Taller 2

Fecha: Septiembre de 2021

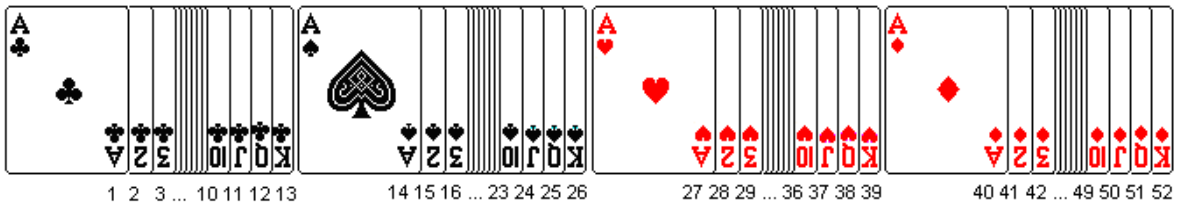
Indicador de logro a medir: Aplicar los conceptos de la lógica de programación, la orientación a eventos, y los lenguajes de marcado en el desarrollo de una aplicación que funcione en navegadores de Internet (aplicativo web del lado del cliente).

NOTAS:
- Este taller se debe hacer como preparación para evaluaciones. En ningún caso representará una calificación.
- Los primeros ejercicios se entregan resueltos como ejemplo para el desarrollo de los demás

Elaborar la respectiva aplicación en *HTML y JavaScript* para los siguientes enunciados:

1. Simular la repartición de 10 cartas al azar en un juego con la baraja inglesa y permitir consultar el estado de las agrupaciones encontradas, es decir, que ternas, cuartas, quintas, etc. de cada nombre de carta aparecen en el conjunto de las 10 cartas.

   Para quien no conozca la baraja inglesa, esta se compone de un juego de 52 cartas que combina 13 nombres de cartas con 4 tipos de pinta:



   Cada carta tendrá un número (índice) cuyo orden permitirá definir la pinta de acuerdo a la siguiente convención:
   - Valores entre 1 y 13 para las cartas con pinta ♣
   - Valores entre 14 y 26 para las cartas con pinta ♠
   - Valores entre 27 y 39 para las cartas con pinta ♥
   - Valores entre 40 y 52 para las cartas con pinta ♦

   Y el nombre dentro de los mismos valores agrupados según la siguiente tabla:

Pinta	Nombre Carta												
	A	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	***J***	***Q***	***K***
♣	1	2	3	4	5	6	7	8	9	10	11	12	13
♠	14	15	16	17	18	19	20	21	22	23	24	25	26
♥	27	28	29	30	31	32	33	34	35	36	37	38	39
♦	40	41	42	43	44	45	46	47	48	49	50	51	52

R/

Para este aplicativo web se tendrán las siguientes consideraciones:
- Se utilizará el framework de JavaScript *AngularJS*

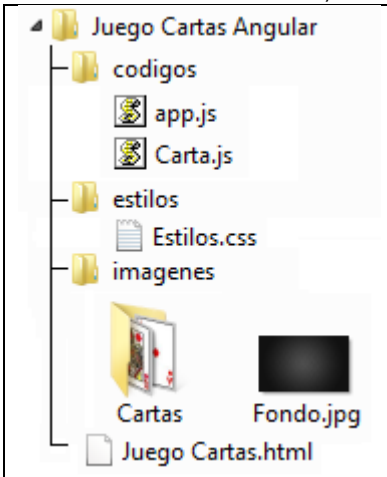
- Se aplicará el paradigma orientado a objetos y la arquitectura MVC
- El formato de salida se definirá mediante hojas de estilo (en inglés, CSS)

Definamos por lo tanto, las tecnologías antes mencionadas:

Tecnología	Descripción
AngularJS	Es una librería escrita en el lenguaje JavaScript que extiende al HTML nuevos atributos y directivas permitiendo un mejor desempeño y agilidad en la programación. Se distribuye como un archivo de JavaScript por lo cual debe ser agregado el respectivo tag a la página: `<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>`
CSS	Sigla de ***Cascading StyleSheets***, que puede traducirse como "*Hojas de estilo en cascada*", es un lenguaje que permite presentar, de manera estructurada, un documento que fue escrito en un lenguaje de marcado como lo es *HTML*. Lo que hace el CSS es encargarse de la descripción de las formas y de la sintaxis del lenguaje de marcado. De esta manera describe cómo se tienen que renderizar (generar las imágenes) los elementos que aparecen en pantalla. El diseño del CSS posibilita establecer una separación entre el contenido y la forma de presentación del documento (dada por las fuentes, los colores y las capas empleadas). Así se puede lograr que muchos documentos HTML compartan la apariencia, utilizando una única hoja de estilo para todos (que se especifica en un archivo .css). Gracias a esta particularidad, se evita tener que repetir el código en la estructura. La forma de hacer referencia a un archivo con estilos es la siguiente: `<link href="estilos/Estilos.css" rel="stylesheet" type="text/css" />`
MVC	Sigla de ***Modelo-Vista-Controlador*** el cual es un patrón de arquitectura de software que separa los datos y la lógica de negocio de la interfaz de usuario (junto con el módulo encargado de gestionar los eventos) en una aplicación. El objetivo de esta arquitectura es la reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.
UI	Término que se refiera a "interfaz de usuario" que, dicho de forma muy simple, es con lo que el usuario va a interactuar. En otras palabras, aquello con lo que podrá manejar los procesos digitales que desea realizar. Al ser la parte de ingeniería, es trabajo común de un programador y un diseñador.
UX	Término que se traduce como "experiencia de usuario". En este caso se refiere a lo que la persona percibe interna y emocionalmente al usar una interfaz, en pocas palabras, su

