Web Design Report

**Requirements:**

A 4-page website made up of a landing page, gallery page, a page for a JavaScript minigame and a contact page. The website is to be made for a Taxi Rental company.

**List of Members:**

Evan Keeley- Wrote the report and coded the contact page.

Benita Amaechi- Coded the landing page and gallery page.

Leon Montgomery- Coded the Java minigame page.

**Coding Challenge:**

Evan- A coding challenge I faced was adding a submit and reset button to the contact form. I solved this by using W3Schools to find examples of forms with a submit and reset button and their code. I then changed the code used to whatever suited the form the most.

Benita- A coding challenge I faced was deciding which order to put the web pages in to make the website look more attractive. I resolved this by getting another person's opinion in order to decide with more than just one pair of eyes.

Leon- A coding challenge I faced was figuring out what mistakes were made in the code and how to fix them whenever the web page was not responsive. I resolved this by looking up videos on YouTube for tips and examples of Java code.

**Teamwork:**

As a team we worked very well. We split up our work as fairly and evenly as possible. We shared information on our separate jobs to ensure that we were all working to the same level. We also made sure that everyone had completed their work and tried to keep each other updated on our progress.

# Get to the other side – Taxi Game.

Using Squares and circles I put then on a canvas to represent a person trying to cross the road with various obstacles.

The method used was to break the elements down into component parts. The circle represented the person crossing the road and the different rectangles represented to the traffic in the road.

Using Javascript I was able to capture the key-press from the user and with this was able to move the circle on the screen when the user used the arrow keys as they were passed to the function checkKey(e). The If else if statement used the keycodes to change the x,y co -ordinates accordingly.

document.onkeydown = checkKey;

/ Get a key press from player to move

function checkKey(e) {

if (e.keyCode == '38') {

// up arrow

yf = yf-15

}

else if (e.keyCode == '40') { ---------- logic is the same for the other key codes.

To get the obstacles drawn on the canvas I went through several iterations but could not get the while loop to work without crashing the page. My idea was to populate an array and then change the x,y components for each element in the array. I could not navigate the timing as to when to update the array and the calling of the draw objects function.

Calling functions to draw the obstacles was relatively simple using in built functions of beginPath(), arc() and rect()

|  |  |
| --- | --- |
| function drawBall() {  ctx.beginPath();  ctx.arc(xf, yf, 10, 0, Math.PI\*2);  ctx.fillStyle = "#0095DD";  ctx.fill();  ctx.closePath();} | function drawTruck() {  ctx.beginPath();  ctx.rect(xT, 20, 90, 50);  ctx.fillStyle = "#000000";  ctx.fill();  ctx.closePath();} |

I was able to place the obstacles on the canvas but not able to create a loop as I continued to get a frozen page. I presume this was due a conflict of drawing the page and updating the new variables.

These conflicts were probably due to not getting the DOM event listeners to work correctly and not quite understanding the timing of the draw function.

I was happy with the array work and getting the for loop to generate vehicles for the program. The types and the lookup was frustrating but there was a little randomness introduced in the colour scheme and position of the obstacles.

|  |  |
| --- | --- |
| var vehicles = [];  var vTypes = ['Car', 'Truck', 'Bus'];  var vType = vTypes[Math.floor(Math.random() \* vTypes.length)];  function varray() {  for (var i = 0; i < 11; i++) {  if (i < 3) {  vehicles[i] = new Vehicle(x,yl-(i\*40),vType);  } else if (i < 6) {  vehicles[i] = new Vehicle((x-50),yl-(i\*60),vType);  } else if (i < 9) {  vehicles[i] = new Vehicle((x-100),yl-(i\*60),vType);  } else {  vehicles[i] = new Vehicle((x-150),yl-(i\*80),vType);  }} } | function Vehicle(x,y,vType){  this.x = x;  this.y = y;    if (vType == 'Car') {  this.display = function(){  ctx.rect(this.x,this.y,50,50);  ctx.fillStyle = "#009c3b";  ctx.fill();  }  } else if (vType == 'Truck') {  this.display = function(){  ctx.rect(this.x,this.y,80,50);  ctx.fillStyle = "#ffdf00";  ctx.fill();  }  } else {  this.display = function(){  ctx.rect(this.x,this.y,90,50);  ctx.fillStyle = "#000000";  ctx.fill();  }    }  this.move = function(){  this.x = this.x + 2;  this.y = this.y + 0;  }  }  window.addEventListener('load', function () {  //alert("It's loaded!")  varray();  }) |

The draw() function calls on the work previously done by calling the functions and is able to loop through the vehicles array and use its methods to move and update the display. It does this initially on the screen just to clear the canvas.

function draw() {

ctx.clearRect(0, 0, canvas.width, canvas.height);

drawBall();

drawTruck();

drawBus();

drawCar();

x += dx;

xT += dx;

y += dy;

for (var i = 0; i < vehicles.length; i++) {

vehicles[i].move();

vehicles[i].display();

}}

//canvas.addEventListener("keypress", moveF(e));

setInterval(draw, 10);

# Javascript implementation

Functionality of game:

|  |  |
| --- | --- |
| Movement of the icon/person with arrow keys | Good functionality could have put a variable for speed variation on levels. |
| Create an obstacle park with a traffic | Able to fill screen but unable to get the while loop to work. |
| Create a scoreboard and reward/level change | Not able to get there but would have implemented the score board with an event listener on the player icon when it passes a certain y coordinate |
| Game over and crash audio files not there | Did not get to looking at the audio and graphics. |
| Game to divert the customer while he waits for a cab. There was an idea to have a status bar using the taxi logo to show wait time | The status bar is there on top of the page – just the loop to run out a count and update it on a timer. |