JavaScript Term Project：小貓上下樓梯

0513404 姜林寬

遊戲主畫面

小貓吃蘋果

左上角顯示分數，右上角分別有小貓的血量(Life)、跳躍用的能量(Power)、座標位置(cat_x,cat_y)。
遊戲畫面分為左右兩塊，右邊的台階會往下移動，左邊的台階會往上移動，小貓會從中間的起始平台開始，選擇合適的時機跳下便開始遊戲。撞到天花板的尖刺減**3**生命，掉到遊戲畫面底部直接出局。
左右兩區各有三顆蘋果可
以吃，吃蘋果有益身體健康，可以補血補能量。

幫助老師了解的事

若此報告排版跑掉，老師可以到以下網址觀看。

https://hackmd.io/Rrh4DJ62TFKWE6GTo-71AQ?view (https://hackmd.io/Rrh4DJ62TFKWE6GTo-71AQ?view)

遊戲是從離開初始平台才算開始，所以不能在一開始的平台使用跳，遊戲開始後還是可以跳回一開始的平台。因為此遊戲是我上網找別人提供的圖片素材加上自己手刻的canvas，重力加速度的部分實現上較為困難。

系統功能

- 基本操作：
  - 左　右 移動， 上 消耗能量跳起來
  - 基本血量 10
  - 基本能量 10
  - 基礎體重 37
- 結束條件：
  - 血量歸0
  - 掉到畫面底部
- 多種類平面：
  - 尖刺平面：站上去減**3**生命

    - 普通平面(藍)：站上去加**1**生命
    - 能量平面(棕)：站上去加**2**能量
    - 左輸送帶：站上去會被向左運輸
    - 右輸送帶：站士去會被向右運輸
    - 彈跳床：站上去的期間會不斷彈跳

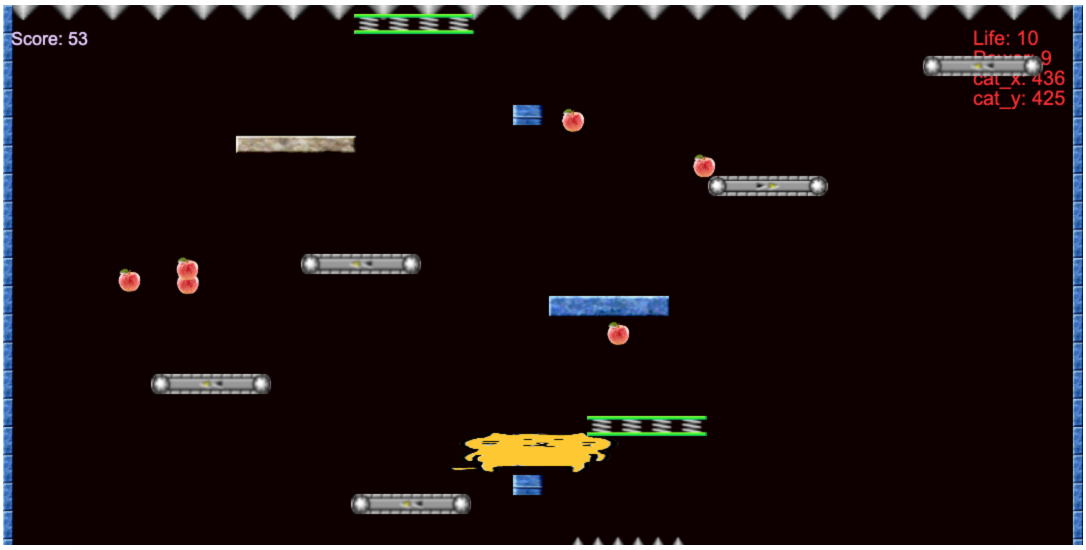
- 能量跳：
  小貓可消耗**3**能量向上跳躍一段距離，是本遊戲的要拿高分的關鍵

- 吃蘋果：
  每過一段時間隨機在地圖某處生成一粒蘋果，地圖左半由於難度較高，吃到蘋果得**2**分，右半部的蘋果則是**1**分

- 體重過重系統：
  每吃一粒蘋果，小貓會變肥，寬度變寬，雖然夠容易從邊上站上平台，但不同體重將有一定的機率踩壞地板往下掉，因此即使踩到台階也不能鬆懈，踩空時可以借用一踩到台階的瞬間馬上跳起來