	reacción. El profesional que realiza esta labor debe tener amplios conocimientos en copywriting, en neuromarketing y en neurociencia en general.

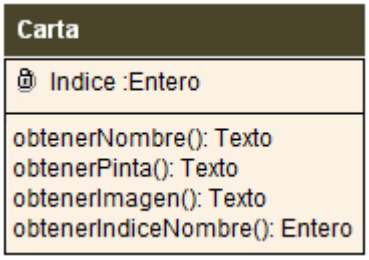
Con base en lo anterior, el aplicativo se estructura de la siguiente manera:

	• Un archivo de código JS con el controlador principal, el cual contiene las variables y métodos globales de la aplicación • Un archivo de código JS con la definición de la clase de objetos *Carta*. Esta incluye la funcionalidad para operar con cada carta de la baraja • Un archivo CSS que incluirá el diseño de todos los objetos que se desplegarán en la página web • Un archivo de imagen para el fondo • Una carpeta con los archivos de las imágenes de las cartas • Un documento *HTML* que contiene la estructura de lo que se visualizará

Se iniciará el proyecto editando la clase *Carta* la cual tendrá la funcionalidad asociada a cada carta. Esta se compone de:

- La propiedad *indice* que identificará la carta de acuerdo a la siguiente convención:
  - Valores entre 1 y 13 para las cartas con pinta ♥
  - Valores entre 14 y 26 para las cartas con pinta ♦
  - Valores entre 27 y 39 para las cartas con pinta ♠
  - Valores entre 40 y 52 para las cartas con pinta ♣

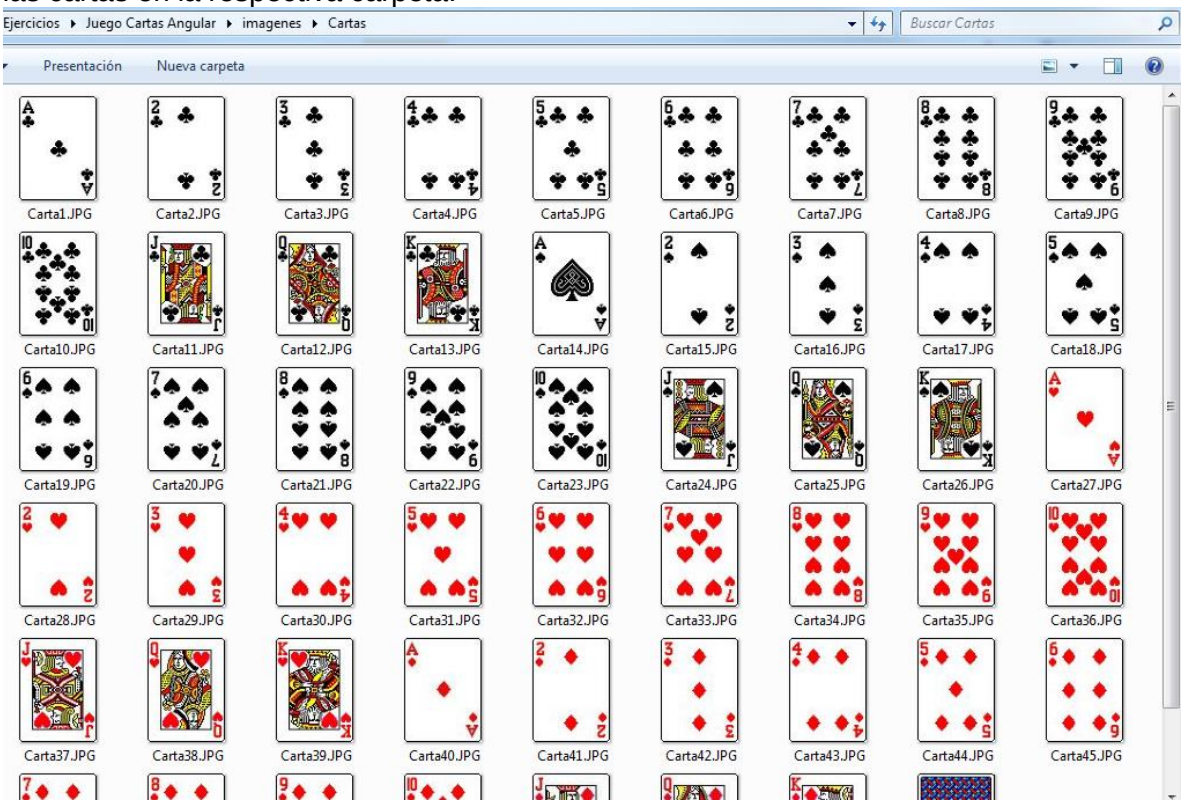
  Esta variable se generará aleatoriamente.



