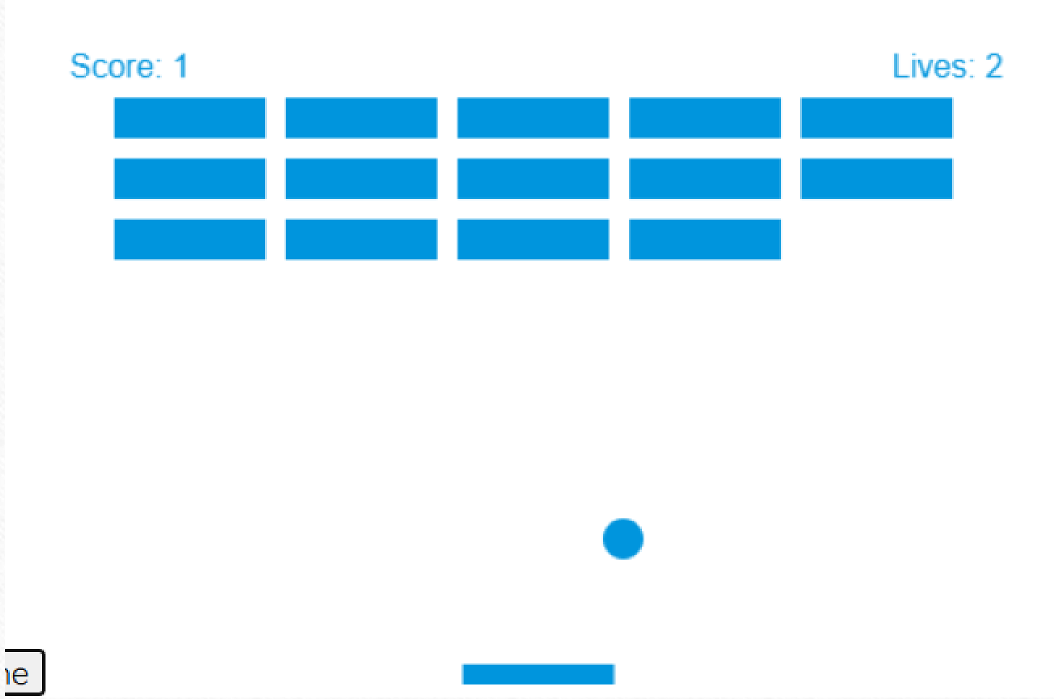
網際網路管理

期末報告

40823238 吳容輝

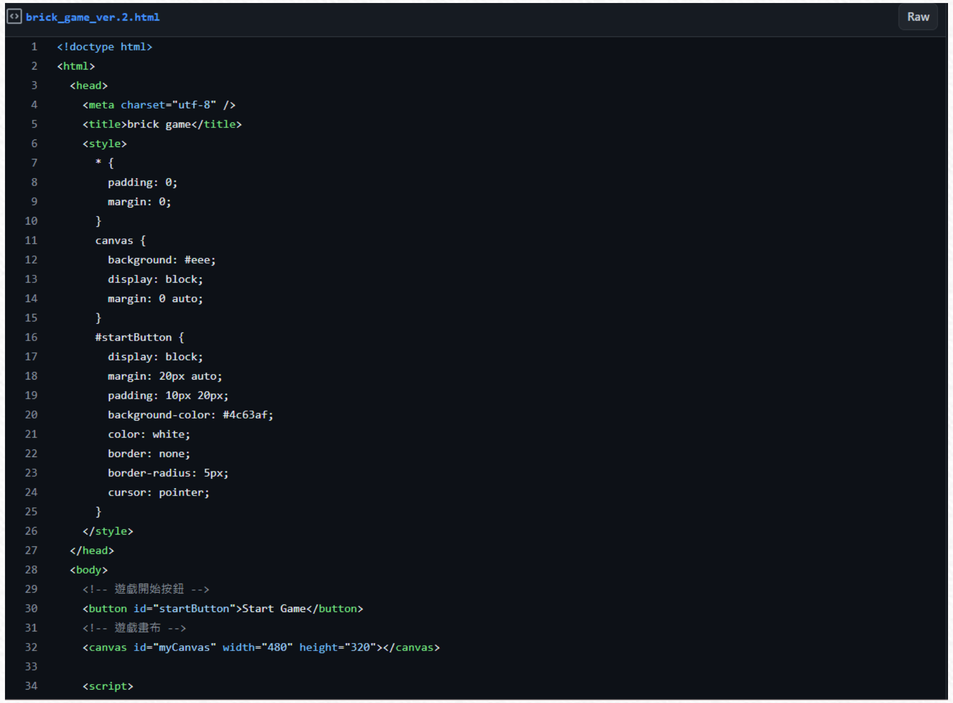
