Flappy bird

1. Aufgabenstellung

- Zusammenarbeit in Gruppen von 2 (max. 3)
- Einfaches HTML/JS-Browsergame
- Zu verwendende Mechanismen: Canvas, Objekt-Listen, Timer, Images, eventuell auch Audios
- Bei der Präsentation zu zeigen: Umgang mit Debugger, kleine Programmänderungen (um sicherzugehen, dass jeder das Präsentierte selbst entwickelt hat)
- Abgabe eine Beschreibung des Spiels (Spiellogik, technischer Hintergrund, …) im Stil eines Laborprotokolls
- Abgabe eines gezippten Ordners (der alle Dateien enthält); Haupt-HTML-Seite heißt "index.html" (Default-Start-Datei)
- Motto: Weniger ist Mehr (nichts kopieren, sondern was im Spiel vorkommt, ist selbst programmiert!)

2. Spielidee

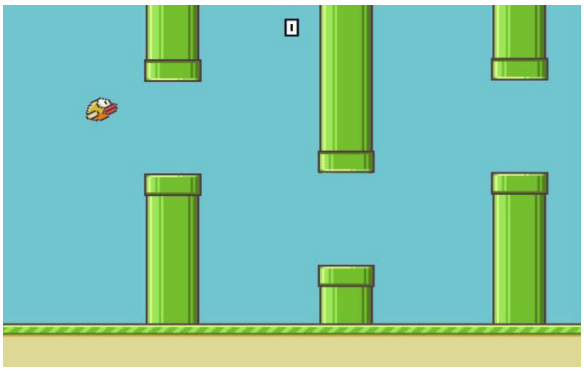
Ursprung des Originals:

Flappy Bird ist eine App des vietnamesischen Entwicklers Dong Nguyen aus dem Jahr 2013. Veröffentlicht wurde sie von. Gears Studios. Die App wurde ursprünglich im Mai 2013 für Android und für das Apple iPhone 5, später für iOS 6 und anschließend im September 2013 für iOS 7 entwickelt.

Spielprinzip:

Die grafische Gestaltung von Flappy Bird ähnelt der Grafik von Super Mario World. Der Spieler führt durch das Tippen auf den Bildschirm einen Vogel durch eine von rechts nach links scrollende Spielwelt, wobei der Vogel die paarweise von oben und unten ins Bild ragenden grünen Röhren nicht berühren darf, sondern zwischen ihnen hindurchfliegen muss.