- El método *obtenerNombre()* devuelve un texto con el nombre de la carta (“*A”, “2”, “3”, “4”, …, “10”, “J”, “Q”, “K”*) con base en el índice
- El método *obtenerPinta()* devuelve un texto con la pinta de la carta (“*Corazón”*, *“Diamante”*, “*Pica”*, “*Trébol”*) con base en el índice
- El método *obtenerImagen()* que devuelve la ruta del archivo con la imagen de la respectiva carta.

Para obtener dicho nombre se debe seguir el siguiente convencionalismo: El nombre del archivo con la imagen de la carta se compondrá de la palabra "*Carta"* seguida del índice y finalizando con la respectiva extensión.

La siguiente imagen ilustra como aparecen los archivos de las imágenes de las cartas en la respectiva carpeta:



- El método *obtenerIndiceNombre()* calcula la posición en la lista de nombres de las cartas a partir del índice:

Pinta	Nombre Carta												
	A	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*J*	*Q*	*K*
♣	1	2	3	4	5	6	7	8	9	10	11	12	13
♠	14	15	16	17	18	19	20	21	22	23	24	25	26
♥	27	28	29	30	31	32	33	34	35	36	37	38	39
♦	40	41	42	43	44	45	46	47	48	49	50	51	52
Indice	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**

El código completo de la clase sería el siguiente:

```
//Declaración Clase CARTA
function Carta() {
      //Propiedades
      this.indice = Math.floor(Math.random() * 52) + 1;

      //Metodo que devuelve el nombre de la carta
      this.obtenerNombre = function() {
```

```
            numero = this.indice % 13;
            switch(numero){
                  case 0:
                        return "K";
                  case 1:
                        return "A";
                  case 11:
                        return "J";
                  case 12:
                        return "Q";
                  default:
                        return numero.toString();
            }
      };

      //Metodo que devuelve la pinta de la carta
      this.obtenerPinta = function() {
            if(this.indice<=13)
                  return "Trebol";
            else if (this.indice <= 26)
                  return "Pica";
            else if (this.indice <= 39)
                  return "Corazon";
            else
                  return "Diamante";
      };

      //Metodo que entrega la ruta de la imagen de la carta
      this.obtenerImagen = function() {
            return             "imagenes/Cartas/Carta"             +
this.indice.toString()+".jpg";
      };

      /* Metodo que devuelve el indice en la lista de nombres de
carta
      Ace=0, Dos=1, Tres=2, ..., Diez=9, Jack=10, Queen=11, King=12
*/
      this.obtenerIndiceNombre= function() {
            numero = this.indice % 13;
            if(numero == 0)
                  return 12;
            else
                  return numero-1;
      };
}
```

Siguiendo con el proyecto, se procederá ahora con la edición del documento *HTML* respectivo. Este tendrá los tags necesarios para cumplir con el diseño planteado en la siguiente ilustración:

Se puede apreciar que tendrá tres secciones:

- El título del aplicativo (que irá en la parte inferior)
- Los botones que activarán las acciones del aplicativo
- La mesa donde se mostrarán las cartas y los grupos encontrados

Cada sección se define en el documento *HTML* mediante el tag * *

El tag *** … *** permite definir una sección en un documento *HTML*. Es usado muy a menudo como contenedor de otros elementos *HTML* para aplicarles un formato en conjunto o desarrollar alguna tarea en JavaScript.

Los botones para las acciones realmente no van a ser elementos *<button>* ni *<input type="button">*. En su lugar utilizaremos el tag ancla ***<a>…***. Este tag es el que sirve para definir un hipervínculo. En este caso se utilizará para permitir aplicar un efecto especial de resalte cuando es presionado:

Para la lista de cartas se utilizará el tag de listas no ordenadas ***
…
*** el cual se combina con el tag ***
…
*** el cual define cada una de las opciones de la lista.

Para listar los grupos de cartas encontrados se incluirá una tabla donde cada fila corresponderá a un grupo.

