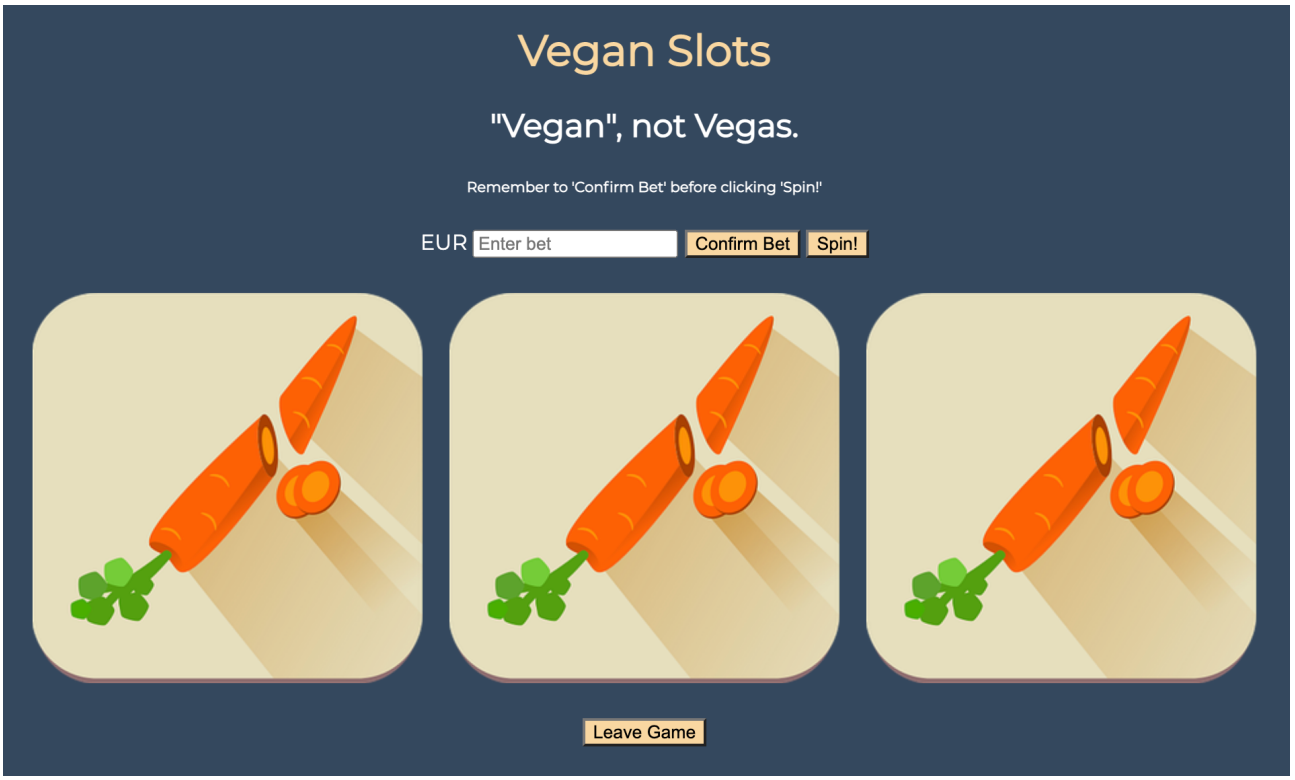

Ana Claver Barceló

PAC5 - Actividad de desarrollo (UF3)

20 de abril 2021

Ana Claver Barceló

Índice

HTML **3**

CSS **4**

JAVASCRIPT **5**

BLOQUE 1 - Declaración de variables 5
BLOQUE 2 - Cinco funciones 6
BLOQUE 3 - Llamadas a las funciones al clicar un botón 9

HTML

El **head** incluye metadata, los enlaces a los archivos externos .css y .js, además del enlace a la Google font.

```
<head>

  <meta charset="UTF-8">
    <meta name="description" content="Slots Machine">
    <meta name="keywords" content="slots, money, free, gambling, gaming">
    <meta name="author" content="Ana Claver">
  <title>Vegan Slots</title>
  <link rel="stylesheet" href="css/style.css">
  <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Montserrat">
  <script type = "text/javascript" src ="../js/javascript.js"> </script>

</head>
```

Todos los elementos del **body** están centrados y están en una fuente de Google, Montserrat:

1. Nombre y eslogan del juego (h1 y h2), además de un pequeño recordatorio (h6)
2. Una caja para introducir el valor de la apuesta
3. Tres divs para los tres slots (anidados en otro div container)
4. Tres botones: CONFIRM BET, SPIN y LEAVE GAME
5. Texto que muestra los resultados de la última tirada (h3)
6. Historial de la sesión de cada tirada: apuesta, premio y ganancias/pérdidas de la misma. (En divs para poder centrarlos sin problemas con el número del elemento <ol>).

```
<body>

<h1>Vegan Slots</h1>
<h2>"Vegan", not Vegas.</h2>
<h6> Remember to 'Confirm Bet' before clicking 'Spin!'</h6>

<p class="center">EUR <input type="number"  id="bet" placeholder="Enter bet"/>
<button id="insert">Confirm Bet</button>
<button id="spin">Spin!</button>
      </p>

<div class="container">
   <div id="slot0" class="slot"></div>
   <div id="slot1" class="slot"></div>
   <div id="slot2" class="slot"></div> </div>

<button id="leave">Leave Game</button>
<h3>Current Game Results </h3>
 <p id="history"></p><br>
<h4> Game History Results </h4>

<div class="container">
<div><ol id="history2"></ol></div>
 </div>
 </body>
```

CSS

Los elementos más destacables son el div container (para conseguir un centrado fácil a través de flex) y los tres divs de los slots. A todos se les pasa el estilo a través de su clase (‘container’ y ‘slot’, respectivamente):

```
.container {
  display: flex;
  justify-content: center;}


.slot {
  background-image: url("../img/zanahoria.png");
  background-size: 100% 100%;
  display: inline-block;
  margin: 10px;
  padding: 20px;
  height: 250px;
  width: 250px;
}
```

Sin complicaciones para los principales elementos como paragraph, headers y listas. La parte clave reside en el ‘body’:

```
body {
    background: #34495E;
    font: 16px ;
    font-family: "Montserrat", sans-serif;
    text-align: center;
   }

p, ol, li {
    color: white;
    font: 14px;
}

h1 {
     color: #FAD7A0;
     font: 40px;
     font-weight: bold;
}
```

JAVASCRIPT

El código del programa consiste en **3 bloques de código** (dentro de la función 'load' de la página):

BLOQUE 1 - Declaración de variables

```
window.addEventListener ('load', function(){

// Declaramos las variables que necesitamos

var slot0 = document.getElementById("slot0");
var slot1 = document.getElementById("slot1");
var slot2 = document.getElementById("slot2");
var spinButton = document.getElementById("spin");
var winnings = 0;
var leaveButton = document.getElementById("leave");
var arrayVerduras = ["zanahoria", "aguacate", "ajo", "cebolla", "pepino", "puerro", "tomate"];
var arrayResultados = [];
var arrayHistorialTexto = [];
var arrayHistorialGanancias = [];
var insertButton = document.getElementById("insert");

```

Se pueden observar las que corresponden a elementos HTML por el "document.getElementById" que muestra su relación con algún evento o acción.

Vemos también el array de las variables de las hortalizas (arrayVerduras) cuya longitud marcará la amplitud de la llamada al random, y también las arrays donde guardamos los resultados de los randoms (arrayResultados), los mensajes mostrados para hacer un historial (arrayHistorialTexto) y la de ganancias para poder decirle al usuario su saldo cuando abandone el juego (arrayHistorialGanancias).

BLOQUE 2 - Cinco funciones

```
17
18  // A través del BOTÓN INSERT deshabilitamos la caja y asingamos valor a la variable global
    apuesta.
19  function toValidate() {}
27
28  // A través del BOTÓN SPIN se inactiva la caja de la entrada de apuesta, se llama al random 3
    veces y se populan los slots con los resultados.
29  function toSpin() {}
66
67    // FUNCION ANIDADA para calcular monedas sengún la apuesta del usuario y los multiplicadores
    del enunciado.
68  function calcular(){}
109
110   // FUNCION ANIDADA que muestra los resultados de la última partida y también el historial
    completo de la sesión del jugador.
111 function mostrarResultados(){}
129
130   // A través del BOTÓN LEAVE GAME se mostrará el total de ganancias en una ventana emergente,
    cuando el jugador abandone el juego.
131 function toLeave() {}
140
141
```

Tres se disparan con el evento 'click' de los tres botones de nuestra web:

1. Botón **CONFIRM BET** → llamada a la función **toValidate()**
2. Botón **SPIN!** → llamada a la función **toSpin()**
3. Botón **LEAVE GAME** → llamada a la función **toLeave()**

Las otras dos, las funciones **calcular()** y **mostrarResultados()**, son funciones anidadas dentro de **toSpin()** (sus llamadas son la última línea de código en la función precedente).

Decidí anidar estas dos funciones para que en el futuro podamos distinguir los tres pasos principales de la función **toSpin()**:

1. Llamada a los randoms
2. Cálculo de las ganancias basados en los multiplicadores
3. Visualización de los resultados

De este modo, cuando diseñemos otro juego para nuestro casino, podemos fácilmente identificar la estructura del código.

Si entramos en más detalle, veremos que la llamada al random empieza con dos variables: **numSlots** y **numFiguras**. En un futuro si tenemos un nuevo juego de tragaperras, solo tendremos que indicar cuantos slots y cuantas figuras tenemos y podremos aprovechar toda la función **toSpin()**:

```
function toSpin() {

// Si en un futuro creamos un juego de slots con un número distinto de figuras o de slots solo
tendremos que cambiar las 2 variables declaradas a continuación. El resto de código de la función se
aprovechará sin cambios

var numSlots = 3;
var numFiguras= 7;

//lamada a los randoms:
for (var i=0; i < numSlots; i++){

  var randomNumber = Math.floor((Math.random() * numFiguras));

  // Añadir valor a la array de Resultados
  arrayResultados.push(arrayVerduras[randomNumber]);

  //Muestro la hortaliza en el div:
  document.getElementById("slot"+i).style.backgroundImage="url('./img/" + arrayVerduras[randomNumber] +
".png')";

  }

 calcular();
}
```

Para la función **calcular()**, la primera de las dos funciones anidadas en **toSpin()**, empezamos con el recuento de las zanahorias con un bucle que se realizará tantas veces como slots tengamos:

```
55 function calcular(){
56
57    //Primero cuento las zanahorias para facilitar la estructura de control:
58
59    var zanahorias= 0;
60    for (var i=0; i < arrayResultados.length; i++){
61      if (arrayResultados[i]== "zanahoria")  {zanahorias++}
62    }
63
```

Seguimos con la estructura de control basada en el número de zanahorias que hayamos sacado:

```
    if (zanahorias==0)
      {
        if (arrayResultados[0] == arrayResultados[1]) {
              if (arrayResultados[1]==arrayResultados[2]) {winnings = betNumber * 5}}
        if (arrayResultados[0] == arrayResultados[1]) {
              if (arrayResultados[1]≠arrayResultados[2]) {winnings = betNumber * 2}}
        if (arrayResultados[0] ≠ arrayResultados[1]) {
              if (arrayResultados[1]==arrayResultados[2] || arrayResultados[0]==arrayResultados[2])
{winnings = betNumber * 2}
              else {winnings=betNumber * 0}}
      }

    if (zanahorias==1)
      {
              if (arrayResultados[0] == arrayResultados[1] || arrayResultados[1]==arrayResultados[2] ||
arrayResultados[0]==arrayResultados[2] ) {winnings=betNumber * 3}
              else {winnings=betNumber * 1}
      }


    if (zanahorias==2) {winnings=betNumber * 4}
    if (zanahorias==3) {winnings=betNumber * 10}

  //Reseteamos la array para la siguiente tirada.
  arrayResultados = [];
```

El último paso de la función **toSpin()** es la segunda función anidada **mostrarResultados().** Con esta se mostrarán los resultados al usuario, tanto de la última tirada (var **textoResultado**) como el historial de todas las tiradas de la sesión en curso (**arrayHistorialTexto**).

También deberemos alimentar un array que contenga las monedas ganadas/perdidas para poder mostrarla al usuario cuando pulse el botón TO LEAVE (**arrayHistorialGanancias**)

```
function mostrarResultados(){
  // Mostrar monedas ganadas en el HTML párrafo para el usuario
  var benefit = winnings - betNumber;
  var verb;
  if (benefit < 0) {verb= "lost"; benefit= benefit*-1;} else {verb="won"};
  var textoResultado = "BET: " + betNumber + " coins || PRIZE: " + winnings + " coins.  ⟶ You " +
verb +" "+ benefit + " coins.";

  document.getElementById("history").innerHTML= textoResultado;

  // Alimentamos array de resultados de la tirada en curso y también el array del historia de ganancias
para luego poder dar el saldo final al judador cuando salga del juego.
  arrayHistorialTexto.push(textoResultado);
  arrayHistorialGanancias.push(benefit);


  // Mostramos el historial a la salida en forma de lista enumerada
  document.getElementById('history2').innerHTML =
    '<li>'+  arrayHistorialTexto.join('</li><li>') + '</li>';
}
```

Dicha función, **toLeave()**, mostrará un pop-up con el sumatorio de todos los valores (positivos y negativos) almacenados en **arrayHistorialGanancias**:

```
120 ▾ function toLeave() {
121     //Recorremos la array que contiene todos los resultados y hacemos sumatorio de los valores:
122     var totalBalance=0;
123 ▾   for (var i=0; i < arrayHistorialGanancias.length; i++){
124       totalBalance =  totalBalance + arrayHistorialGanancias[i];
125     }
126 
127     alert("Your current balance is: " + totalBalance + " coins.");
128   }
129 
```

BLOQUE 3 - Llamadas a las funciones al clicar un botón

Son las siguientes y coinciden con los tres botones de nuestra página web:

```
141 
142   //Llamadas a las funciones al pulsar los tres botones de la web:
143   insertButton.addEventListener("click", toValidate);
144   spinButton.addEventListener("click", toSpin);
145   leaveButton.addEventListener("click", toLeave);
146 
```