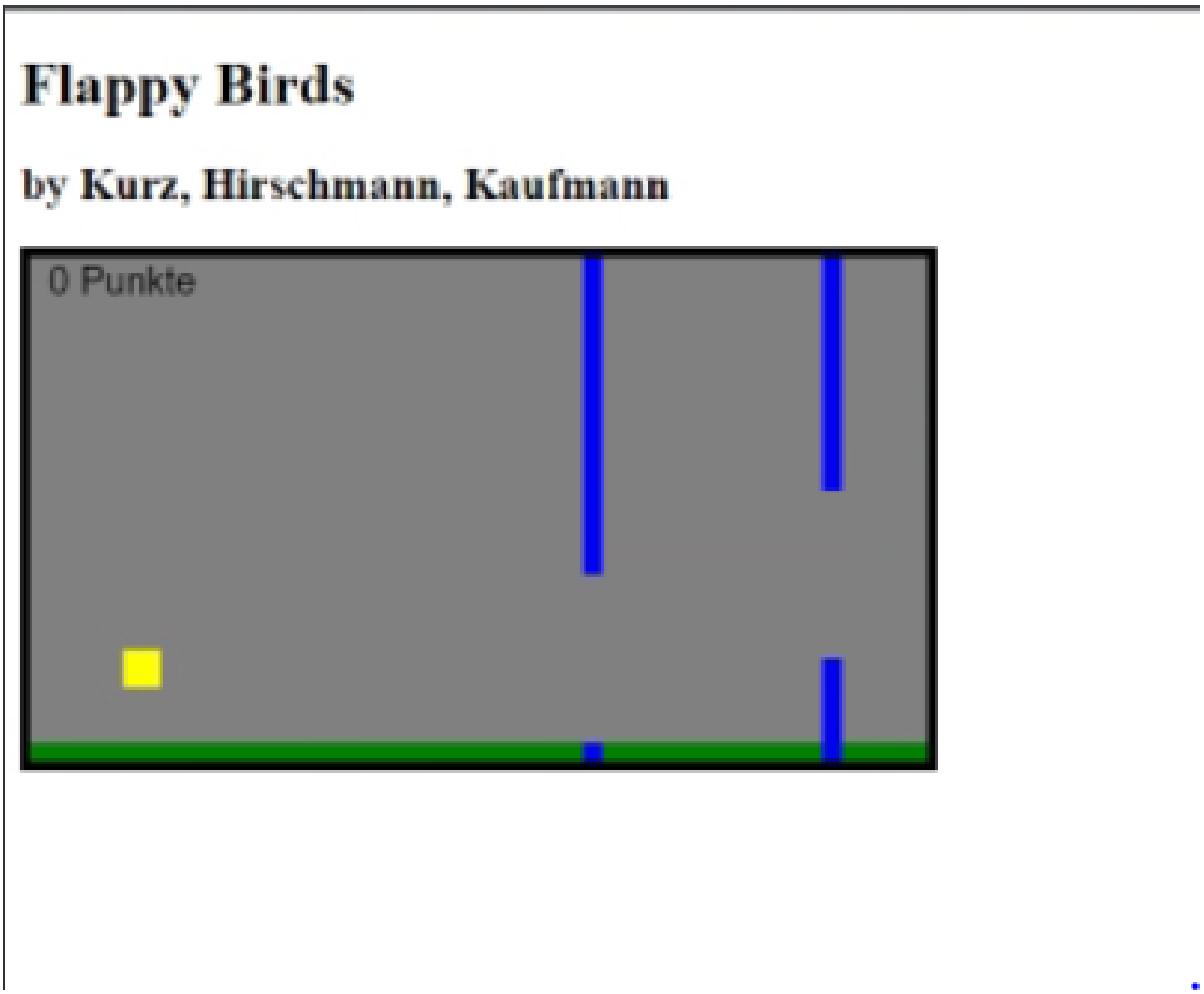
Die Position der Flugschneise variiert dabei. Jedes Mal, wenn der Vogel zwischen einem Rohrpaar hindurchfliegt, sammelt er einen Punkt. Dieser Vorgang wird automatisch abgebrochen, wenn der Vogel eines der Rohre berührt.



3. Spiellogik

Unser Spiel FlappyBird ist wie oben angeführt, an das Spiel von Dong Nguyen angelehnt. Man muss mit der Leertaste so schnell drücken damit das Quadrat in der Luft bleibt. Das Quadrat sollte nicht den Boden, den an den Boden gezeichneten Rechtecke oder die von der Decke hängenden Rechtecke nicht berühren sonst hat man im Spiel „verloren“ und das Game resetet sich.

Außerdem läuft im linken oberen Eck eine Score Anzeige, welche die Punkte anzeigt die man im Spiel, durch das überwinden der Rechtecke, erhält.



4. Aufbau des Programmiercodes und technischer Hintergrund

In der .html Datei wird wird die Javascriptdatei aufgerufen und das Canvas erstellt.

In den ersten 31 Zeilen werden die Variablen für die Spielschwierigkeit, für die Spieler, für die Hindernisse, für die Grasfläche und für den Spielscore angegeben.

```
var cvs = document.getElementById("myCanvas");
var ctx = cvs.getContext("2d");

//Variablen für Spielschwierigkeit
    //Hier kann man sich das Spiel beliebig schwerer als auch leichter machen
    var gravity=1.2;
    var gap =90;
    var Geschwindigkeit= 2;


//Variablen für Spieler
    var PlayerX=50;
    var PlayerY=200;
    var PlayerBreite=20;
    var PlayerHoehe=20;

//Variablen für Hindernisse
    var Hindernisse =[];
    var HindernissX=300;
    var HindernissY=120;
    var HindernisshoeheOben=170;
    var Hindernissunten = HindernisshoeheOben+gap;
    var OberesHindernissBreite=10;
    var UnteresHindernissBreite=10;
    var HindernisshoeheUnten=240;

//Variable für Grassfläche
    var GrassflaecheHoehe = 10;

//Variable für Punkte
    var Punkte = 0;
```

Dann haben wir die Funktion eingebaut, damit wir unseren Vogel in Form eines Quadrates bewegen können.

```
//Spieler bewegen
    var c = document.getElementById("myCanvas");
    document.addEventListener("keydown", keyspace);

    function keyspace() {
        PlayerY-=25;
    }

//x und y Koordinaten definieren
    Hindernisse[0] ={
        x:cvs.width,
        y:0
    };
```

Unsere nächste Funktion schrieben wir um das Gras, den Vogel in Form eines Quadrates und die Hindernisse zeichnen zu können. Für das Quadrat benötigten wir eine Gravitation damit der Spieler auf den Boden zurück fällt.

Um die Hindernisse zeichnen zu können benötigten wir eine for- und eine if- Schleife.

```
//Spiel Gegenstände zeichnen
    function drawAll(){
        var ele = document.getElementById("myCanvas");
        var ctx = ele.getContext("2d");

        //Clear everything
        ele.width =ele.width;

        //Grassfläche zeichnen
        ctx.fillStyle="green";
        ctx.fillRect (0, 260, 480, GrassflaecheHoehe);

        //Spieler zeichnen
        ctx.fillStyle = "yellow";
        ctx.fillRect (PlayerX, PlayerY, PlayerHoehe, PlayerBreite);

        /*Gravitation Spieler*/ PlayerY+=gravity;

        //Hindernisse zeichnen
        for (var i=0; i<Hindernisse.length; i++){
            ctx.fillStyle="blue";
            ctx.fillRect (Hindernisse[i].x, Hindernisse[i].y, OberesHindernissBreite, HindernisshoeheOben);

            ctx,fillStyle="blue";
            ctx.fillRect (Hindernisse[i].x, Hindernisse[i].y+Hindernissunten, UnteresHindernissBreite, HindernisshoeheUnten);

            Hindernisse[i].x-=Geschwindigkeit;  //Geschwindigkeit der Hindernisse kann oben bei den Variablen geändert werden. Man kan sozusagen die Schwierigkeit erhöhen.

            if(Hindernisse[i].x == 350){
                Hindernisse.push({
                    x : cvs.width,
                    y : Math.floor(Math.random()*HindernisshoeheOben)-
                    HindernisshoeheOben
                });
            }
```

Unsere letzte beiden letzten Schleifen bestanden daraus, dass wenn eine Kollision zwischen unseren „Player“ und dem Hindernis entsteht, die Seite reloadet wird.

Doch wenn man ein Hindernis überstanden hat bekam man einen Punkt in der Scoreliste hinzugefügt.

```
            //Bei Kollision vorbei
            if(PlayerX + PlayerBreite >= Hindernisse[i].x && PlayerX <= Hindernisse[i].x + OberesHindernissBreite
                && (PlayerY <= Hindernisse[i].y + HindernisshoeheOben || PlayerY+PlayerHoehe >=Hindernisse[i].y+Hindernissunten)
                ||PlayerY + PlayerHoehe >= cvs.height - GrassflaecheHoehe){
                location.reload(); //Seite wird neu geladen wenn eine Kollision passiert
            }
            //Wenn Hinderniss geschaft Punkte +1
            if (Hindernisse[i].x==50){
                Punkte++;
            }
        }

        //Punkte anzeigen
        ctx.fillStyle = "Black   ";
        ctx.font = "20px Arial"
        ctx.fillText(Punkte+" Punkte",10, 20)


        requestAnimationFrame(drawAll);
    }
```

Die letzte Zeile bestand daraus das Spiel zu Starten und drawAll aufzurufen.