El siguiente es el código del documento *HTML*:

```
<html ng-app="JuegoCartas">
<head>
      <title>Juego de Cartas</title>
      <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.
min.js"></script>
      <script src="codigos/app.js"></script>
      <script src="codigos/Carta.js"></script>
      <link       href="estilos/Estilos.css"       rel="stylesheet"
type="text/css" />
</head>
<body ng-controller="MetodoPrincipal">
      <div id="Titulo">Juego de Cartas</div>
      <div class="botones">
            <a   id="btnRepartir"   href="#"   ng-click="repartir()"
class="boton">Repartir</a>
            <a   id="btnVerificar"   href="#"   ng-click="chequear()"
class="boton">Chequear</a>
      </div>

      <div id="Mesa">
            <ul>
                  <li ng-repeat="carta in cartas" class="carta">
                        <img         src="{{carta.obtenerImagen()}}"
title="{{carta.obtenerNombre()}} {{carta.obtenerPinta()}}"  />
                  </li>
            </ul>
            <table>
              <tr ng-repeat="grupo in grupos">
                  <td>{{ grupo.tipo }}</td>
                  <td>{{ grupo.nombreCarta }}</td>
              </tr>
            </table>
      </div>
</body>
</html>
```

El anterior código comprende, además de los tags del diseño anterior, lo siguiente:

- Una referencia a la librería JS de *Angular*. Esto se hace mediante el tag ***<script>…</script>*** utilizando el atributo ***src***
- Una referencia al archivo con código JS *"app.js"*. Este archivo aún no se ha codificado pero incluirá el toda la funcionalidad que controlará la aplicación. Estará basa en *Angular JS*

- Una referencia al archivo CSS con los estilos, el cual se ha denominado *"Estilos.css"*. Tampoco ha sido escrito aún, pero será el que dará formato a la presentación del documento. En este caso se utiliza la etiqueta ***<link />*** y los atributos ***href, rel*** y ***type***
- Para poder realizar la implementación del *Angular JS* se requieren los siguientes atributos en algunos de los tags:

Tag	Atributo	Descripción
html	***ng-app***	Define el elemento raíz de la aplicación *Angular JS.* Este atributo puede ir en cualquier elemento agrupador y dependiendo de este depende el alcance. Al ponerlo en el tag *html* implica que afectará todo el documento
body	***ng-controller***	Define el objeto controlador para la aplicación. Su definición debe estar en el código JS que controlará la aplicación
Cualquier tag visible	***ng-click***	Define el código a ejecutar cuando se hace clic en el elemento que incluye este atributo
Cualquier tag visible	***ng-repeat***	Define una plantilla para cada dato de una colección

- Para utilizar el atributo *ng-repeat* deben ser especificados los siguientes datos:
  - Directamente en el atributo debe ir el nombre que se dará a cada instancia a ser repetida y el nombre de la colección a la cual pertenece (Este debe estar declarado en el código del controlador), separadas por la palabra *in*

```
instancia in colección
```

  - En al menos alguno de los elementos a ser repetidos debe ir alguna de las propiedades o métodos de la instancia. Esta se especifica en medio de doble llaves:

```
{{ instancia.propiedad }}
```

Lo anterior se puede evidenciar en el código del documento *HTML* en dos casos distintos:

- Se tiene una colección de cartas que debe ser mostrada cuando se haga clic en el botón *"Repetir"*. En este caso se tiene una lista no ordenada y el elemento a repetir es el correspondiente al tag ***
…
***. Quiere decir que el atributo *ng-repeat* debe ir en él.
  El elemento contenido en este tag es una imagen la cual corresponde al tag ***<img />***. En este caso el método de la instancia que se repite es `carta.obtenerImagen()` el cual se asigna al

atributo ***src*** para indicar que es la ruta del archivo con la imagen de la carta a ser mostrada

- Cuando se haga clic al botón *"Chequear"* se debe mostrar la lista de grupos que forma el anterior conjunto de cartas. En este caso se desplegará una tabla donde cada fila es un grupo encontrado y habrá dos columnas: una para indicar el tipo de grupo y la otra el nombre de carta del respectivo grupo.
  El elemento a repetir en este caso es el que corresponde al tag de las filas de la tabla ***<tr>…</tr>***. Por lo tanto en este irá el atributo *ng-repeat*.
  Los elementos contenidos serán las 2 celdas de la fila que corresponden a los tag ***<td>…</td>***. El contenido de cada celda corresponderá a las propiedades `tipo` y `nombreCarta` de la instancia `grupo` respectivamente.

- El atributo *ng-click* se incluye en los dos botones que activarán las acciones. Por lo tanto en cada tag ***<a>…***. aparece este atributo cuyo valor es el método del controlador a ser invocado cuando se haga clic en cada uno:
  - Para el botón identificado como *"btnRepartir"* el método es `repartir()`
  - Para el botón identificado como *"btnVerificar"* el método es `chequear()`

Se continua con la codificación del código controlador de la aplicación, el cual irá en el archivo *app.js*. Este código debe incluir lo siguiente:

- Declaración del módulo principal de la aplicación mediante la instrucción `angular.module()`
- Declaración del constructor del controlador mediante la función `controller()` del módulo principal
- Como parámetro de la anterior función, se debe pasar el objeto ***$scope***. La finalidad de este objeto es establecer el enlace entre el código *HTML* (vista) y el código JavaScript (controlador).
  En este objeto se deben declarar las propiedades y métodos disponibles. Para este ejercicio tenemos:

Propiedad o método	Descripción
cartas	Vector de objetos de la clase *Carta* que representa las cartas repartidas
grupos	Vector con los grupos encontrados entre las cartas repartidas
repartir()	Método que simula la repartición de las cartas. Para ello crea las instancias de las 10 cartas y deja limpio el vector de los grupos. Cuando se ejecuta afecta los *ng-repeat* de la vista.
chequear()	Método que calcula los grupos que se encuentran en las cartas generadas. Afecta el *ng-repeat* de la tabla donde se despliegan.