變肥了~

小貓吃蘋果



系統開發平台

使用Mac的Visual Studio Code開發、javascript全部寫在canvas上、使用Google Chrome執行

程式說明

Html

Final_project-小貓吃蘋果 > final_0513404.html > html

```html
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>0513404 final project</title>

    <style>
        * { padding: 0; margin: 0; }
        canvas { background: rgb(14, 0, 0); display: block; margin: 0 auto; }
    </style>
</head>
<body>
<h1 style="text-align:center">小貓吃蘋果</h1>
<canvas id="mycanvas" width="1080" height="540"></canvas>
<script type="text/javascript" src="final.js"></script>


</body>
</html>
```

這部分非常簡單，創建一個canvas，剩下的都是在js處理。

JavaScript

創建物件–貓咪

基本參數設定如下：

```javascript
//玩家物件
class Player{
    constructor(args){
        this.cat_x = 500; //初始x座標
        this.cat_y = 45; //初始y座標
        this.cat_w = 37; //初始肥度
        this.cat_h = 45; //初始身高
        this.live = 10; //初始血量
        this.power = 10; //初始能量
        this.max_live = 10; //最大血量
        this.max_power = 10; //最大能量
        this.height = 0; //離地面高度
        this.touch = 1; //是否有觸碰到東西
        this.status = 0; //小貓的動作
        this.hurt = 0; //防止連續扣血的flag
        this.heal = 0; //防止連續補血的flag
        this.addpower = 0; //防止連續補能量的flag
    }
```

貓咪畫圖：包含依照貓咪的status改變其動作、畫血量、能量和座標值

```
draw(status){
    if(this.status == 0){
        ctx.drawImage(cat,this.cat_x,this.cat_y,this.cat_w,this.cat_h);
    }else if(this.status == 1){
        ctx.drawImage(cat_l,this.cat_x,this.cat_y,this.cat_w,this.cat_h);
    }else if(this.status == 2){
        ctx.drawImage(cat_r,this.cat_x,this.cat_y,this.cat_w,this.cat_h);
    }else if(this.status == 3){
        ctx.drawImage(cat_jump,this.cat_x,this.cat_y,this.cat_w,this.cat_h);
    }
    ctx.font = "20px Arial";
    ctx.fillStyle = "#FF2D2D";
    ctx.fillText("Life: " + this.live, 970, 40);
    ctx.font = "20px Arial";
    ctx.fillStyle = "#FF2D2D";
    ctx.fillText("Power: " + this.power, 970, 60);
    ctx.fillStyle = "#FF2D2D";
    ctx.fillText("cat_x: " + this.cat_x, 970, 80);
    ctx.fillStyle = "#FF2D2D";
    ctx.fillText("cat_y: " + this.cat_y, 970, 100);
}
```

其他此物件的方法：設定重力、改變血量、改變能量

```
gravity(){
    //console.log(this.touch )
    if(this.touch == 0){
        this.cat_y = this.cat_y + G; //重力下墜
    }
}
live_delta(d){
    this.live += d
    if (this.live > this.max_live){
      this.live = this.max_live;
    }
    if (this.live <= 0){
      this.live = 0
    }
}
power_delta(d){
    this.power += d
    if (this.power > this.max_power){
      this.power = this.max_power;
    }
    if (this.power <= 0){
      this.power = 0
    }
}
```

創建物件–平台

基本參數設定如下：

```
class Wall{
    constructor(x,y,type){
        this.wall_x = x; //初始x座標
        this.wall_y = y; //初始y座標
        this.v = 3; //移動速度
        this.width =  120; //平台寬度
        this.height = 20; //平台高度
        this.extraHeight = 0; //最高能到的邊界
        this.wall_type = type; //平台種類
        this.active = true; //是否還在遊戲畫面中
        this.touch = 0; //是否有是否有觸碰到東西
    }
```

讓平台移動，左右兩半部移動方向不同分開寫，如果touch==1代表有接觸到貓咪，就帶著貓一起移動，超出遊戲畫面範圍的平台active會被設為flase，若此時還有貓在上面若在右半部會扣血，左半部會直接出局。

```
update(){ //右邊 往上跑
    this.wall_y = this.wall_y - this.v;
    if(this.touch == 1){
        cat1.cat_y = cat1.cat_y - this.v;
    }
    if (this.wall_y < -20){
        this.active=false;
        if(this.touch == 1){ //如果消失時是有人站上面直接出局
            touch_count = 0;
            cat1.cat_y = 0;
            cat1.live_delta(-3);
        }
  }
}
update2(){ //左邊 往下跑
    this.wall_y = this.wall_y + this.v;
    if(this.touch == 1){
        cat1.cat_y = cat1.cat_y + this.v;
    }
    if (this.wall_y > 540){
        this.active=false;
        if(this.touch == 1){ //如果消失時是有人站上面直接出局
            alert("遊戲結束 總共得到"+String(score)+"分");
            document.location.reload();
            init(); // 重置
        }
  }
}
```

