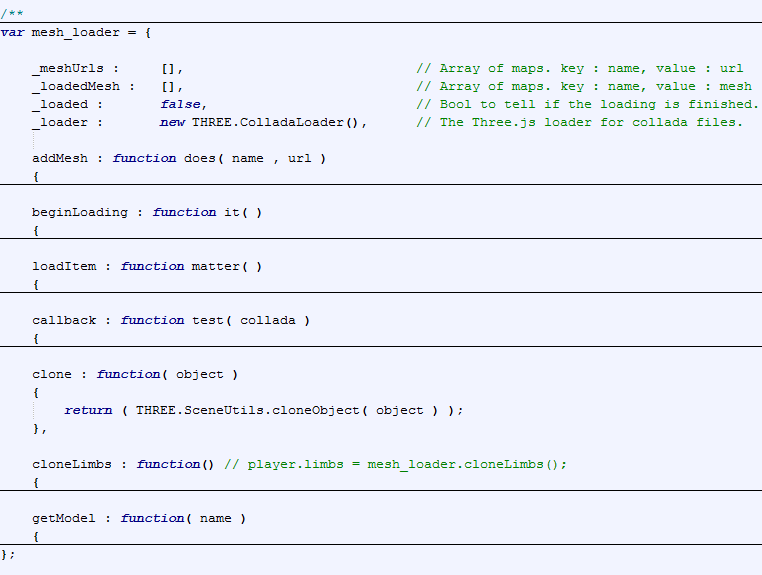


Figure ) Adding the clone function

With knowing the above I can now begin integrating this into the project. I created a function in the mesh loader called “Clone” which accepts a mesh argument to clone. It uses the “cloneObject” function and returns the new mesh. Shown in figure 1 is the integration into the mesh loader. In figure 2 below you can see an example usage where I clone a set of 14 meshes stored in an array and return them to the objects that want them.

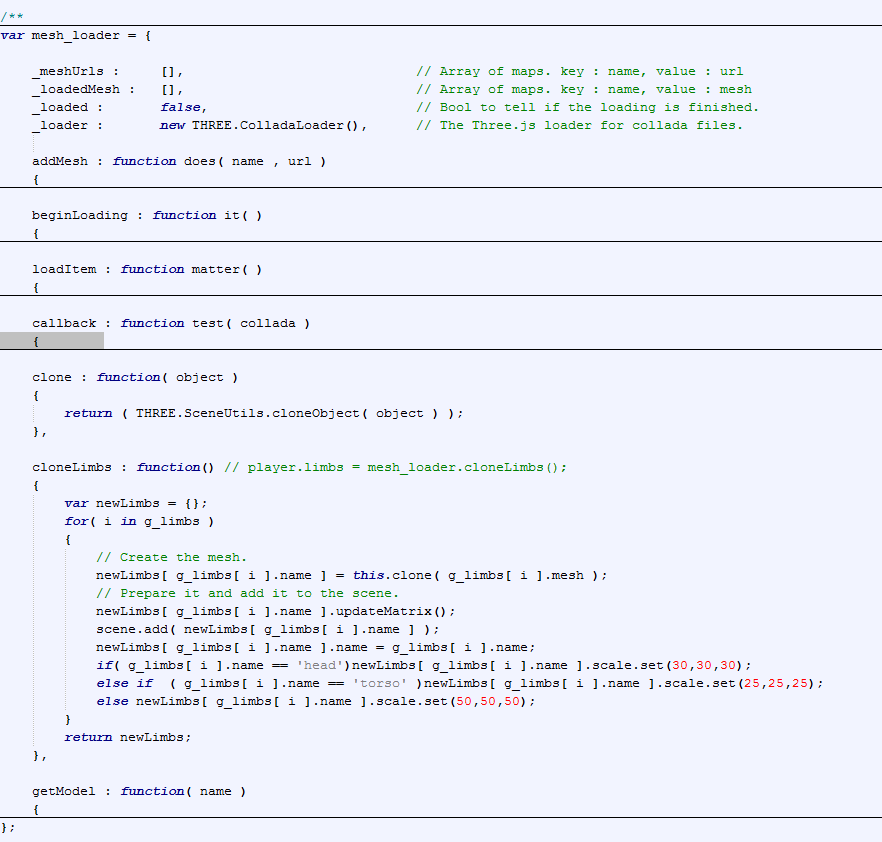


Figure ) Cloning in use

## LO 2: Creating intervals

### The goal:

To create a game loop that uses interval timing and storing the id of the timer.

### Problem:

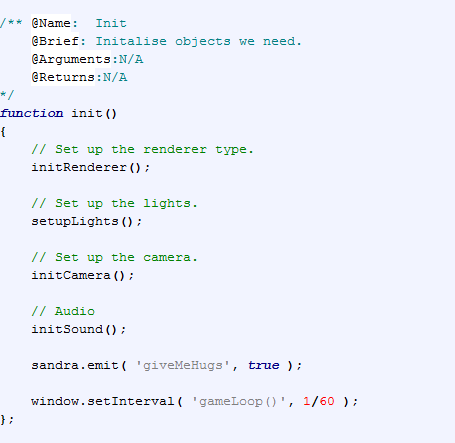
I want to be able to start and stop a game loop whenever I want to.

### Solution:

Research links:

<http://www.w3schools.com/jsref/met_win_setinterval.asp>

Following reading w3schools article on “setInterval” I implemented it like this.