	La estrategia para hallar los grupos consiste en contar cuantas veces aparece cada nombre de carta (sin tener en cuenta la pinta). Luego sólo se incluyen en el vector los contadores que sean mayores o iguales a 2.

El siguiente sería el código que corresponde al controlador:

```
angular.module('JuegoCartas', [])
	.controller('ControlJuegoCartas', function ($scope) {
	// Conjunto de cartas
	$scope.cartas = [];
	//Lista de grupos
	$scope.grupos=[];

	//Metodo que simula la repartición de las cartas
	$scope.repartir = function () {
		for (var i = 0; i < 10; i++) {
			$scope.cartas[i] = new Carta();
		}
		$scope.grupos=[];
	};
	//Metodo que obtiene los grupos encontrados en las cartas
repartidas
	$scope.chequear = function () {
		$scope.grupos=[];
		//declarar los contadores
		var contadores = [];
		for (var i = 0; i < 13; i++) {
			contadores[i]=0;
		}

		//recorrer las cartas y contar su aparición
		for (var i = 0; i < $scope.cartas.length; i++) {
			contadores[$scope.cartas[i].obtenerIndiceNombre()]++;
		}

		//Declarar los nombres de las grupos y las cartas
		var tipoGrupo = ["Par", "Terna", "Cuarta", "Quinta",
			"Sexta", "Septima", "Octava", "Novena", "Decima"];

		var nombre = ["As", "2", "3", "4", "5", "6", "7", "8", "9",
"10", "Jack", "Queen", "King"];

		//recorrer los contadores para verificar las grupos
		var grupo = {tipo : "Tipo de Grupo", nombreCarta : "Nombre
de Carta"};
		$scope.grupos.push(grupo);
		var tGrupos=0;
		for (var i = 0; i < 13; i++) {
			if (contadores[i] >= 2) {
				grupo = {tipo : tipoGrupo[contadores[i] - 2],
nombreCarta : nombre[i]};
				$scope.grupos.push(grupo);
```

```
                    }
                }
                if($scope.grupos.length==0)
                {
                    grupo = {tipo : "No se encontraron", nombreCarta :
""};
                    $scope.grupos.push(grupo);
                }
            }

});
```

Por último se editará el archivo correspondiente a los estilos *CSS* que serán aplicados a la vista (documento *HTML*).

La lista de nombres de estilo será la siguiente:

Nombre del Estilo	Descripción
html	Afecta a todo el documento y es reutilizable. En este caso estable una imagen de fondo que cubrirá toda la ventana del navegador
#Titulo	Solo afecta el elemento identificado con el nombre *"Titulo"*. En este caso establece que la sección definida mediante el tag *** … *** tenga una apariencia de cristal ahumado ubicado en la parte inferior del navegador. El texto irá centrado dentro de la sección, con fuente "*Verdana 25pt"* y color blanco
.botones	Afecta cualquier elemento que lo incluya mediante el atributo ***class*** y es reutilizable. En el ejercicio afectará la sección que incluye los botones y define que iría a 30 pixeles del borde superior y con los objetos centrados
#Mesa	Solo afecta el elemento identificado con el nombre *"Mesa"*. Trata de darle un estilo metálico a la sección identificada con este nombre la cual contendrá la colección de cartas y la lista de grupos encontrados
.boton	Afecta cualquier elemento que lo incluya mediante el atributo ***class*** y es reutilizable. En este caso se le invocará en los elementos ancla (tag ***<a>…***) los cuales representan los botones de la aplicación. Este estilo dará la apariencia de botón a dichos elementos
.boton:active	Es un estilo complementario al anterior y que responde un evento que sucede con el elemento, en este caso cuando se activa. Se define para cambiar la apariencia del botón cuando se presiona, en este caso simular un hundimiento y una iluminación.
.boton:hover	También es un estilo complementario del botón y sucede cuando el puntero del mouse pasa por encima

.carta	Afecta cualquier elemento que lo incluya mediante el atributo ***class*** y es reutilizable. En este caso es para indicar que los elementos de la lista de cartas no se desplegarán verticalmente como es el valor predeterminado sino horizontalmente.
table, th, td	Afecta a cualquier elemento de estos tipos que haya. En este caso es a la tabla que listará los grupos encontrados entre las cartas
table tr:nth-child(odd)	Afecta a las filas impares de la tabla
table tr:nth-child(even)	Afecta a las filas pares de la tabla

Es importante distinguir en el nombre que se le da a cada estilo lo siguiente:

- Si comienza con # significa que es el ID de un elemento del documento y sólo se aplica a este
- Si comienza con . (punto) es porque para poder aplicarse a un elemento, debe ser asignado al respectivo atributo ***class.***
- En caso de no comenzar con ningún carácter significa que debe ser algún tag del *HTML*

El siguiente sería el código que corresponde a los estilos antes mencionados:

```
html {
      background: url("../Imagenes/Fondo.jpg") no-repeat; /* Fondo */
      background-size: cover; /* Cubrir el navegador */
}

#Titulo{
      /*** Apariencia ***/
      background: rgba(0, 0, 0, 0.4); /* Fondo negro transparente */
      height: 150px; /* Alto */
      /* Caja con sombra transparente exterior negra e interior blanca
*/
      box-shadow: 0 0 25px rgba(0, 0, 0, 0.5), inset 0 0 25px
rgba(255,255,255, 0.5);

      /*** Texto ***/
      color: rgba(255, 255, 255, 0.7); /* Color blanco opaco */
      text-align: center; /* Centrado */
      font: 25pt Verdana; /* Tipo de letra */
      line-height: 150px;

      /*** Ubicación***/
      position: absolute;  /* Ubicación fija */
      bottom: 20px;  /* Distancia del borde inferior de 20 pixeles */
      right: 20%; /* Distancia del borde derecho del 20% del ancho */
      left: 20%; /* Distancia del borde izquierdo del 20% del ancho */
}

.botones{
      margin-top: 30px; /* Distancia del borde superior de 30 pixeles */
      text-align: center; /* Centrar objetos contenidos */
}
```

```
#Mesa {
      position: absolute;  /* Ubicación fija */
      top: 75px;  /* Distancia del borde superior de 75 pixeles */

      right: 20%; /* Distancia del borde derecho del 20% del ancho */
      left: 20%; /* Distancia del borde izquierdo del 20% del ancho */
      height: 300px; /* alto del objeto */
      padding: 10px; /* Espacio alrededor de los objetos contenidos */
      margin: 20px auto; /* Espacio entre los elementos alrededor */
      text-align: center; /* Alineación de los objetos contenidos */

      /* Sombra del área*/
      -webkit-box-shadow: 0px 0px 1px 0px rgba(0,0,0,1), 0px 3px 15px
2px rgba(0,0,0,1);
      -moz-box-shadow: 0px 0px 1px 0px rgba(0,0,0,1), 0px 3px 15px 2px
rgba(0,0,0,1);
      box-shadow: 0px 0px 1px 0px rgba(0,0,0,1), 0px 3px 15px 2px
rgba(0,0,0,1);

      /* Borde radial */
      border: 1px solid #5c5c5c;
      border-top: 1px solid #999;
      -webkit-border-radius: 5px;
      -moz-border-radius: 5px;
      border-radius: 5px;
      background: rgba(0,0,0,0.8); /* Color del fondo negro con
transparencia */
      /* Relleno de fondo: Lineas en diagonal*/
      background-image:                repeating-linear-gradient(45deg,
rgba(255,255,255,0.05) 0%, rgba(255,255,255,0.05) 25%, transparent 25%,
transparent 50%);
      background-image:         -webkit-repeating-linear-gradient(45deg,
rgba(255,255,255,0.05) 0%, rgba(255,255,255,0.05) 25%, transparent 25%,
transparent 50%);
      background-image:            -moz-repeating-linear-gradient(45deg,
rgba(255,255,255,0.05) 0%, rgba(255,255,255,0.05) 25%, transparent 25%,
transparent 50%);
      background-image:             -ms-repeating-linear-gradient(45deg,
rgba(255,255,255,0.05) 0%, rgba(255,255,255,0.05) 25%, transparent 25%,
transparent 50%);
      background-image:              -o-repeating-linear-gradient(45deg,
rgba(255,255,255,0.05) 0%, rgba(255,255,255,0.05) 25%, transparent 25%,
transparent 50%);
      /* Dimension del fondo */
      -webkit-background-size: 5px 5px;
      -moz-background-size: 5px 5px;
      -o-background-size: 5px 5px;
      background-size: 5px 5px;
}

.boton {
      font-size: 28px; /* Tamaño de la fuente */
      background-color: #434343; /* Color de fondo */
```

```
      /* Relleno de fondo */
      background-image:    -webkit-linear-gradient(100%    100%    90deg,
#515151, #7A7A7A);
      background-image:  -moz-linear-gradient(100%  100%  90deg,  #515151,
#7A7A7A);
      background-image:  -o-linear-gradient(100%  100%  90deg,  #515151,
#7A7A7A);
      background-image:  -ms-linear-gradient(100%  100%  90deg,  #515151,
#7A7A7A);
      background-image:   linear-gradient(100%   100%   90deg,   #515151,
#7A7A7A);
