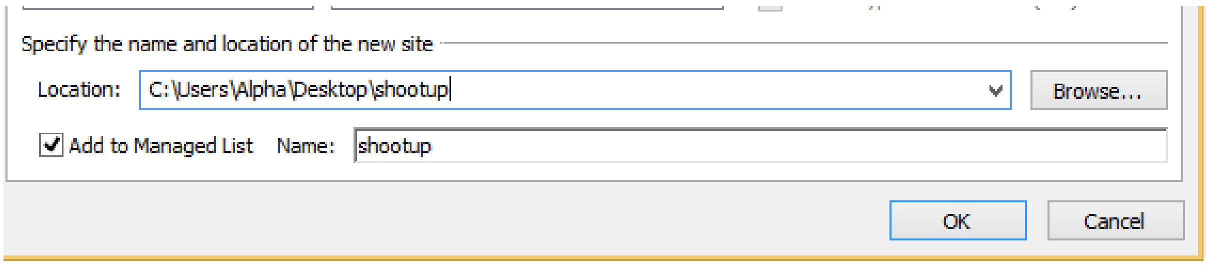
Tutorial 7

Activities

- Code
  - Editor: Expression Web
  - Focus : Base Layout, navigation with folders, external stylesheets, javascript game,

Javascript Game

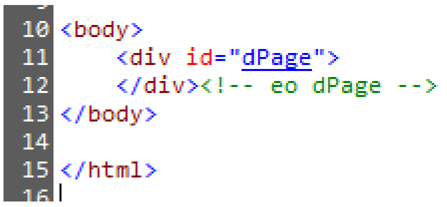
So start a new web site

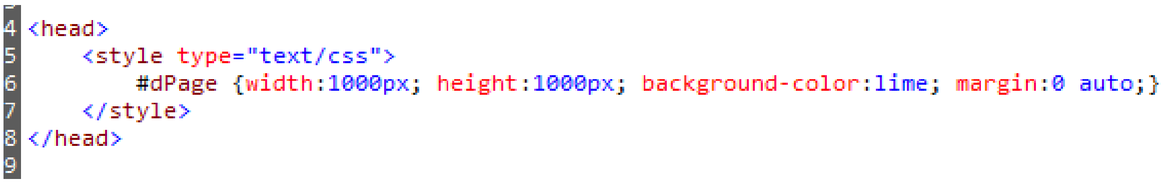

Specify the name and location of the new site

Location: C:\Users\Alpha\Desktop\shootup

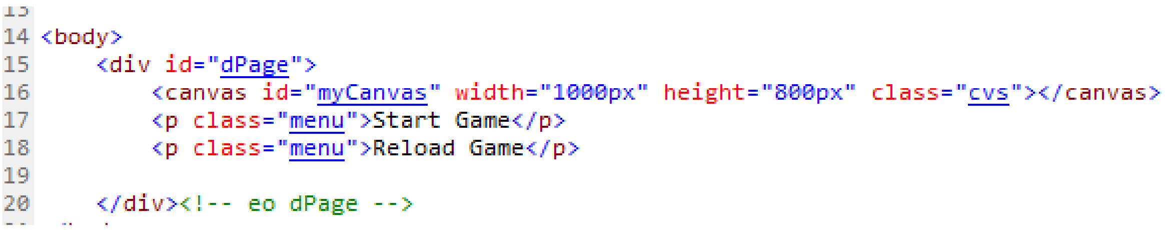
[✓] Add to Managed List   Name: shootup

- Rename to index.html
- From here we're going to make a very simple structure, just create a 1000px X 1000px dPage and float it in the centre of the screen

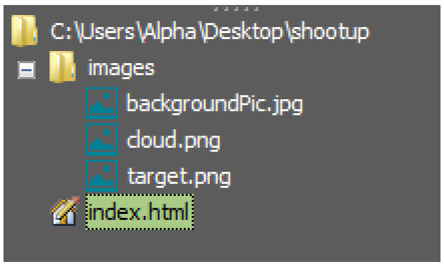

```
<body>
    <div id="dPage">
    </div><!-- eo dPage -->
</body>

</html>
```

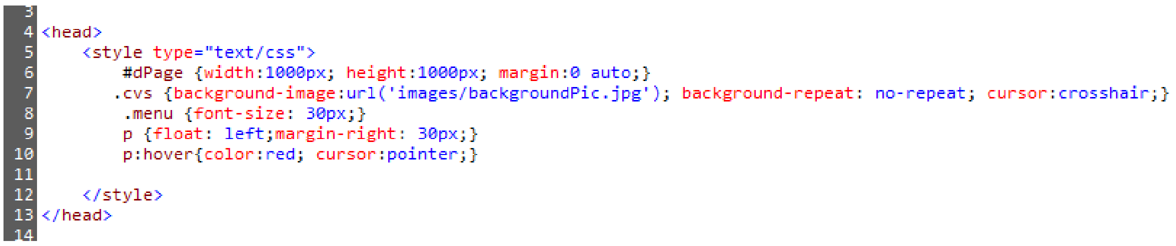

```
<head>
    <style type="text/css">
        #dPage {width:1000px; height:1000px; background-color:lime; margin:0 auto;}
    </style>
</head>
```

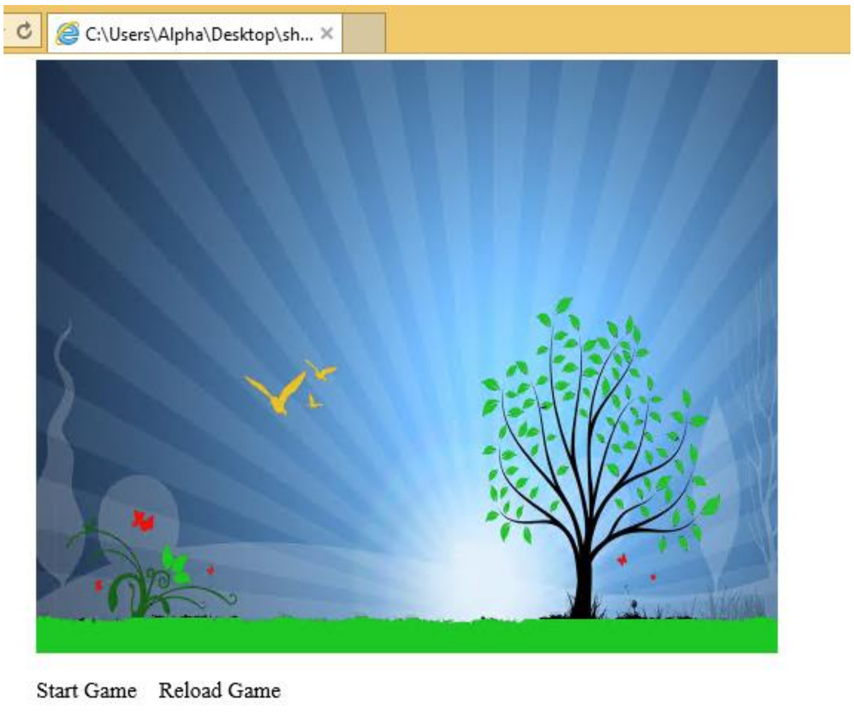
- Now to create the javascript game , everything we do is going to occur within a canvas tag, so we'll create a canvas tag and a couple of links for starting and reloading the game


```
<body>
    <div id="dPage">
        <canvas id="myCanvas" width="1000px" height="800px" class="cvs"></canvas>
        <p class="menu">Start Game</p>
        <p class="menu">Reload Game</p>

    </div><!-- eo dPage -->
```