## LO 3: Stopping intervals

### The goal:

To be able to do some logic, such as the fading, uninterrupted I needed to stop the game loop for a while.

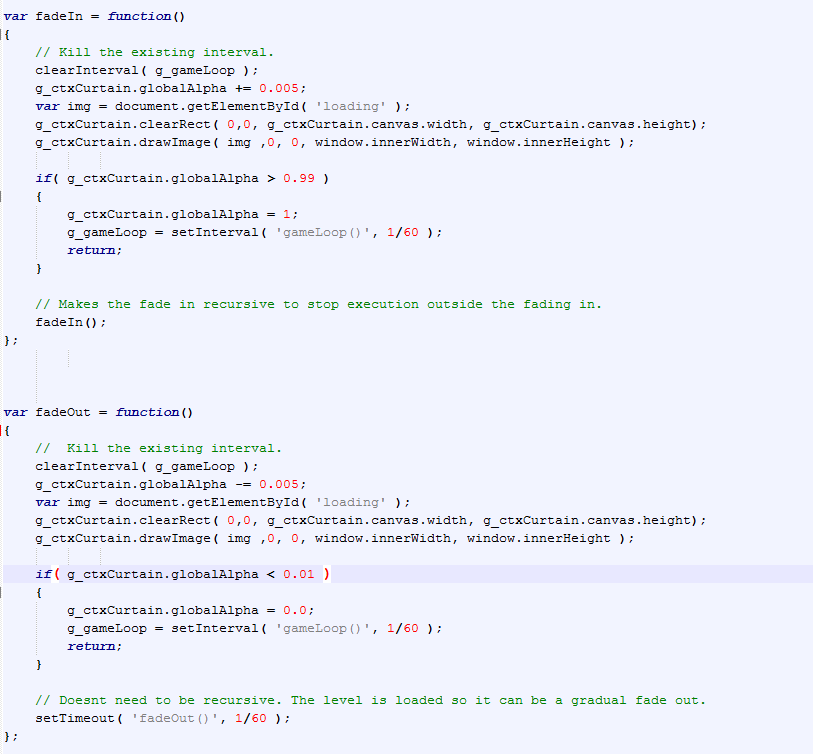
### Problem:

Figuring out how to halt/pause/stop a “setInterval” function in JS/HTML.

### Solution:

<http://www.w3schools.com/jsref/met_win_clearinterval.asp>

After reading the article above, I used the code and implemented it as follows. Firstly I clear the interval and then proceeded by recursion the logic I wanted to do. After certain conditions were met I then started the game loop interval again.

****

## LO 4: Fading a canvas

### The goal:

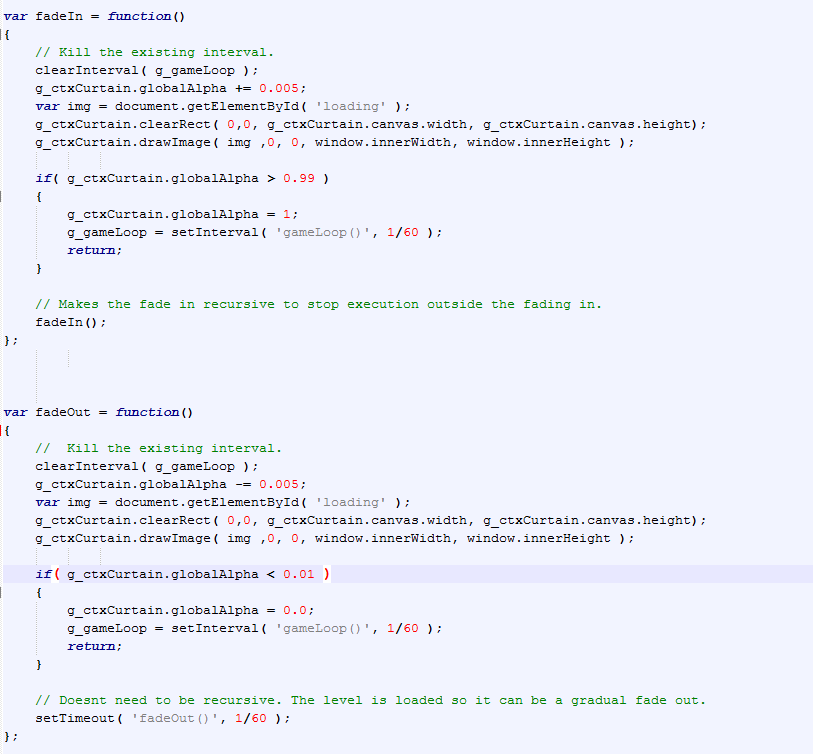
To blend two canvas elements together gradually

### Problem:

The transitions between the levels were too quick and sloppy so I needed to fade in between levels.

### Solution:

What I did was create a fadeIn and a fadeout function which stops the current game loop and either increases or decreases the canvas with a loading image displayed on it. After the alpha level is at an optimum level I started back up the game loop again.



## LO 5: Drawing videos using the canvas

### The goal:

To display a video on screen as a tutorial method for the users to know what to do during the game

### Problem:

The users didn’t know automatically how to play the game so I needed to find a way of playing a video to give them the information they needed.

### Solution:

I loaded in the video from the webpage and used “Play()” to play it. To show it onscreen I used a canvas to draw the current frame of the video to the screen.

