**Part 2:**

HTML5

**CANVAS**

Animation & app initialisation

//how to set up for animation

Function initCanvas() {

Var ctx = document.getElementById(‘canvas’).getContext(‘2d’);

Var cW = ctx.canvas.width;

Var cH = ctx.canvas.height;

Var y = 0;

Function animate() {  
 ctx.save(); //saves canvas

Ctx.clearRect(0,0,cW,cH); //erase canvas

//draw here

Ctx.fillRect(0,y,50,50);

Y++;

Ctx.restore(); //restores the most recently saved canvas

}

Var animateInterval = setInterval(animate, 30); //30ms

Ctx.canvas.addEventListener(‘click’, function(e){

clearInterval(animateInterval);

}); //when user clicks on canvas, animate function stops

}

Window.addEventListener(‘load’, function(e){

initCanvas()

});

Object oriented assets

You can also create elements on the canvas using OOP as demonstrated below:

//constructor

Function RectObj() {

This.x = 0;

This.y = 0;

This.render = function(ctx, x, y, w, h, clr) {

Ctx.fillStyle = clr;

Ctx.fillRect(x,y,w,h);

}

}

Var rect1 = new RectObj();

Var rect2 = new RectObj();

Rect2.y = 30; //set new y property away from default (0)

//then, in animate function

Rect1.render(ctx, rect1.x, rect1.y, 50,50,”blue”);

Rect1.x++; //to move object

Mouse coordinates

Get mouse coordinates when in the canvas

Ctx.canvas.addEventListener(‘mousemove’, function(e) {  
 var mouseX = e.clientX – ctx.canvas.offsetLeft;

Var mouseY = e.clientY – ctx.canvas.offsetTop;

Var status = document.getElementById(‘status’); //div tag ID

Status.innerHTML = mouseX + “ | “ + mouseY;

})

//create rectangle when canvas clicked on in the location of mouse

Ctx.canvas.addEventListener(‘click’, function(e) {

var mouseX = e.clientX – ctx.canvas.offsetLeft;

Var mouseY = e.clientY – ctx.canvas.offsetTop;

Ctx.fillStyle = “orange”;

Ctx.fillRect(mouseX-15, mouseY-15, 30, 30); //in order to paste rectangle from center

});

Animated background layers

//set up script correctly (like under the app initialisation heading)

//set canvas width to 1000 and height to 500

//create photoshop image width 1500, height 500 and export (watch Youtube tutorial for more information on this)

Var bg = new Image();

Bg.src = “stars.jpg”;

Function initCanvas() {  
 //see proper set up of this method

Function Background() {  
 this.x = 0;

This.y = 0;

This.w = bg.width;

This.h = bg.height;

This.render = function() {  
 ctx.drawImage(bg, this.x--, 0);

If(this.x <= 499) {

This.x = 0;

}

}

}

Var background = new Background();

Function animate() {

Ctx.save();

Background.render();

Ctx.restore();

}

//call animate using setInterval function

}

Keyboard control & movement

Var dist = 3; //initiate number of pixels for rect to move by

//using previous example – under background constructor

Function Player() {  
 this.x = 0;

This.y = 0;

This.w = 50;

This.h = 50;

This.render = function() {  
 ctx.fillStyle = “Orange”;

Ctx.fillRect(this.x, this.y, this.w, this.h);

}

}

Var player new Player();

Player.x = 100; //set default x position

Player.y = 225; //set default y position

//place player.render() under background.render()

//after animate interval

Document.addEventListener(‘keydown’, function(e) {

Var key\_press = string.fromCharCode(e.keyCode);

If(key\_press === “w”) {

Player.y -= dist; //dist variable initiated in first line above

}else if { //do same for ‘a’, ‘s’, ‘d’ – make sure to adjust to correct coordinates

JSON, object, arrays & loops

Function initCanvas(){  
 var ctx = document.getElementById(‘canvas’).getContext(‘2d’);

Var buildings = [ {“id”: “house”, “x”: 100, “y”: 100, “w”: 50, “h”: 50, “bg”:”red”},

{“id”: “post\_office”, “x”: 300, “y”: 100, “w”: 50, “h”: 50, “bg”:”blue”}

];

For(var i=0; i<buildings.length; i++) {

Var b = buildings[i];

Ctx.fillStyle = b.bg;

Ctx.fillRect(b.x, b.y, b.w, b.h);

}

}

Window.addEventListener(‘load’, initCanvas);

Hit detection mouse touch over

//using the above JSON example, in the initCanvas function add the below code

Ctx.canvas.addEventListener(‘mousedown’, function(e) {

Var mX = e.clientX – ctx.canvas.offsetLeft;

Var mY = e.clientY – ctx.canvas.offsetTop;

For(var i=0; i<buildings.length; i++) {  
 var b = buildings[i];

If(mX >= b.x && mX < b.x+b.w && mY >= b.y && mY < b.y+b.h) {

Document.getElementById(‘status’).innerHTML = “You touched “ + b.id;

}

}

}

Collision detection

Function initCanvas() {  
 var ctx = document.getElementById(‘canvas’).getContext(‘2d’);

Var cW = ctx.canvas.width;

Var cH = ctx.canvas.height;

Var enemies = [ {“id”: “enemy1”, “x”: 100, “y”: -20, “w”: 40, “h”:20},

{“id”: “enemy2”, “x”: 225, “y”: -20, “w”: 40, “h”:20},

{“id”: “enemy3”, “x”: 350, “y”: -20, “w”: 40, “h”:20},

{“id”: “enemy4”, “x”: 100, “y”: -70, “w”: 40, “h”:20},

{“id”: “enemy5”, “x”: 225, “y”: -70, “w”: 40, “h”:20},

{“id”: “enemy6”, “x”: 350, “y”: -70, “w”: 40, “h”:20}

]

Function renderEnemies() {

For(var i=0; i<enemies.length; i++) {

Ctx.fillStyle = “blue”;

Ctx.fillRect(enemies[i].x, enemies[i].y +=5, enemies[i].w, enemies[i].h);  
 }

}

Function Launcher() {  
 this.y = 280, this.x = cW\*.5 – 25, this.w = 50, this.h = 50, this.dir, this.bg = “orange”, this.missiles = [];

This.render = function() {  
 if(this.dir === “left”) {

This.x -= 5;

}else if(this.dir === “right”){

This.x += 5;

}

Ctx.fillStyle = this.bg;

Ctx.fillRect(this.x,this.y,this.w,this.h);

For(var i=0; i<this.missiles.length; i++) {  
 var m = this.missiles[i];

Ctx.fillStyle = m.bg;

Ctx.fillRect(m.x, m.y -=5, m.w, m.h);

This.hitDetect(this.missiles[i], i);

If(m.y <= 0) { //if missiles goes off canvas

This.missiles.splice(i,1);

}

}

If(enemies.length === 0) {

clearInterval(animateInterval); // stop animation

ctx.fillStyle = ‘#FC0’;

ctx.font = “italic bold 36px Arial, sans-serif”;

ctx.fillText(‘Level Complete’, cW\*.5 – 130, 50, 300);

}

} //end of render method

This.hitDetect = function(m, mi) {

For(var i=0; i<enemies.length; i++) {

Var e = enemies[i];

If(m.x + m.w >= e.x && m.x <= e.x + e.w && m.y >= e.y && m.y <= e.y + e.h) {

This.missiles.splice(this.missiles[mi], 1);

Enemies.splice(i,1);

Document.getElementById(‘status’).innerHTML = “You destroyed ” +e.id;

}

}

}

}//end of Launcher

Var launcher = new Launcher();

Function animate() {  
 ctx.clearRect(0,0,cW, cH);

launcher.render();

renderEnemies();

}

Var animateInterval = setInterval(animate, 30);

//have below buttons in DOM

Left\_btn.addEventListener(‘mousedown’, function(e) {

launcher.dir = “left”;

});

Left\_btn.addEventListener(‘mouseup’, function(e) {

launcher.dir = “”;

});

//do same for right\_btn

Fire\_btn.addEventListener(‘mousedown’, function(e) {

launcher.missiles.push( {“x”: launcher.x + launcher.w \*.5, “y”: launcher.y, “w”: 3, “h”: 10, “bg”: “red”} );

});

}//end of initCanvas

Window.addEventListener(‘load’, initCanvas);

Particle effect – snow tutorial

Var bg = new Image();

Bg.src = “snow\_background.jpg”;

Function initCanvas() {

Var ctx = document.getElementById(‘canvas’).getContext(‘2d’);

Var cW = ctx.canvas.width;

Var cH = ctx.canvas.height;

Var flakes = [];

Function addFlake() {

Var x = Math.floor(Math.random() \* cW) + 1;

Var y = 0;

Var s = Math.floor(Math.random() \* 3) + 1; //speed/size

Flakes.push( { “x”:x, “y”: y, “s”: s } );

}

Function snow() {

addFlake();

for(var i=0; i<flakes.length; i++) {  
 ctx.fillStyle = “rgba(255,255,255,.75)”;

ctx.beginPath();

ctx.arc(flakes[i].x, flakes[i].y += flakes[i].s \*.5, flakes[i].s\*.5, 0, Math.PI\*2, false);

ctx.fill();

if(flakes[i].y > cH) {

flakes.splice(i,1);

}

Document.getElementById(‘status’).innerHTML = “Snow flakes count: “ + flakes.length;

}

}

Function animate() {  
 ctx.clearRect(0,0,cW,cH);

Ctx.drawImage(bg, 0, 0);

Snow(); //this can be invoked numerous times for more snow!

}

Var animateInterval = setInterval(animate, 30);

}

Window.addEventListener(‘load’, initCanvas);