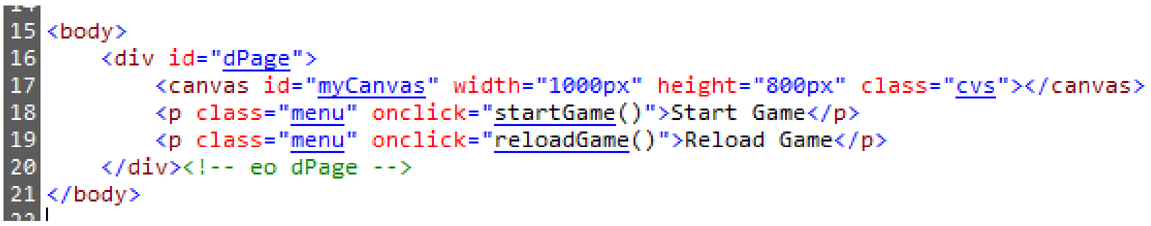
- Get a background image, cloud image and a target image, place them in a folder.

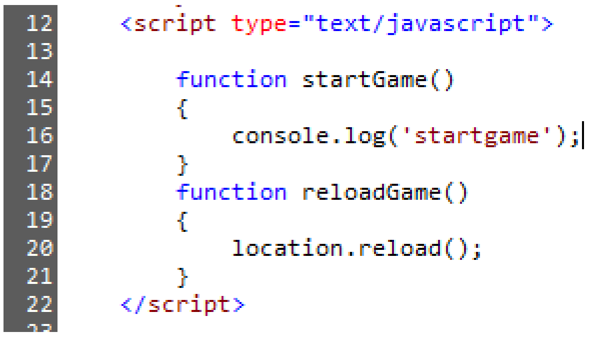
- 
```
C:\Users\Alpha\Desktop\shootup
  images
    backgroundPic.jpg
    cloud.png
    target.png
  index.html
```
- Now we apply the styles for each element of the page.

- 
```
<head>
    <style type="text/css">
        #dPage {width:1000px; height:1000px; margin:0 auto;}
       .cvs {background-image:url('images/backgroundPic.jpg'); background-repeat: no-repeat; cursor:crosshair;}
        .menu {font-size: 30px;}
        p {float: left;margin-right: 30px;}
        p:hover{color:red; cursor:pointer;}

    </style>
</head>
```
- Run this up and test,

- 

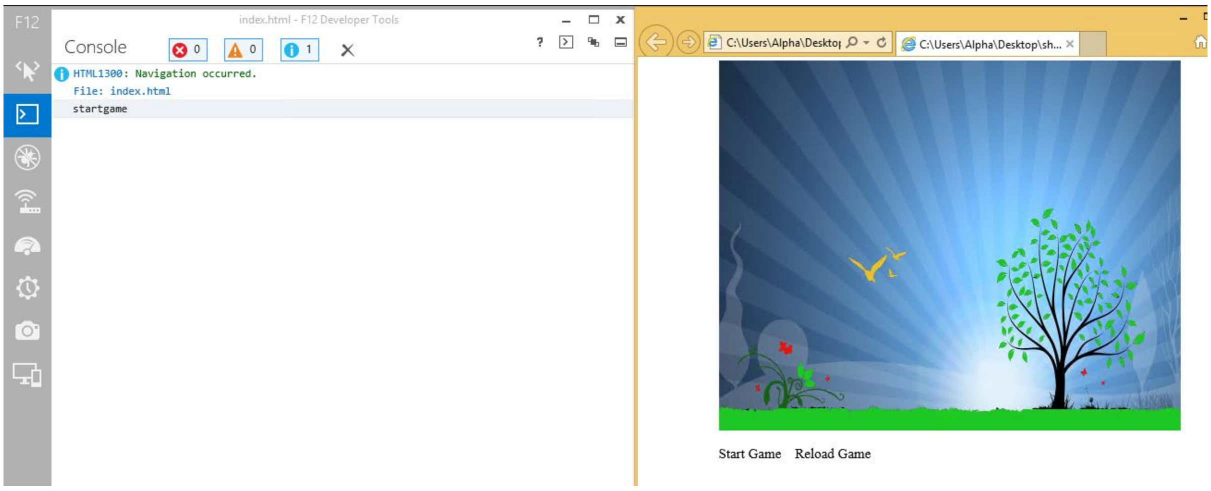
- When you run your mouse over the background image it should change to a crosshair, when you run your mouse over the Start Game/Reload Game, it should change to red text and the cursor changes to the pointer icon.
- Now for the javascript.

- 
```
<body>
    <div id="dPage">
        <canvas id="myCanvas" width="1000px" height="800px" class="cvs"></canvas>
        <p class="menu" onclick="startGame()">Start Game</p>
        <p class="menu" onclick="reloadGame()">Reload Game</p>
    </div><!-- eo dPage -->
</body>
```
- Add the javascript



```
<script type="text/javascript">

    function startGame()
    {
        console.log('startgame');
    }
    function reloadGame()
    {
        location.reload();
    }
</script>
```

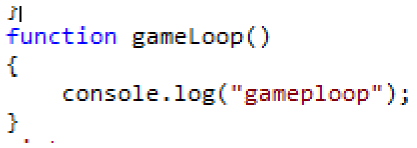
- Now test it and open the console log, you want to see the words startgame appear when you click that link and the page to refresh when you hit reload
- Remember to allow blocked content to see the javascript in action



```
HTML1300: Navigation occurred.
File: index.html
startgame
```

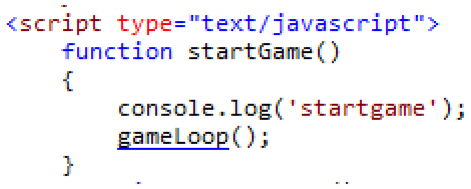
index.html - F12 Developer Tools
Console
C:\Users\Alpha\Desktop
C:\Users\Alpha\Desktop\sh...
Start Game
Reload Game

- Now, my development environment, ie console, will look different as I am using internet explorer 11 on my virtual machine.
- Now we create the gameLoop code