      background-image:   -webkit-gradient(linear,   0%   0%,   0%   100%,
from(#7A7A7A), to(#515151));

      /* Borde radial */
      border: none;
      border-top: 3px solid #c2c2c2;
      -webkit-border-radius: 40px;
      -moz-border-radius: 40px;
      border-radius: 40px;

      /* Sombra del botón*/
      -webkit-box-shadow: inset 0 1px 2px rgba(255,255,255,0.2), 0 1px 0
#2D2D2D, 0 2px 0 #2D2D2D, 0 3px 0 #2C2C2C, 0 4px 0 #2A2A2A, 0 0 0 6px
black, 0 4px 0 6px black, 0 0 0 7px #222, 0 4px 0 7px #222;
      -moz-box-shadow: inset 0 1px 2px rgba(255,255,255,0.2), 0 1px 0
#2D2D2D, 0 2px 0 #2D2D2D, 0 3px 0 #2C2C2C, 0 4px 0 #2A2A2A, 0 0 0 6px
black, 0 4px 0 6px black, 0 0 0 7px #222, 0 4px 0 7px #222;
      box-shadow:  inset  0  1px  2px  rgba(255,255,255,0.2),  0  1px  0
#2D2D2D, 0 2px 0 #2D2D2D, 0 3px 0 #2C2C2C, 0 4px 0 #2A2A2A, 0 0 0 6px
black, 0 4px 0 6px black, 0 0 0 7px #222, 0 4px 0 7px #222;
      padding: 10px; /* Espacio alrededor de los objetos contenidos */
      text-shadow: 0 1px 0 rgba(255,255,255,0.2); /* Sombra del texto */
      margin-right: 10px; /* Margen derecha */
      text-decoration:  none;  /*  Elimina  el  formato  que  tiene  el
navegador para los hipervínculos */
      color: #242424; /* Color texto */

      /* Relleno de fondo adicional */
      background-image: -webkit-radial-gradient(   50%    0%,   8%  50%,
hsla(0,0%,100%,.5) 0%, hsla(0,0%,100%,0) 100%),
      -webkit-radial-gradient(   50%  100%,  12%  50%,  hsla(0,0%,100%,.6)
0%, hsla(0,0%,100%,0) 100%),                                  -webkit-
radial-gradient(      0%    50%,   50%    7%,   hsla(0,0%,100%,.5)   0%,
hsla(0,0%,100%,0) 100%),                                      -webkit-
radial-gradient(   100%    50%,   50%    5%,   hsla(0,0%,100%,.5)   0%,
hsla(0,0%,100%,0) 100%),
                        -webkit-repeating-radial-gradient( 50% 50%, 100%
100%, hsla(0,0%,  0%,0) 0%, hsla(0,0%,  0%,0)   3%, hsla(0,0%,  0%,.1)
3.5%),                                  -webkit-repeating-radial-
gradient(  50% 50%, 100% 100%, hsla(0,0%,100%,0) 0%, hsla(0,0%,100%,0)
6%, hsla(0,0%,100%,.1) 7.5%),                                 -webkit-
repeating-radial-gradient(   50% 50%, 100% 100%, hsla(0,0%,100%,0) 0%,
hsla(0,0%,100%,0) 1.2%, hsla(0,0%,100%,.2) 2.2%),
```

```
                                                    -webkit-radial-gradient(
      50% 50%, 200% 50%, hsla(0,0%,90%,1) 5%, hsla(0,0%,85%,1) 30%,
hsla(0,0%,60%,1) 100%);
}

.boton:active {
      border-top: 0px solid #dde1e7; /* Borde */
      /* Sombra del botón*/
      -webkit-box-shadow: inset 0 1px 2px rgba(255,255,255,0.2), 0 1px 0
#2D2D2D, 0 2px 0 #2D2D2D, 0 3px 0 #2C2C2C, 0 4px 0 #2A2A2A, 0 0 0 6px
black, 0 4px 0 6px black, 0 0 0 7px #222, 0 4px 0 7px #222,0px 0px 5px
#00aeff,0px 0px 50px #00aeff,0px 0px 50px #93d9fa;
      -moz-box-shadow: inset 0 1px 2px rgba(255,255,255,0.2), 0 1px 0
#2D2D2D, 0 2px 0 #2D2D2D, 0 3px 0 #2C2C2C, 0 4px 0 #2A2A2A, 0 0 0 6px
black, 0 4px 0 6px black, 0 0 0 7px #222, 0 4px 0 7px #222,0px 0px 5px
#00aeff,0px 0px 50px #00aeff,0px 0px 50px #93d9fa;
      box-shadow: inset 0 1px 2px rgba(255,255,255,0.2), 0 1px 0
#2D2D2D, 0 2px 0 #2D2D2D, 0 3px 0 #2C2C2C, 0 4px 0 #2A2A2A, 0 0 0 6px
black, 0 4px 0 6px black, 0 0 0 7px #222, 0 4px 0 7px #222,0px 0px 5px
#00aeff,0px 0px 50px #00aeff,0px 0px 50px #93d9fa;
      /* Reubicación del botón */
      -webkit-transform: translateY(4px);
      -moz-transform: translateY(4px);
      -ms-transform: translateY(4px);
      -o-transform: translateY(4px);
      transform: translateY(4px);
}

.boton:hover {
      color: #00aeff; /* Color del texto */
      text-shadow: -1px -2px 1px #000; /* Sombra del texto */
}

.carta{
      display: inline; /* Distribuye horizontalmente los objetos */
}

table, th , td  {
  border: 1px solid grey; /* caracteristicas del  borde */
  border-collapse: collapse; /* los bordes no se separan */
  padding: 5px; /* Espacio alrededor de los objetos contenidos */
  margin: 0 auto; /* Ayuda a centrar en el contenedor */
}
table tr:nth-child(odd) {
  background-color: #f1f1f1; /* Color de fila impar gris */
}
table tr:nth-child(even) {
  background-color: #ffffff; /* Color de fila par blanco */
}
```