依照種類的不同畫出不同的平台：

```
draw(){
    //受傷
    if (this.wall_type == 0){ //"hurt"
        ctx.drawImage(stick1,this.wall_x ,this.wall_y,this.width,this.height);
    }
    //跳跳板
    if (this.wall_type == 1){ //"jump"
        ctx.drawImage(stick2,this.wall_x ,this.wall_y,this.width,this.height);
    }
    //live普通
    if (this.wall_type == 2){ //"normal"
        ctx.drawImage(stick3,this.wall_x ,this.wall_y,this.width,this.height);
    }
    //power普通
    if (this.wall_type == 3){ //"fade"
        ctx.drawImage(stick4,this.wall_x ,this.wall_y,this.width,this.height);
    }
    //右滑
    if (this.wall_type == 4){ //"right"
        ctx.drawImage(stick5,this.wall_x ,this.wall_y,this.width,this.height);
    }
    //左滑
    if (this.wall_type == 5){ //"left"
        ctx.drawImage(stick6,this.wall_x ,this.wall_y,this.width,this.height);
    }
}
```

每個種類踩到都有不同的事發生：

```
step(){
    if (this.wall_type == 0){ //受傷
        if(cat1.hurt == 0){
            cat1.live_delta(-3);
            cat1.hurt = 1;
        }
        cat1.heal = 0;
        cat1.addpower = 0;
    }
    if (this.wall_type == 1){ //跳跳板
        isjump = 1;
        isjump2 = 1;
        //cat1.cat_y = cat1.cat_y - 120;
        cat1.hurt = 0;
        cat1.heal = 0;
        cat1.addpower = 0;
    }
    if (this.wall_type == 2){ //live普通
        if(cat1.heal == 0){
            cat1.live_delta(1);
            cat1.heal = 1;
        }
        cat1.hurt = 0;
        cat1.addpower = 0;
    }
    if (this.wall_type == 3){ //power普通
        if(cat1.addpower == 0){
            cat1.power_delta(2);
            cat1.addpower = 1;
        }
        cat1.hurt = 0;
        cat1.heal = 0;
    }
    if (this.wall_type == 4){ //右滑
        cat1.cat_x = cat1.cat_x + 2;
        cat1.hurt = 0;
        cat1.heal = 0;
```

```
            cat1.addpower = 0;
        }
        if (this.wall_type == 5){ //左滑
            cat1.cat_x = cat1.cat_x - 2;
            cat1.hurt = 0;
            cat1.heal = 0;
            cat1.addpower = 0;
        }
    }
```

創建物件–蘋果

基本參數設定和畫蘋果、偵測是否碰到兩種方法：

```
class Apple{
    constructor(x,y,apple_score){
        this.apple_x = x; //初始x座標
        this.apple_y = y; //初始y座標
        this.apple_w =  30; //蘋果寬度
        this.apple_h = 30; //蘋果高度
        this.active = true; //是否有缺蘋果，預設是左右各三顆，有缺就隨機生成新的
        this.touch = 0; //是否有是否有觸碰到東西
        this.score = apple_score; //左右的蘋果分數不同
    }
    draw(){
        ctx.drawImage(apple,this.apple_x ,this.apple_y, this.apple_w, this.apple_h);
    }
    touch_detection(){
        if(cat1.cat_x + cat1.cat_w >= this.apple_x && cat1.cat_x <= this.apple_x + this.apple_w){
            if (cat1.cat_y + cat1.cat_h >= this.apple_y && cat1.cat_y <= this.apple_y + this.apple_h ){
                score = score + this.score;
                cat1.power_delta(1);
                cat1.live_delta(1);
                this.active = false;
                cat1.cat_w = cat1.cat_w + 3;
            }
        }
    }
}
```

是否觸碰到–touch()

判斷各種觸碰到的狀況，包括由於貓太肥導致的不算在判定內

```
function touch(eachwall){

    if(touch_count == 1){ //目前已在某平台上
        if(eachwall.touch == 1){
            if(cat1.cat_x >= 510 - cat1.cat_w && cat1.cat_x <=510 + wall_w * 3 ){ //x座標是否在初始平台範圍內
                if (cat1.cat_y + cat1.cat_h <= 100 && cat1.cat_y + cat1.cat_h >= 100 - G - eachwall.v){  //y座標是否離初始平台面算接觸
                    cat1.cat_y = 100 - cat1.cat_h ;
                    cat1.touch = 1;

                }else if(cat1.cat_y + cat1.cat_h <= 470 && cat1.cat_y + cat1.cat_h >= 470 - G ){ //y座標是否離中下平台面算接觸
                    cat1.cat_y = 470 - cat1.cat_h ;
                    cat1.touch = 1;
                    down_stair = 1;

                }else{ //確定不在中間兩個固定平台
                    console.log(3);
                    cat1.touch = 0;
                    eachwall.touch = 0;
                    touch_count = 0;

                }
            }else if (cat1.cat_x >= eachwall.wall_x - cat1.cat_w && cat1.cat_x <= eachwall.wall_x + eachwall.width ){ //x座標是否在平台範圍內
              if (cat1.cat_y + cat1.cat_h <= eachwall.wall_y && cat1.cat_y + cat1.cat_h >= eachwall.wall_y - G - eachwall.v){ //y座標
                cat1.cat_y = eachwall.wall_y - cat1.cat_h ;
                cat1.touch = 1;
                eachwall.touch = 1;
                touch_count = 1;
                eachwall.step();

              }else{ //在平台x座標範圍但高度不對
                cat1.touch = 0;
                eachwall.touch = 0;
                touch_count = 0;
              }
            }else{ //不在平台x座標範圍也不在中柱
                cat1.touch = 0;
                eachwall.touch = 0;
                touch_count = 0;
            }
        }
    }else if(touch_count == 0){ //目前不在任何平台上 剩下的與上面部分邏輯相同
        //因為角色下半身窄 加上誤差看起來比較合理
        if(cat1.cat_x >= 510 - cat1.cat_w  && cat1.cat_x <=510 + wall_w * 3 ){
            if (cat1.cat_y + cat1.cat_h <= 100 && cat1.cat_y + cat1.cat_h >= 100 - G ){
                cat1.cat_y = 100 - cat1.cat_h ;
                cat1.touch = 1;

            }else if(cat1.cat_y + cat1.cat_h <= 470 && cat1.cat_y + cat1.cat_h >= 470 - G ){
                cat1.cat_y = 470 - cat1.cat_h ;
                cat1.touch = 1;
                down_stair = 1;

            }else{
                cat1.touch = 0;
                eachwall.touch = 0;

            }
        }else if (cat1.cat_x >= eachwall.wall_x - cat1.cat_w  && cat1.cat_x <= eachwall.wall_x + eachwall.width){
        if (cat1.cat_y + cat1.cat_h <= eachwall.wall_y && cat1.cat_y + cat1.cat_h >= eachwall.wall_y - G){
            cat1.cat_y = eachwall.wall_y - cat1.cat_h ;
            cat1.touch = 1;
            eachwall.touch = 1;
            touch_count = 1;
            eachwall.step();

        }else{
            cat1.touch = 0;
            eachwall.touch = 0;
            //console.log(cat1.cat_x)
            //console.log(cat1.cat_y)
        }
        }else{
            cat1.touch = 0;
            eachwall.touch = 0;
            //console.log(cat1.cat_x)
            //console.log(cat1.cat_y)
        }
    }
```

其他

設定鍵盤按鍵–左右上的按下和放開

```
function keyDownHandler(e) { //判斷按下的鍵是哪個
    if(e.key == "Right" || e.key == "ArrowRight") {
        cat1.status = 2;
        rightPressed = true;
    }
    else if(e.key == "Left" || e.key == "ArrowLeft") {
        cat1.status = 1;
        leftPressed = true;
    }
    else if(e.key == "Space" || e.key == "ArrowUp") {
        cat1.status = 3;
        upPressed = true;
    }
}

function keyUpHandler(e) { //判斷放開的鍵是哪個
    if(e.key == "Right" || e.key == "ArrowRight") {
        cat1.status = 0;
        rightPressed = false;
    }
    else if(e.key == "Left" || e.key == "ArrowLeft") {
        cat1.status = 0;
        leftPressed = false;
    }
    else if(e.key == "Space" || e.key == "ArrowUp") {
        cat1.status = 0;
        down_stair = 0;
        upPressed = false;
    }
}
```

畫出場地、分數及偵測是否符合遊戲結束條件

```
function drawstage() {
    ctx.drawImage(ceiling,0,0,1080,15);
    ctx.drawImage(left_wall,0,0,wall_w,540);
    ctx.drawImage(right_wall,1070,0,wall_w,540);
    ctx.drawImage(middle_wall,510,100,wall_w * 3,20);
    ctx.drawImage(middle_wall,510,470,wall_w * 3,20);
}

function drawscore() {
    ctx.font = "18px Arial";
    ctx.fillStyle = "#E6CAFF";
    ctx.fillText("Score: " + score, 8, 40);
}

function GGdetection(){
    if(cat1.live == 0 || cat1.cat_y >= 550 - cat1.cat_h){
        alert("遊戲結束 總共得到"+String(score)+"分");
        document.location.reload();
        init(); // 重置
    }
}
```