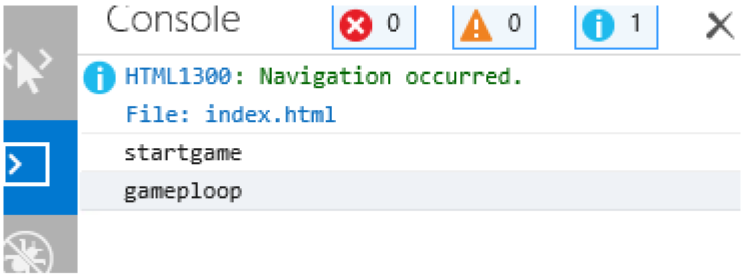
```
function gameLoop()
{
    console.log("gameploop");
}
```

- Call it in the startgame section and test it



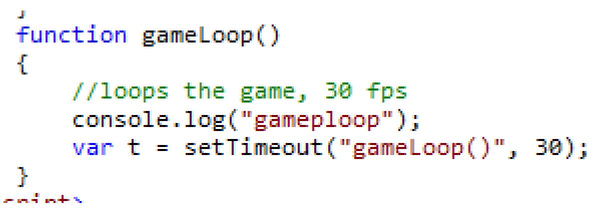
```
<script type="text/javascript">
    function startGame()
    {
        console.log('startgame');
        gameLoop();
    }
```

- When you click on startgame you want to see the following appear in the console

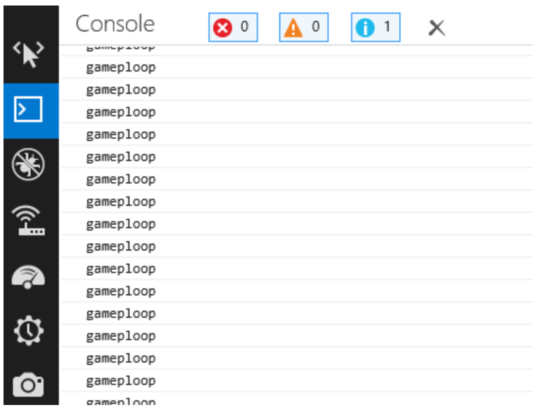


```
HTML1300: Navigation occurred.
File: index.html
startgame
gameploop
```

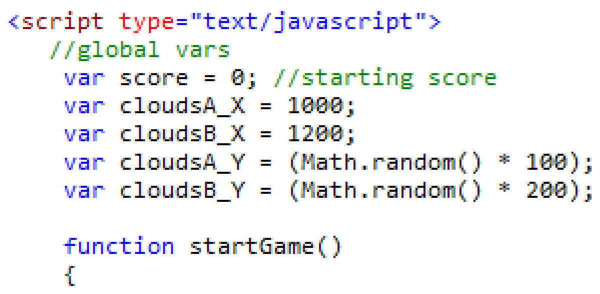
- Next we instigate the timer that cause the game to loop

- 

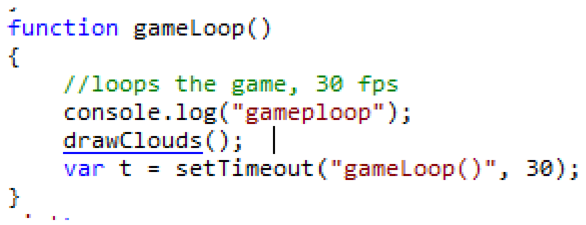
```
function gameLoop()
{
    //loops the game, 30 fps
    console.log("gameploop");
    var t = setTimeout("gameLoop()", 30);
}
```

- Run this up, the gameloop should be continuously listed.
- 

```
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
gameploop
```

- Reload the game to stop that from occurring
- Now we need to put in a moving cloud, just to give the canvas a bit of activity
- To make this happen we need to place some global variables into the code, global in that every function can touch them, not just having them restrained to a singular function.
- 

```
<script type="text/javascript">
   //global vars
    var score = 0; //starting score
    var cloudsA_X = 1000;
    var cloudsB_X = 1200;
    var cloudsA_Y = (Math.random() * 100);
    var cloudsB_Y = (Math.random() * 200);

    function startGame()
    {
```

- We then add the ability to draw the clouds into the gameLoop
- 

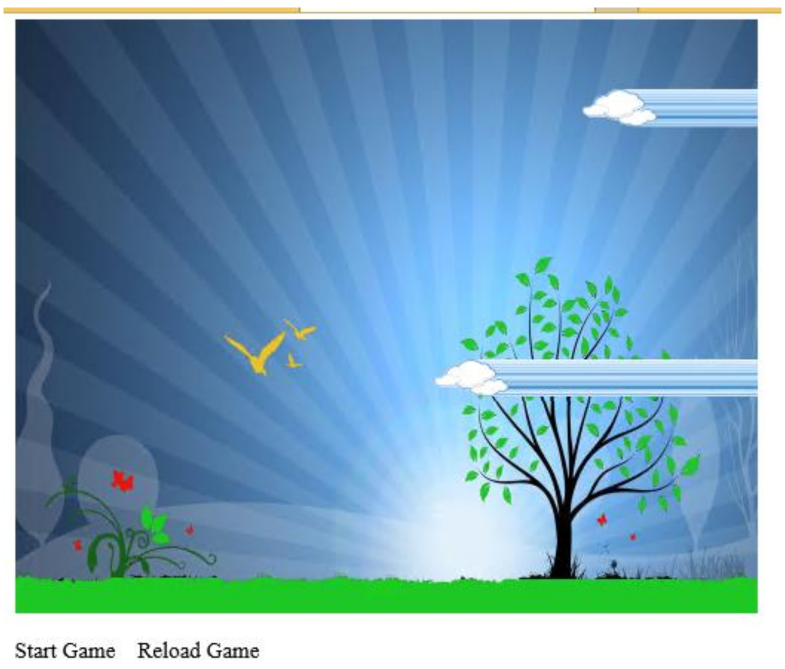
```
function gameLoop()
{
    //loops the game, 30 fps
    console.log("gameploop");
    drawClouds();
    var t = setTimeout("gameLoop()", 30);
}
```

- And create a new function called drawClouds()

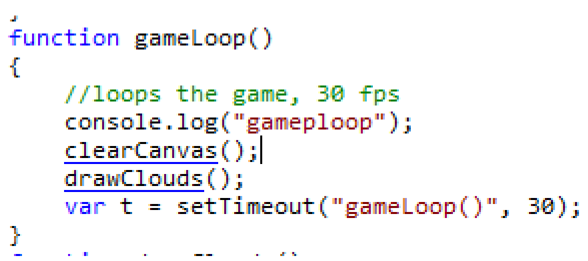
```
}
function drawClouds()
{
    //draws the clouds and moves them from right to left on screen
    var ctx = document.getElementById("myCanvas").getContext('2d');
    var clouds = new Image();
    clouds.src = "images/cloud.png";
    clouds.onload = function () {
        if (cloudsA_Y > 50) {
            ctx.drawImage(clouds, cloudsA_X, 400 + cloudsA_Y);
        }
        else {
            ctx.drawImage(clouds, cloudsA_X, 400 - cloudsA_Y);
        }
        if (cloudsB_Y > 50) {
            ctx.drawImage(clouds, cloudsB_X, 100 + cloudsB_Y);
        }
        else {
            ctx.drawImage(clouds, cloudsB_X, 100 - cloudsB_Y);
        }
    }
    cloudsA_X--;
    cloudsB_X--;
    if (cloudsA_X == -150) {
        cloudsA_X = 1100;
    }
    if (cloudsB_X == -150) {
        cloudsB_X = 1000;
    }
}
```

```
/script>
```