打磚塊遊戲



設計概要

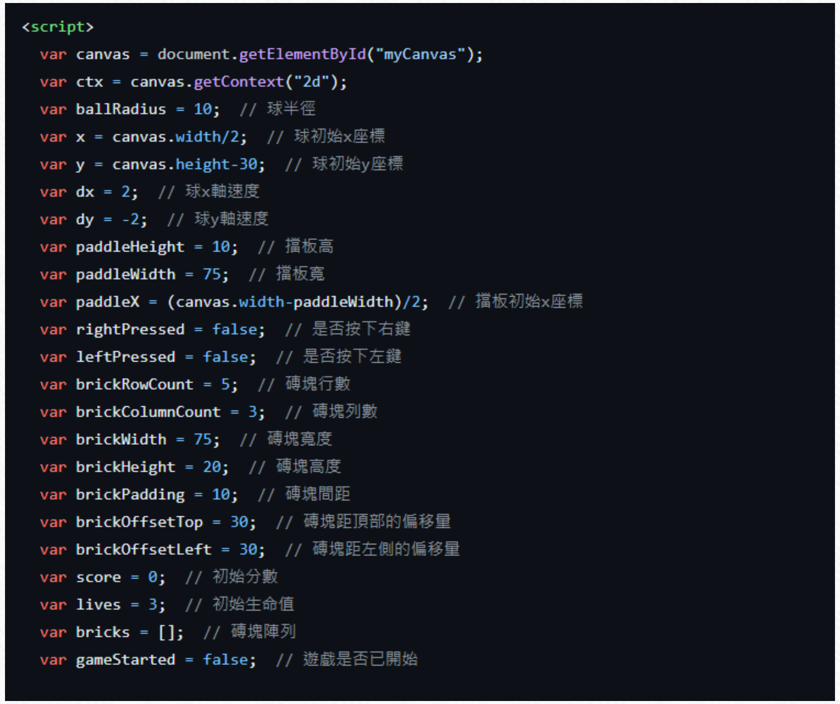
- 設計一個簡單的打磚塊遊戲，透過計分與生命值的設計，讓遊戲更有趣

程式碼(html 部分)