跳起來的function，跳躍時消耗Power，除了要抵消重力外還要給一個往上的速度，希望未來可以多做加速度

```
function jump(){
    if(isjump == 1){
        if(cat1.power >= 3){
            if(jh < jump_max ){
                jh += jump_speed;
                cat1.cat_y = cat1.cat_y - G - jump_speed - 3;
                if(cat1.cat_y < 0 && cat1.hurt == 0){
                    cat1.cat_y = 0;
                    cat1.live_delta(-3);
                    cat1.hurt = 1;
                }
            }else{
                isjump = 0;
                if(isjump2 == 0){
                    cat1.power = cat1.power - power_consume;
                }

                jh = 0;
            }
        }
        else{
            isjump = 0;
        }
    }
}
```

定時生成新蘋果及新台階，隨機屬性與位置，另外每次更新都判定是否遊戲結束(GGdetection())

```
function update(){
    //console.log(cat1.cat_y);
    time++;
    cat1.draw(cat1.status);
    if (time%40 == 0){ //推入新的地板
        stairs_arr.push(new Wall(Math.random()*410 + 540, 540, parseInt(Math.random()*6))); //右
        stairs_arr2.push(new Wall(Math.random()*380, 0, parseInt(Math.random()*6))) //左
    }
    if (time%60 == 0){ //出現新蘋果
        if(apple_arr.length < 3){
            apple_arr.push(new Apple(Math.random()*410 + 540, Math.random()*260 + 70, 1)); //右
        }
        if(apple_arr2.length < 3){
            apple_arr2.push(new Apple(Math.random()*380, Math.random()*300 + 70, 2)) //左
        }
    }
    cat1.gravity();
    GGdetection();
}
```

每秒定時畫圖，觸法上面寫到的各function

```
function draw() {
    //ctx.clearRect(cat_x,cat_y,cat_w+10,cat_h);
    ctx.clearRect(0, 0, canvas.width, canvas.height); //把原本的點刪掉
    drawstage();
    drawscore();
    update();
    jump();
    console.log(touch_count)
    apple_arr = apple_arr.filter(apple=>apple.active);
    apple_arr.forEach(apple=>apple.draw());
    apple_arr.forEach(apple=>apple.touch_detection());
    apple_arr2 = apple_arr2.filter(apple=>apple.active);
    apple_arr2.forEach(apple=>apple.draw());
    apple_arr2.forEach(apple=>apple.touch_detection());
    //鍵盤操作
    if(rightPressed) {
        cat1.cat_x += 5;
        if (cat1.cat_x + cat1.cat_w > canvas.width - wall_w){
            cat1.cat_x = canvas.width - cat1.cat_w - wall_w;
        }
    }
    else if(leftPressed) {
        cat1.cat_x -= 5;
        if (cat1.cat_x < 0 + wall_w){
            cat1.cat_x = 0 + wall_w;
        }
    }
    if(upPressed) {
        if(touch_count == 1 || down_stair == 1){ //down_stair是指下面的小平台
            isjump = 1;
            isjump2 = 0;
            down_stair = 0;
        }
    }
    stairs_arr = stairs_arr.filter(wall=>wall.active); //把已經出界的去掉
    stairs_arr.forEach(wall=>wall.update());
    stairs_arr.forEach(wall=>wall.draw());
    stairs_arr.forEach(wall=>touch(wall));

    stairs_arr2 = stairs_arr2.filter(wall=>wall.active);
    stairs_arr2.forEach(wall=>wall.update2());
    stairs_arr2.forEach(wall=>wall.draw());
    stairs_arr2.forEach(wall=>touch(wall));

}
```

心得

這個遊戲是從一款非常有名的遊戲–小朋友下樓梯延伸的，當初在網路上看到有人提供圖片素材包跟大致程式邏輯應該如何實現的教學，有很多方法都可以寫，最終決定使用之前作業也用過的canvas來實作，原本是想讓小貓可以在地圖間飛來飛去吃東西，然後左右兩邊有不同的重力場，不過只做出固定速度，現實中應該還要包含重力加速度，有試著實作，但運算變得很複雜，怕來不及做出來，因此先做沒有加速度的版本。得意的成果應該就是能讓小貓跳來跳去吃蘋果這目標有完成，然後我覺得吃的蘋果越多會變越肥，有可能把平台踩壞這部分蠻有趣的，有種貪食蛇的感覺。