En el anterior código es importante observar como algunos estilos se repiten pero cada uno viene con un prefijo distinto. El siguiente es el significado de estos prefijos:

Prefijo	Descripción
-webkit	Aplica al navegador *Safari, Chrome* y ultimas versiones de *Opera*
-ms	Aplica al navegador *Internet Explorer*
-moz	Aplica al navegador *Mozilla*
-o	Aplica al navegador *Opera* (versiones anteriores)

Editado lo anterior, basta con ejecutar el documento *HTML* en algún navegador, lo cual tendría el siguiente aspecto:

2. El juego de azar "*Bingo*" se realiza entre varios jugadores y un "*cantor*". Cada jugador debe tener una tabla que viene dividida en cuadrados numerados. La tabla tiene 5 columnas y 5 filas. Las columnas están encabezadas con las letras "B","I","N","G" y "O", cada una de las cuales alberga debajo 5 números comprendidos entre los siguientes rangos:

- ***"B"***: 1 a 15
- ***"I"***: 16 a 30
- ***"N"***: 31 a 45
- ***"G"***: 46 a 60
- ***"O"***: 61 a 75

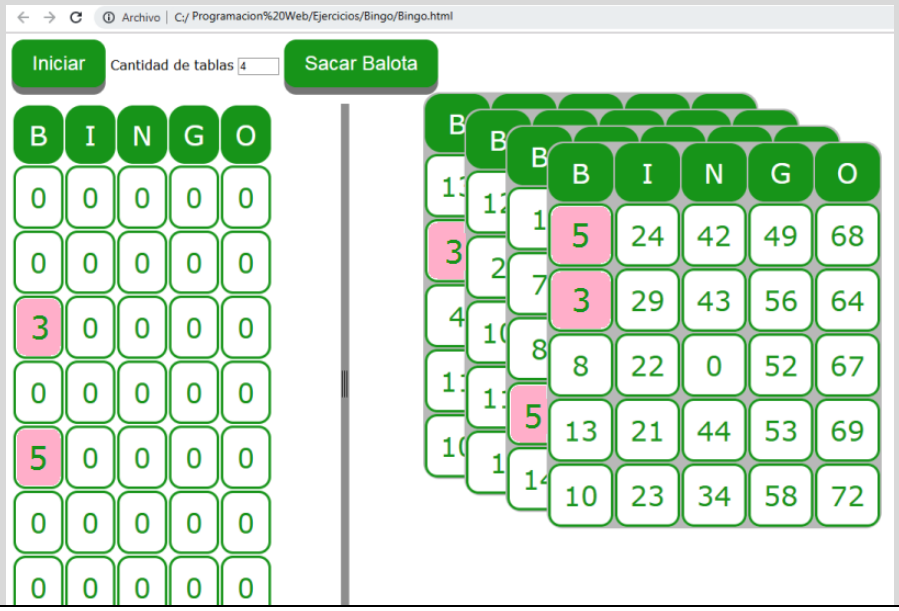
A la derecha se tiene un ejemplo de una tabla. Se puede observar que el centro de la tabla no tiene número.

El Bingo es un juego muy similar a la Lotería. La persona denominada "Cantor" saca al azar balotas con números del 1 al 75. Las anuncia y los participantes del juego deben marcar los números si es que los tienen en sus respectivas tablas.

El objetivo de este juego es marcar todos los números de la tabla ("*Bingo*") o completar una línea horizontal, vertical o diagonal (*"Binguito*"). El jugador que lo logre, grita "Bingo!" y gana un premio.

Se requiere realizar un aplicativo web (solo se necesita que funcione en el lado del cliente, es decir, no requiere de servidor web) que permita simular la ejecución del juego:

- Debe tener un botón que inicie el juego basado en una cantidad de tablas que se debe proporcionar
- Incluir otro botón que permita sacar en caja acción una balota y las actualice tanto en la lista del cantor como en las diferentes tablas.
  En esta misma actualización debe también detectar que alguna tabla haya completado un bingo o un binguito



Para la solución de este ejercicio se sugiere el siguiente modelo (diagrama de clases) del aplicativo bajo el paradigma Orientado a Objetos: