Pseudocode Design Layout: Get the Flappy Bird to move & flap!

10/29/17 – WebScripting 235

Initial start of code:

Functions

- msToSec

Global Variables – left unchanged

- canvasElement **
- gameContext **
- birdImage **
- skyBackgroundImage **
- startX
- birdXSpeed

** (will be used in View: can be moved into View constructor instead, or called as global variable)

Initialize Objects

- startPoint = new Point(pass requisite arguments);
- bird = new Bird(pass requisite arguments [INCLUDING startPoint]);
- c = new Controller(pass requisite arguments);

Call the start function! (which should be written in the controller class)

- c.start();

class Controller

constructor (bird_object [passed from initial start of code]) {

- Initialize Event listener
- Declare new variable and initialize it as a new View() object
- Declare a new variable and assign it to **bird_object**

}

start() {

- Declare a new variable to track the "last time bird moved", assign to 0 to start (variable's important for distance formula in bird.move)
- Create function:
    - runGame = ms => {

      [After one run-through, "last time bird moved" variable should now hold the ms value of last iteration of (runGame)]

      move bird using its move method (ms – "last time bird moved"); [make sure to convert from milliseconds into seconds here!]

      update view by calling its render method;

      assign value of current ms to "last time bird moved" variable;

      call runGame function using requestAnimationFrame [to continue in an infinite loop]

      }
- call runGame function using requestAnimationFrame [to start infinite loop of runGame]

}

class View

constructor (bird_object [passed from Controller class] {

- Declare new variable and Assign it to **bird_object**

}

render() {

```
clear canvas                                          ** global variables should be used
draw the sky (based on bird's new position!)   ** global variables should be used
draw the bird (based on bird's new position!)  ** global variables should be used
```

}

class Point

constructor(x, y [both passed from initial start of code, and move() function from a bird_object]) {

- Declare new variable and Assign to **x**
- Declare new variable and Assign to **y**

}

get x( ) {

- return “variable for x”

}

get y( ) {

- return “variable for y”

}

class Bird [aka the “model” class]

constructor(startPosition [this should be a Point object], startXSpeed, gravity, flapUpSpeed [each passed from initial start of code]) {

- Declare and Assign variables for each value passed via constructor (as defined in the constructor’s parameters above)
- Declare and Assign variable for **currentYSpeed** (initial should just be 0)

}

move(secondsElapsed [passed from controller]) {

- Declare variable for new x location and Assign it to (the current position plus the product of (**secondsElapsed** times the current X speed)
- Declare variable for new y location and Assign it to (the current position plus the product of (**secondsElapsed** times the current X speed)
- Initialize a new Point object; set its x and y to the new x and why location (variables we made above)
- Update current y speed variable by Assigning it to (the current y speed plus the product of (**secondsElapsed** times gravity)

}

flap( ) {

- Update this object’s **currentYSpeed** to be this object’s assigned flap speed (value passed via constructor)

}

get position() {

- return this.”this object’s assigned variable (in constructor) for **startPosition**” }