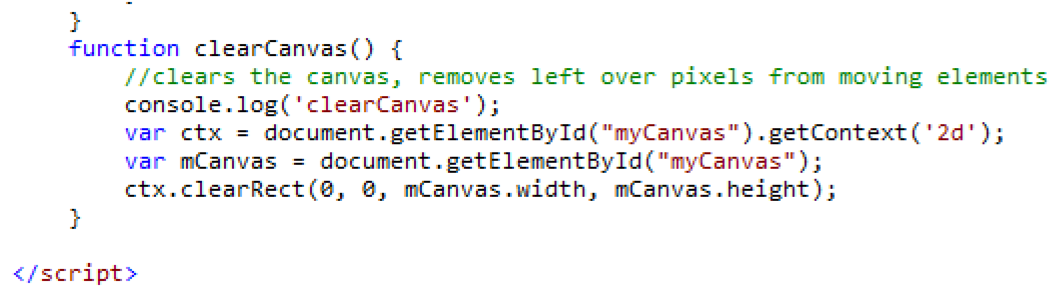
-
- This produces



-
- So the code draws our clouds and starts to move them across the screen, notice the trails that is left behind each of the clouds, this is because after each loop of the game loop the image is redrawn, but not cleaned up, so we are left with trails.
- So now, we need to write up some code to clean up canvas.
- We will call this clearCanvas() and have it sit in the game loop. clearCanvas is a new function

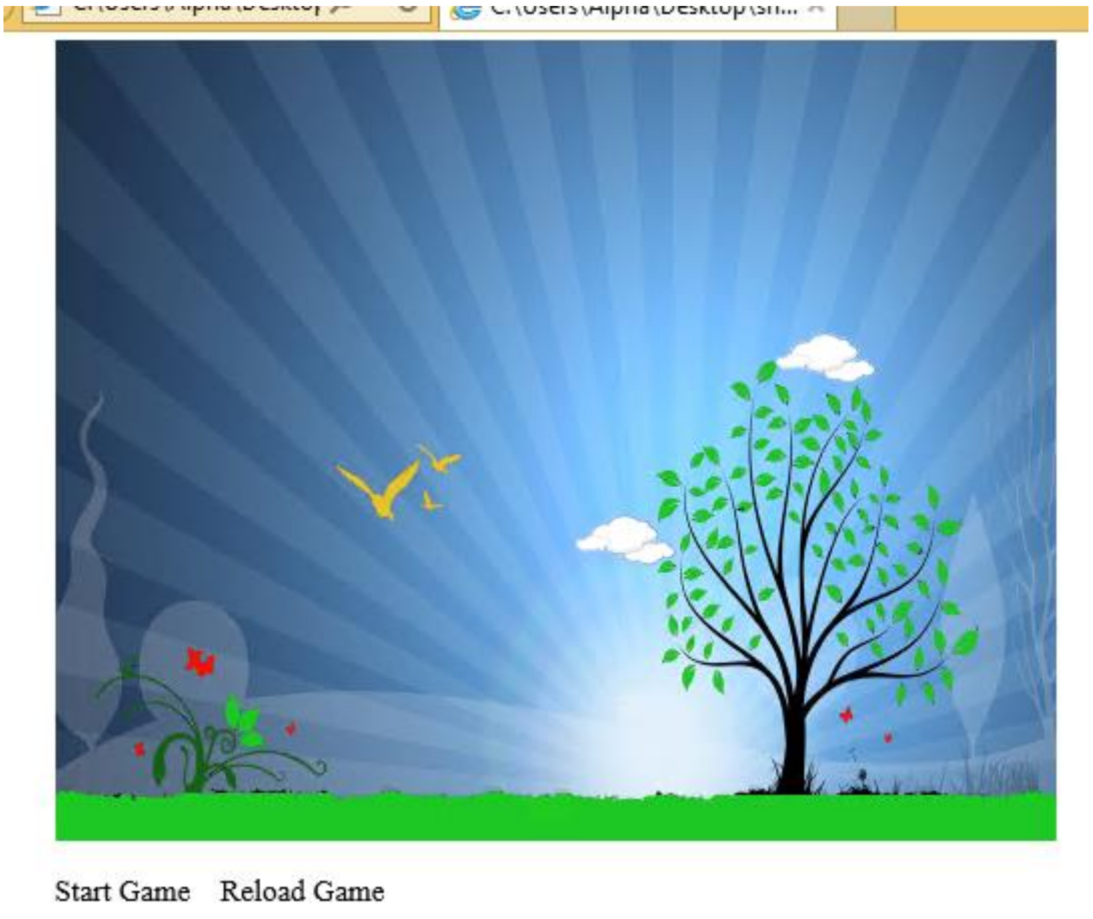


```
function gameLoop()
{
    //loops the game, 30 fps
    console.log("gameploop");
    clearCanvas();
    drawClouds();
    var t = setTimeout("gameLoop()", 30);
}
```

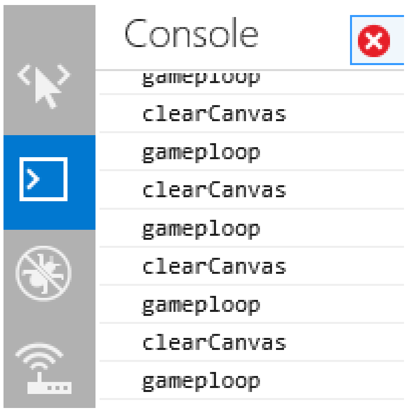
- 

```
    }
    function clearCanvas() {
        //clears the canvas, removes left over pixels from moving elements
        console.log('clearCanvas');
        var ctx = document.getElementById("myCanvas").getContext('2d');
        var mCanvas = document.getElementById("myCanvas");
        ctx.clearRect(0, 0, mCanvas.width, mCanvas.height);
    }

</script>
```

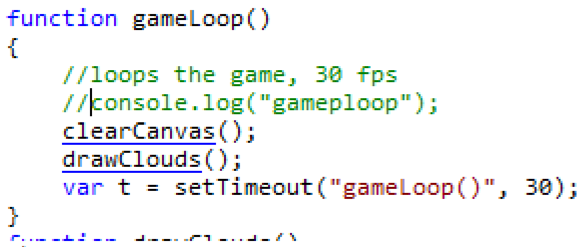
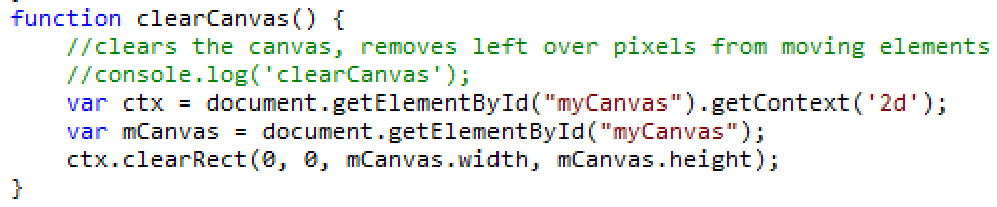
-
- Produces this



-
- The console should be producing the following output

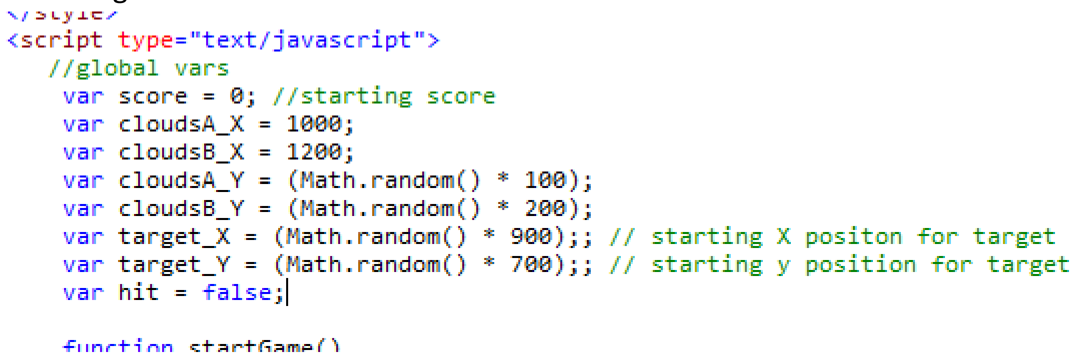
- 

```
gameploop
clearCanvas
gameploop
clearCanvas
gameploop
clearCanvas
gameploop
clearCanvas
gameploop
```

- We use the console to ensure that each function is working correctly. Now that we are sure that they are, we can clean that up. Put // in front of the console.log lines in gameloop and clearCanvas.
-  

```
function gameLoop()
{
    //loops the game, 30 fps
    //console.log("gameploop");
    clearCanvas();
    drawClouds();
    var t = setTimeout("gameLoop()", 30);
}
function drawClouds()
```

```
function clearCanvas() {
    //clears the canvas, removes left over pixels from moving elements
    //console.log('clearCanvas');
    var ctx = document.getElementById("myCanvas").getContext('2d');
    var mCanvas = document.getElementById("myCanvas");
    ctx.clearRect(0, 0, mCanvas.width, mCanvas.height);
}
```

- So this is our background with some moving elements. Now we need to include our target to get shot at it.
- Our target is going to change its location randomly when clicked on so we need to add some global vars for that.
- 

```
</style>
<script type="text/javascript">
   //global vars
    var score = 0; //starting score
    var cloudsA_X = 1000;
    var cloudsB_X = 1200;
    var cloudsA_Y = (Math.random() * 100);
    var cloudsB_Y = (Math.random() * 200);
    var target_X = (Math.random() * 900);; // starting X positon for target
    var target_Y = (Math.random() * 700);; // starting y position for target
    var hit = false;

    function startGame()
```

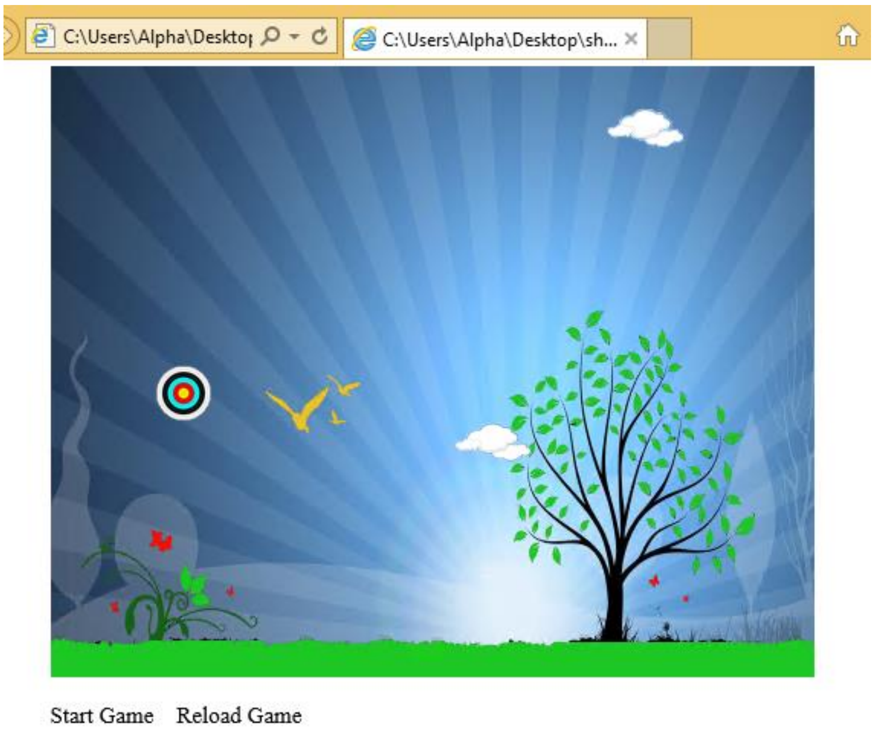
- The hit false is so that we automatically draw the target. First time round.
- So now we modify the gameLoop function

```
function gameLoop()
{
    //loops the game, 30 fps
    //console.log("gameploop");
    clearCanvas();
    drawClouds();
    if (hit == false) {
        drawTarget();
    }
    else {
        score++;
        target_X = (Math.random() * 900);; // new X pos
        target_Y = (Math.random() * 700);; // new Y pos
        drawTarget();
    }

    var t = setTimeout("gameLoop()", 30);
}
```

- And create the function drawTarget()

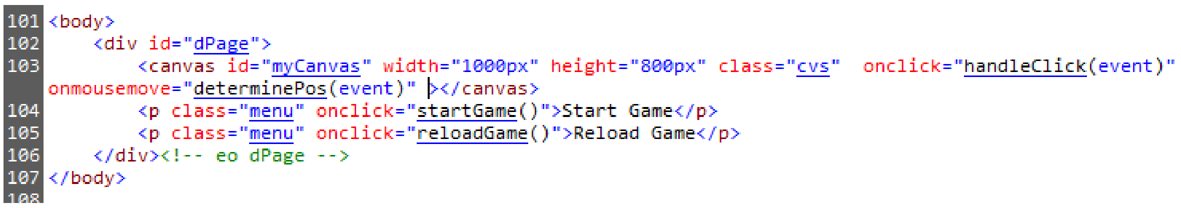
```
function drawTarget()
{
    //draws the target to be shot at
    var ctx = document.getElementById("myCanvas").getContext('2d');
    var target = new Image();
    target.src = "images/target.png";
    target.onload = function () {
        ctx.drawImage(target, target_X, target_Y);
    }
}
```

- Run this up and you should see something similar



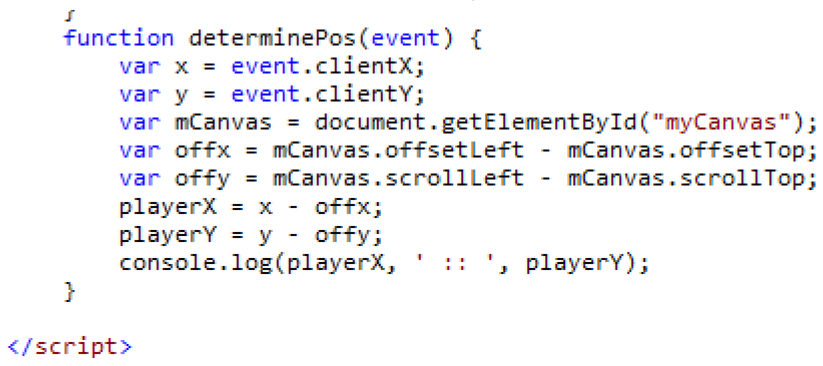
- This is all good, but there is no interactivity with it yet, so now we need to start dealing with interaction.

- To do this we add what we do with the mouse click and we can track the mouse movement as well, modify the canvas line



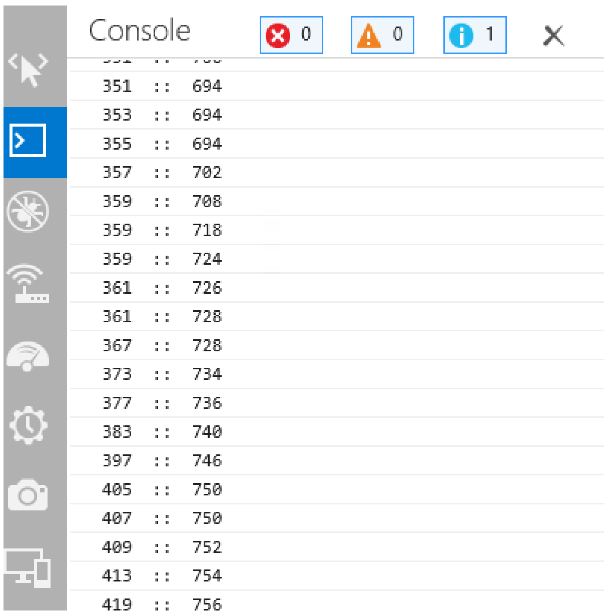
```
<body>
    <div id="dPage">
        <canvas id="myCanvas" width="1000px" height="800px" class="cvs"  onclick="handleClick(event)"
onmousemove="determinePos(event)" ></canvas>
        <p class="menu" onclick="startGame()">Start Game</p>
        <p class="menu" onclick="reloadGame()">Reload Game</p>
    </div><!-- eo dPage -->
</body>
```

- We now have 2 new functions to write, let's start with determinePos(event)



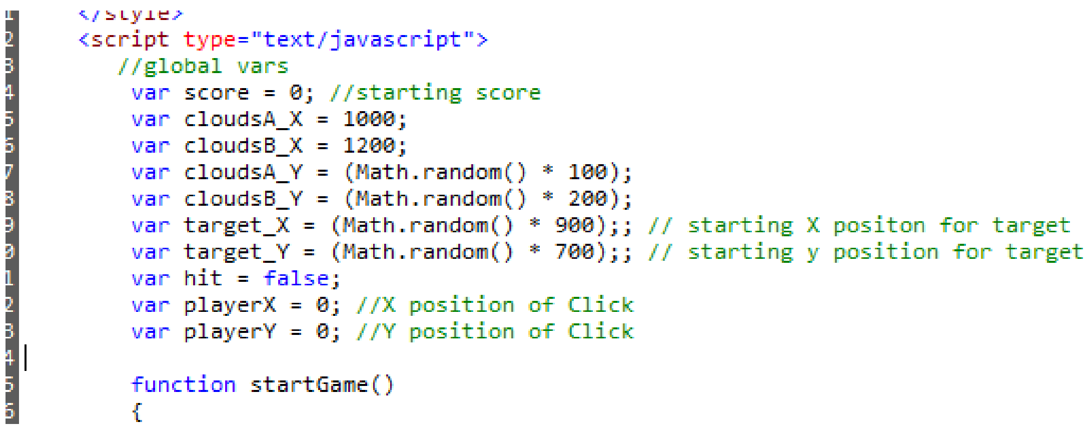
```
    function determinePos(event) {
        var x = event.clientX;
        var y = event.clientY;
        var mCanvas = document.getElementById("myCanvas");
        var offx = mCanvas.offsetLeft - mCanvas.offsetTop;
        var offy = mCanvas.scrollLeft - mCanvas.scrollTop;
        playerX = x - offx;
        playerY = y - offy;
        console.log(playerX, ' :: ', playerY);
    }

</script>
```

- Once it has been written, run it up and run your mouse over the canvas, the console should have a bunch of numbers in it like this

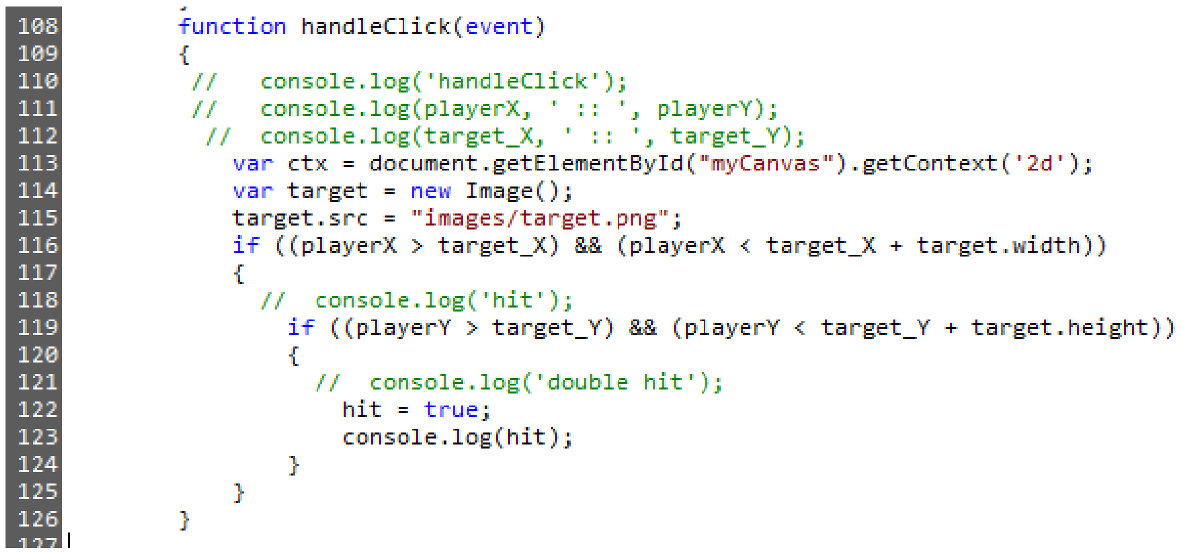


```
351  ::  694
353  ::  694
355  ::  694
357  ::  702
359  ::  708
359  ::  718
359  ::  724
361  ::  726
361  ::  728
367  ::  728
373  ::  734
377  ::  736
383  ::  740
397  ::  746
405  ::  750
407  ::  750
409  ::  752
413  ::  754
419  ::  756
```