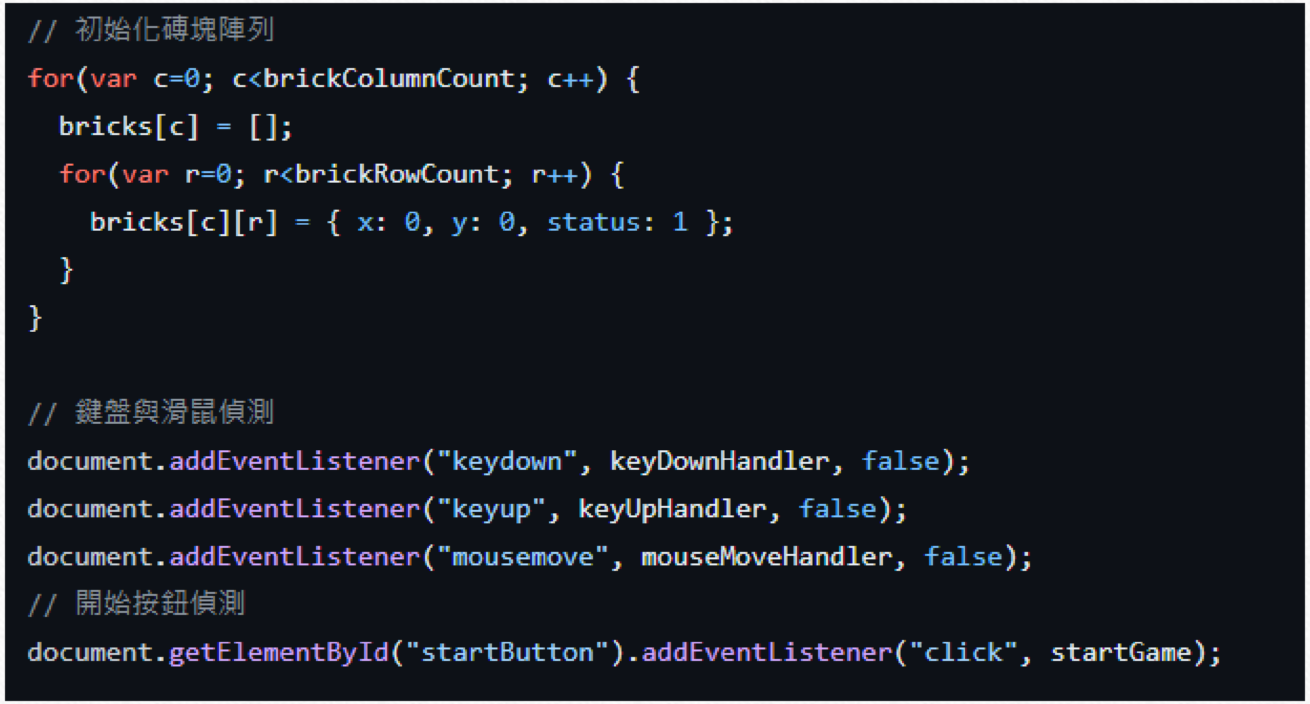
```html
<!doctype html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>brick game</title>
    <style>
      * {
        padding: 0;
        margin: 0;
      }
      canvas {
        background: #eee;
        display: block;
        margin: 0 auto;
      }
      #startButton {
        display: block;
        margin: 20px auto;
        padding: 10px 20px;
        background-color: #4c63af;
        color: white;
        border: none;
        border-radius: 5px;
        cursor: pointer;
      }
    </style>
  </head>
  <body>
    <!-- 遊戲開始按鈕 -->
    <button id="startButton">Start Game</button>
    <!-- 遊戲畫布 -->
    <canvas id="myCanvas" width="480" height="320"></canvas>

    <script>
```

Javascript 部分(宣告)



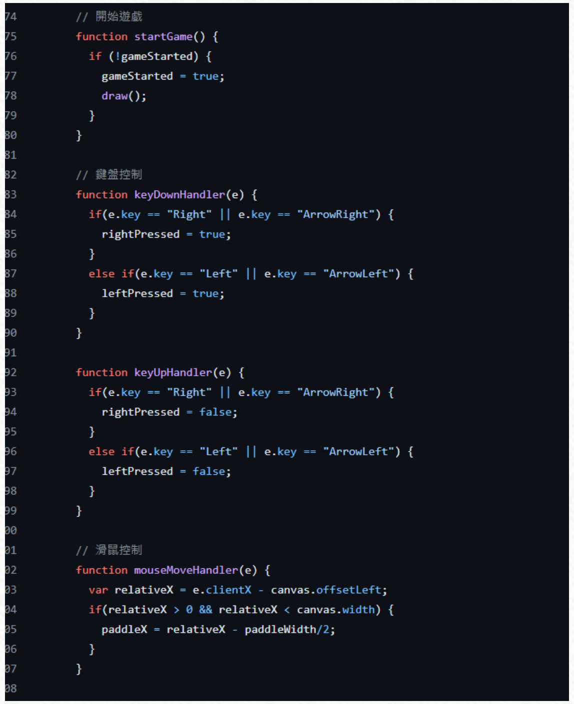
```
<script>
  var canvas = document.getElementById("myCanvas");
  var ctx = canvas.getContext("2d");
  var ballRadius = 10;  // 球半徑
  var x = canvas.width/2;  // 球初始x座標
  var y = canvas.height-30;  // 球初始y座標
  var dx = 2;  // 球x軸速度
  var dy = -2;  // 球y軸速度
  var paddleHeight = 10;  // 擋板高
  var paddleWidth = 75;  // 擋板寬
  var paddleX = (canvas.width-paddleWidth)/2;  // 擋板初始x座標
  var rightPressed = false;  // 是否按下右鍵
  var leftPressed = false;  // 是否按下左鍵
  var brickRowCount = 5;  // 磚塊行數
  var brickColumnCount = 3;  // 磚塊列數
  var brickWidth = 75;  // 磚塊寬度
  var brickHeight = 20;  // 磚塊高度
  var brickPadding = 10;  // 磚塊間距
  var brickOffsetTop = 30;  // 磚塊距頂部的偏移量
  var brickOffsetLeft = 30;  // 磚塊距左側的偏移量
  var score = 0;  // 初始分數
  var lives = 3;  // 初始生命值
  var bricks = [];  // 磚塊陣列
  var gameStarted = false;  // 遊戲是否已開始
```

初始化磚塊陣列 與 輸入偵測(鍵盤與滑鼠)



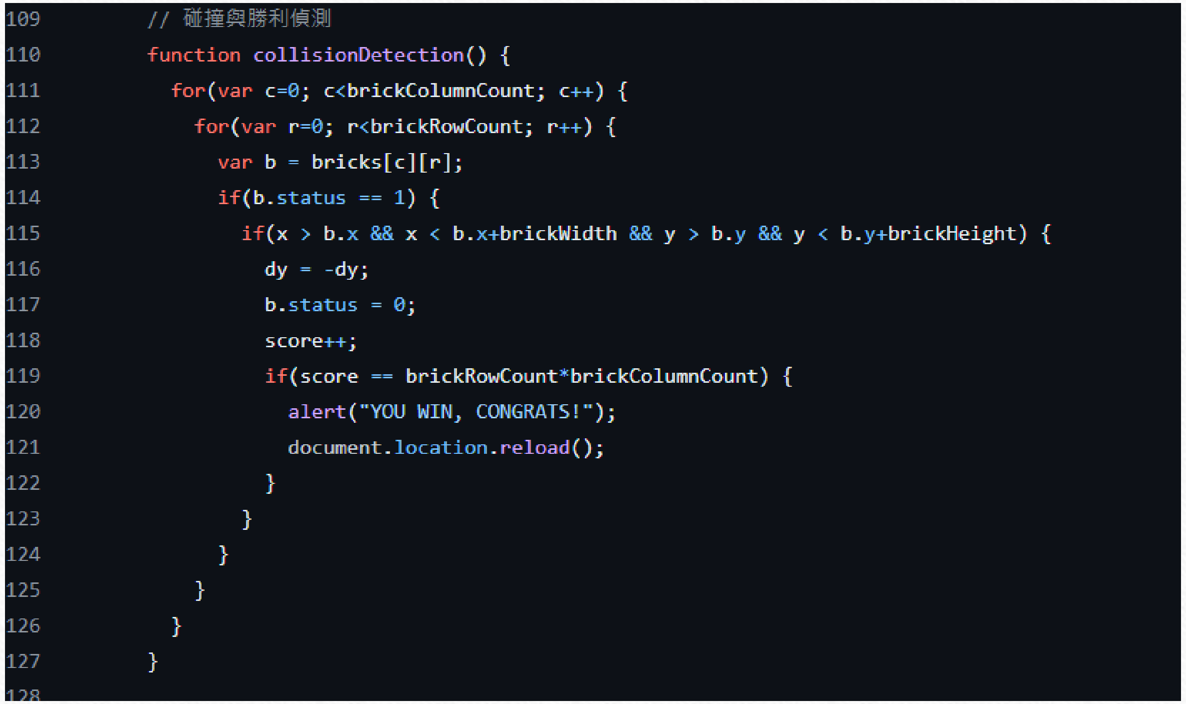
```
// 初始化磚塊陣列
for(var c=0; c<brickColumnCount; c++) {
  bricks[c] = [];
  for(var r=0; r<brickRowCount; r++) {
    bricks[c][r] = { x: 0, y: 0, status: 1 };
  }
}

// 鍵盤與滑鼠偵測
document.addEventListener("keydown", keyDownHandler, false);
document.addEventListener("keyup", keyUpHandler, false);
document.addEventListener("mousemove", mouseMoveHandler, false);
// 開始按鈕偵測
document.getElementById("startButton").addEventListener("click", startGame);
```

開始遊戲 與 輸入控制(鍵盤與滑鼠)



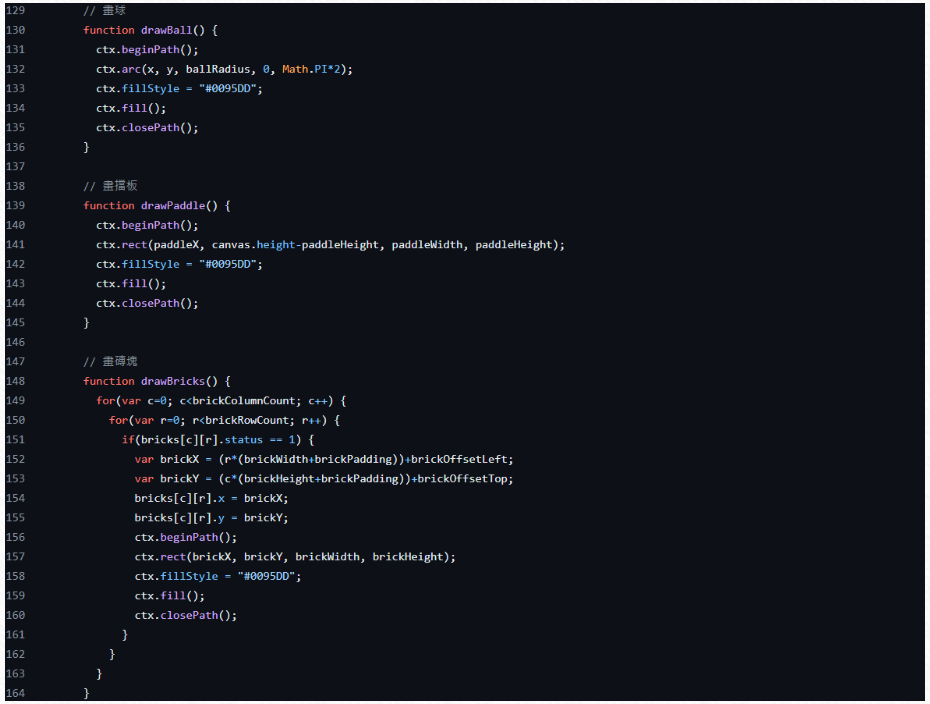
```
// 開始遊戲
function startGame() {
  if (!gameStarted) {
    gameStarted = true;
    draw();
  }
}

// 鍵盤控制
function keyDownHandler(e) {
  if(e.key == "Right" || e.key == "ArrowRight") {
    rightPressed = true;
  }
  else if(e.key == "Left" || e.key == "ArrowLeft") {
    leftPressed = true;
  }
}

function keyUpHandler(e) {
  if(e.key == "Right" || e.key == "ArrowRight") {
    rightPressed = false;
  }
  else if(e.key == "Left" || e.key == "ArrowLeft") {
    leftPressed = false;
  }
}

// 滑鼠控制
function mouseMoveHandler(e) {
  var relativeX = e.clientX - canvas.offsetLeft;
  if(relativeX > 0 && relativeX < canvas.width) {
    paddleX = relativeX - paddleWidth/2;
  }
}
```

碰撞 與 勝利條件 偵測



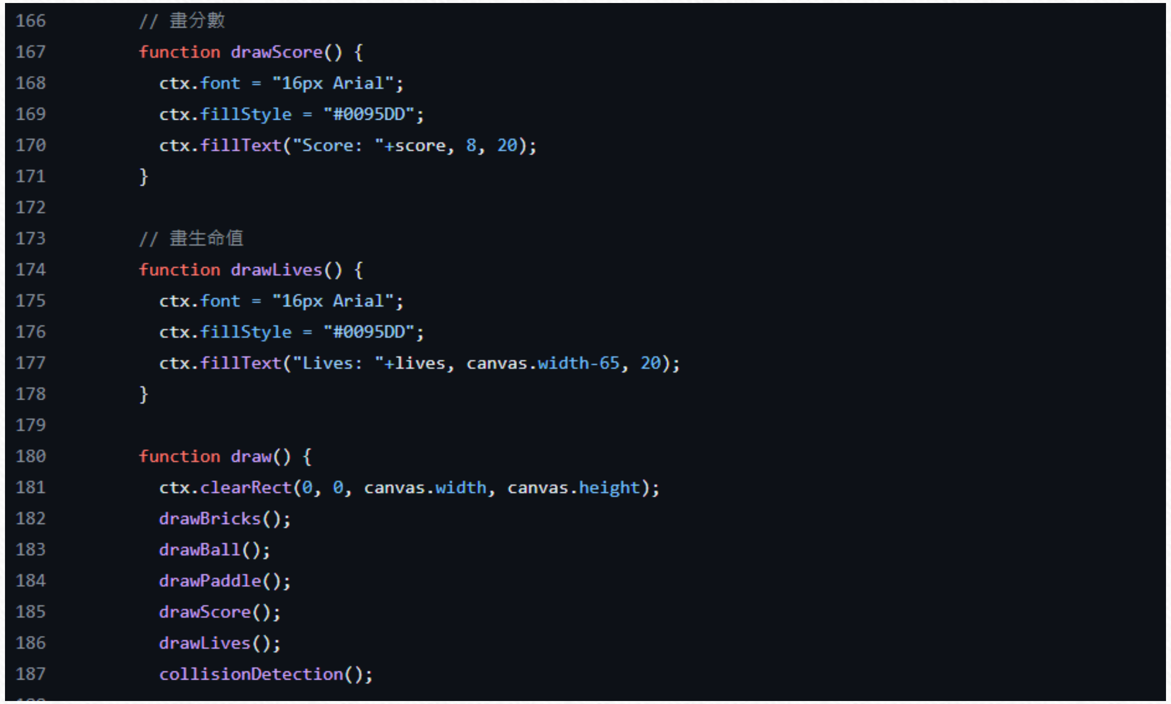
```
        // 碰撞與勝利偵測
        function collisionDetection() {
          for(var c=0; c<brickColumnCount; c++) {
            for(var r=0; r<brickRowCount; r++) {
              var b = bricks[c][r];
              if(b.status == 1) {
                if(x > b.x && x < b.x+brickWidth && y > b.y && y < b.y+brickHeight) {
                  dy = -dy;
                  b.status = 0;
                  score++;
                  if(score == brickRowCount*brickColumnCount) {
                    alert("YOU WIN, CONGRATS!");
                    document.location.reload();
                  }
                }
              }
            }
          }
        }
```

繪製遊戲要素(1)



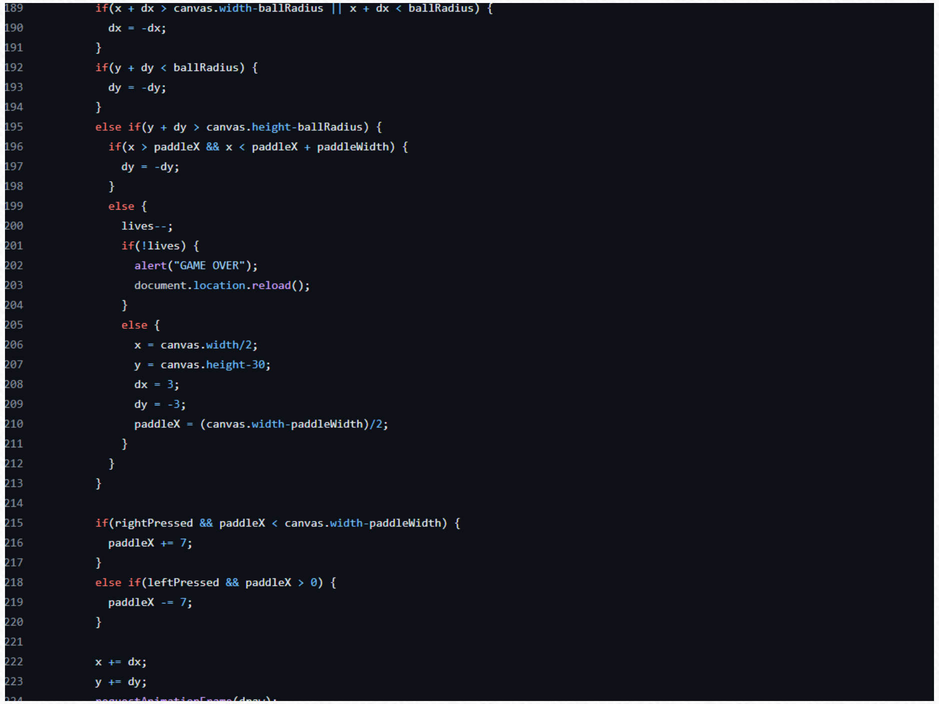
```
// 畫球
function drawBall() {
  ctx.beginPath();
  ctx.arc(x, y, ballRadius, 0, Math.PI*2);
  ctx.fillStyle = "#0095DD";
  ctx.fill();
  ctx.closePath();
}

// 畫擋板
function drawPaddle() {
  ctx.beginPath();
  ctx.rect(paddleX, canvas.height-paddleHeight, paddleWidth, paddleHeight);
  ctx.fillStyle = "#0095DD";
  ctx.fill();
  ctx.closePath();
}

// 畫磚塊
function drawBricks() {
  for(var c=0; c<brickColumnCount; c++) {
    for(var r=0; r<brickRowCount; r++) {
      if(bricks[c][r].status == 1) {
        var brickX = (r*(brickWidth+brickPadding))+brickOffsetLeft;
        var brickY = (c*(brickHeight+brickPadding))+brickOffsetTop;
        bricks[c][r].x = brickX;
        bricks[c][r].y = brickY;
        ctx.beginPath();
        ctx.rect(brickX, brickY, brickWidth, brickHeight);
        ctx.fillStyle = "#0095DD";
        ctx.fill();
        ctx.closePath();
      }
    }
  }
}
```

繪製遊戲要素(2)



```
        // 畫分數
        function drawScore() {
          ctx.font = "16px Arial";
          ctx.fillStyle = "#0095DD";
          ctx.fillText("Score: "+score, 8, 20);
        }

        // 畫生命值
        function drawLives() {
          ctx.font = "16px Arial";
          ctx.fillStyle = "#0095DD";
          ctx.fillText("Lives: "+lives, canvas.width-65, 20);
        }

        function draw() {
          ctx.clearRect(0, 0, canvas.width, canvas.height);
          drawBricks();
          drawBall();
          drawPaddle();
          drawScore();
          drawLives();
          collisionDetection();
```

遊戲結束 偵測



```
    if(x + dx > canvas.width-ballRadius || x + dx < ballRadius) {
      dx = -dx;
    }
    if(y + dy < ballRadius) {
      dy = -dy;
    }
    else if(y + dy > canvas.height-ballRadius) {
      if(x > paddleX && x < paddleX + paddleWidth) {
        dy = -dy;
      }
      else {
        lives--;
        if(!lives) {
          alert("GAME OVER");
          document.location.reload();
        }
        else {
          x = canvas.width/2;
          y = canvas.height-30;
          dx = 3;
          dy = -3;
          paddleX = (canvas.width-paddleWidth)/2;
        }
      }
    }

    if(rightPressed && paddleX < canvas.width-paddleWidth) {
      paddleX += 7;
    }
    else if(leftPressed && paddleX > 0) {
      paddleX -= 7;
    }

    x += dx;
    y += dy;
    requestAnimationFrame(draw);
```