- So, these numbers are the point where the cursor is whilst it is moving on the canvas. Now that we know that information we need to feed it into a global var, even though we used it in the function, without passing it to a global variable, we can't see it anywhere else so add 2 new global vars.

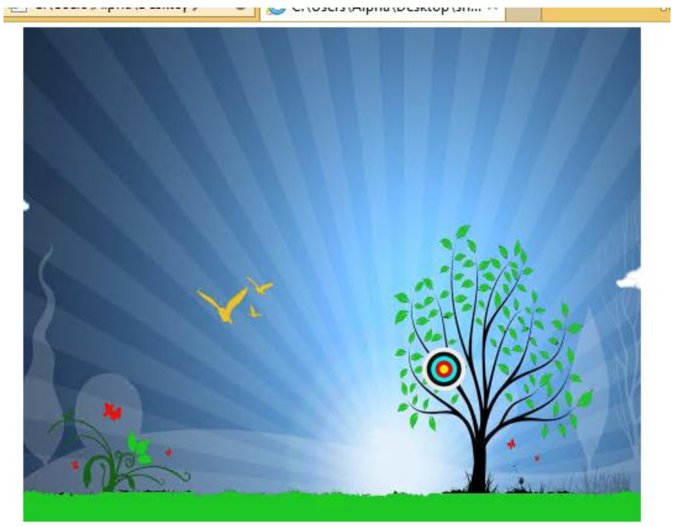
- 

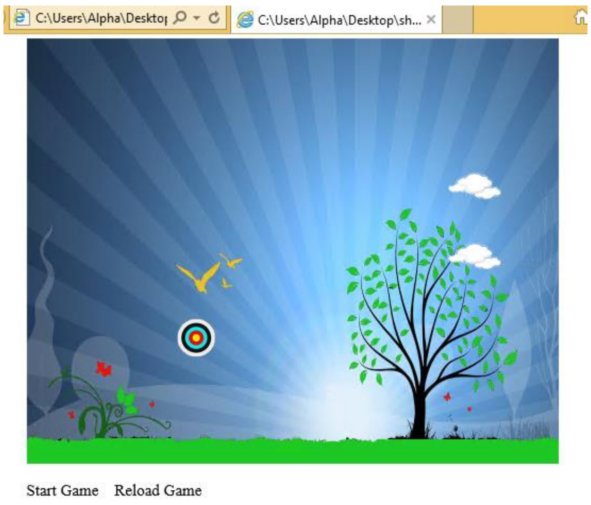
```
    </style>
    <script type="text/javascript">
       //global vars
        var score = 0; //starting score
        var cloudsA_X = 1000;
        var cloudsB_X = 1200;
        var cloudsA_Y = (Math.random() * 100);
        var cloudsB_Y = (Math.random() * 200);
        var target_X = (Math.random() * 900);; // starting X positon for target
        var target_Y = (Math.random() * 700);; // starting y position for target
        var hit = false;
        var playerX = 0; //X position of Click
        var playerY = 0; //Y position of Click

        function startGame()
        {
```

- What happens is whenever we move the mouse over the canvas, the numbers go into the new playerX and player vars. Of which we can then use to determine where our click to hit occurs.
- To dertime what happens when we click, now we write the handleClick(event) function.
- 

```
        function handleClick(event)
        {
         //   console.log('handleClick');
         //   console.log(playerX, ' :: ', playerY);
          //  console.log(target_X, ' :: ', target_Y);
            var ctx = document.getElementById("myCanvas").getContext('2d');
            var target = new Image();
            target.src = "images/target.png";
            if ((playerX > target_X) && (playerX < target_X + target.width))
            {
              //  console.log('hit');
                if ((playerY > target_Y) && (playerY < target_Y + target.height))
                {
                  //  console.log('double hit');
                    hit = true;
                    console.log(hit);
                }
            }
        }
```

- Notice how we have the global variable hit being modified. This means that when we do click on the target it updates to a new position
- Save and run it up. You get the following



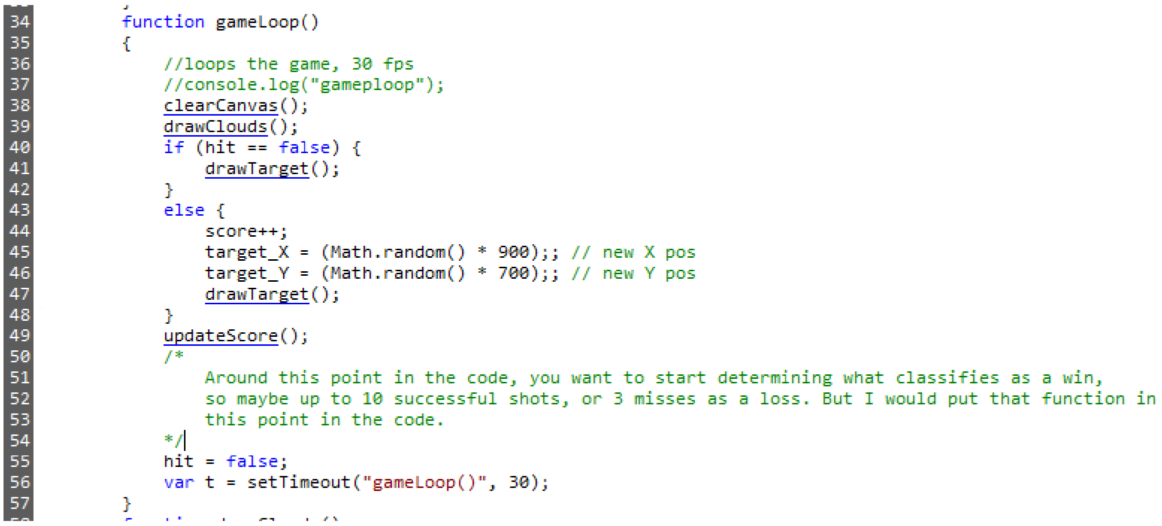
- 

- An updating target position, approximately 30 times a second. So now let’s deal with that.
- The Boolean that deals with telling the target to redraw is the hit variable. So when it gets clicked on it swaps to true and redraws. The drawback is that it never changes back to false. So the system maintains that it is true and must always redraw target. To fix this we put a hit = false after it has been redrawn. Tis occurs in the gameloop.

```
}
function gameLoop()
{
    //loops the game, 30 fps
    //console.log("gameploop");
    clearCanvas();
    drawClouds();
    if (hit == false) {
        drawTarget();
    }
    else {
        score++;
        target_X = (Math.random() * 900);; // new X pos
        target_Y = (Math.random() * 700);; // new Y pos
        drawTarget();
    }
    hit = false;
    var t = setTimeout("gameLoop()", 30);
}
```

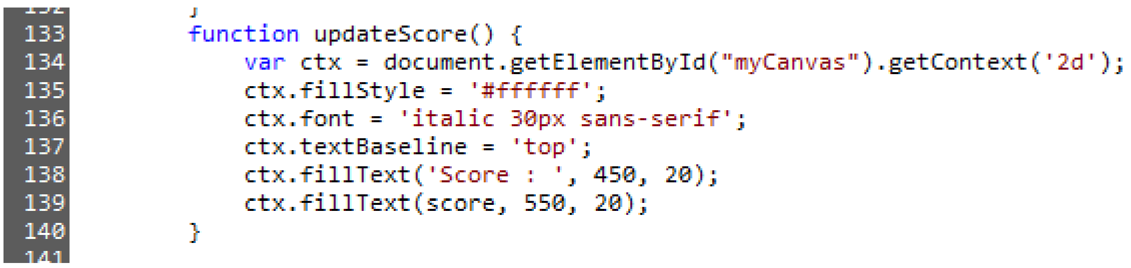
-

- From here, we add in a counter for each time the target gets hit.
- So, we'll add a bit more code to the gameLoop, you might have noticed the score++ var, and global var. These are for calculating the clicks, now we draw it out.



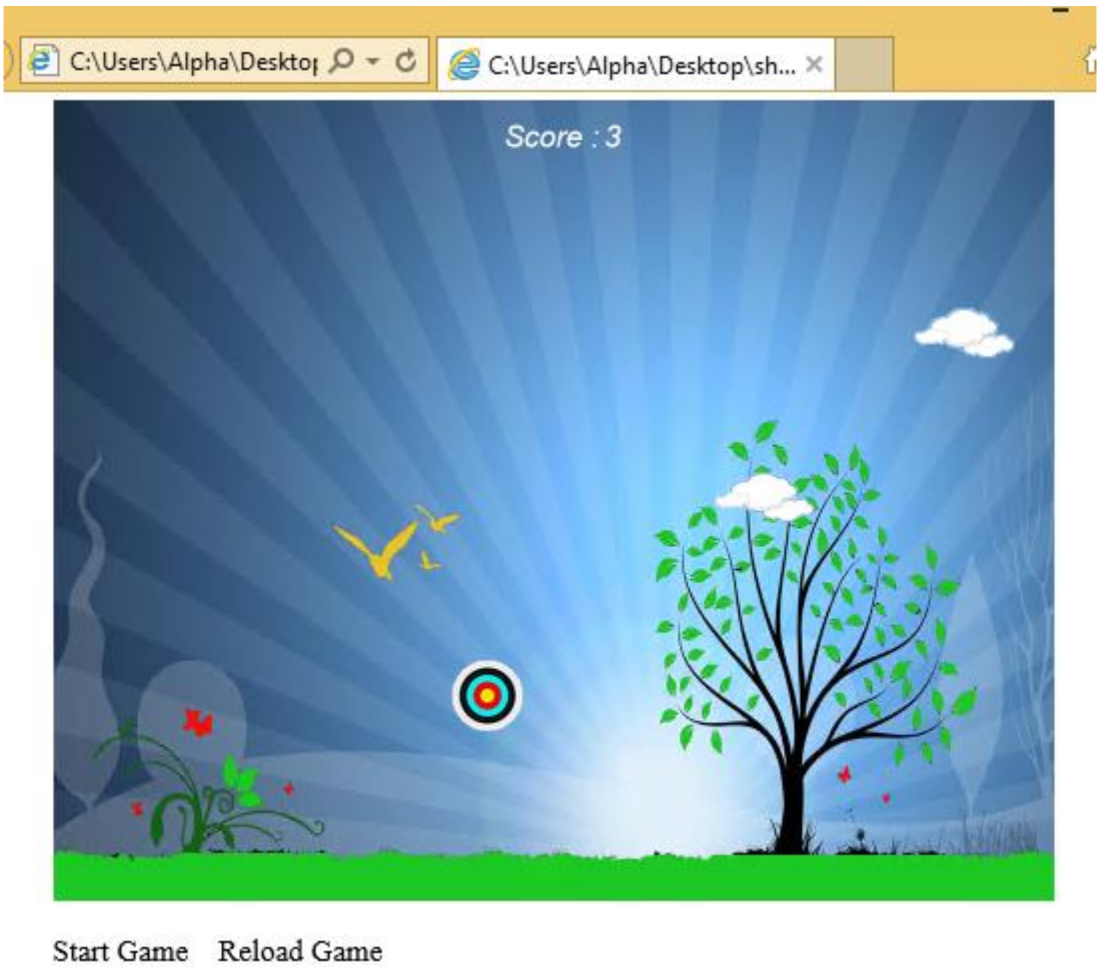
```
function gameLoop()
{
    //loops the game, 30 fps
    //console.log("gameploop");
    clearCanvas();
    drawClouds();
    if (hit == false) {
        drawTarget();
    }
    else {
        score++;
        target_X = (Math.random() * 900);; // new X pos
        target_Y = (Math.random() * 700);; // new Y pos
        drawTarget();
    }
    updateScore();
    /*
        Around this point in the code, you want to start determining what classifies as a win,
        so maybe up to 10 successful shots, or 3 misses as a loss. But I would put that function in
        this point in the code.
    */
    hit = false;
    var t = setTimeout("gameLoop()", 30);
}
```

- updateScore function



```
function updateScore() {
    var ctx = document.getElementById("myCanvas").getContext('2d');
    ctx.fillStyle = '#ffffff';
    ctx.font = 'italic 30px sans-serif';
    ctx.textBaseline = 'top';
    ctx.fillText('Score : ', 450, 20);
    ctx.fillText(score, 550, 20);
}
```

- Now run it up
- And click on the target a few times, you should see this


C:\Users\Alpha\Desktop
C:\Users\Alpha\Desktop\sh...
Score : 3
Start Game
Reload Game

-