**DESARROLLO DE VIDEOJUEGO CON TECNOLOGIA HTML Y JAVASCRIPT**

**Hebert De La Cruz Moreno**

***UNAJMA. Universidad nacional José María Arguedas***

[**hebertdtb779@gmail.com**](mailto:hebertdtb779@gmail.com)**,**

[**hebertdtb777@hotmail.com**](mailto:%20hebertdtb777@hotmail.com)**,**

**Resumen**

El principal objetivo de este proyecto es el desarrollo del VideoJuego, que permita realizar el juego que se emplee la modalidad de un juego de un jugador. Con este objetivo se decidido desarrollar un software que permita, mediante un interfaz que el usuario manipule y visualicé el real y potencial de este juego de entretenimiento en web, el juego que tenge 3 niveles,

**INTRODUCCION**

**CANVAS COMO SUPERFICIE GRÁFICA**

En el pasado, la creación de efectos gráficos y animaciones no resultaba sencilla dentro de páginas web. Un recurso que fue muy utilizado fue insertar applets Java u objetos o películas flash (archivos swf), con el inconveniente de que esto implicaba hacer uso de Flash Player, un plugin adicional que se debía instalar en los navegadores. La tendencia actual es que la animación gráfica quede dentro de los estándares web definidos por HTML, JavaScript y CSS sin dependencias de terceras partes.



 Dentro del desarrollo del estándar de HTML se ha incluido una nueva etiqueta o elemento HTML: <canvas>. Esta etiqueta funciona igual que cualquier otra en el sentido de que se abre como <canvas> y se cierra como </canvas>.

El elemento canvas es un lienzo encima del cual se pueden dibujar gráficos y animaciones. Permite generar gráficos en pantalla, crear animaciones, manipular imágenes e incluso video, etc. En este sentido, permite generar las animaciones gráficas necesarias utilizando HTML+JavaScript+CSS, de modo que todos los desarrollos web puedan completarse con lo que se denominan lenguajes web sin necesidad de recurrir a tecnologías de terceras partes como Flash o los applets de Java.

La incorporación de toda esta potencia gráfica corre a cargo de los navegadores. Dada la complejidad que conlleva, todo lo relacionado con canvas se dice que constituye una API (interfaz de programación de aplicaciones). Algunos navegadores antiguos no admiten los elementos <canvas>, pero todos los navegadores modernos sí lo hacen.

El elemento canvas constituye un soporte al que se le da uso a través de JavaScript.

Nosotros en este curso no vamos a estudiar la api Canvas. Simplemente nos limitaremos a citar algunas de las posibilidades que el uso de este elemento junto a JavaScript permite para los desarrollos web. La creación de elementos gráficos avanzados y animaciones avanzadas constituye un área de especialización dentro de los desarrollos web, por tanto no lo consideramos un conocimiento “básico”. No obstante, es adecuado tener una idea de qué se puede hacer con canvas y JavaScript.

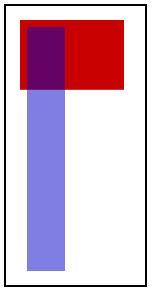
**UTILIZANDO CANVAS**

Un elemento HTML canvas tiene propiedades y métodos que podemos usar a través de JavaScript, de la misma forma que usamos propiedades y métodos sobre el resto de elementos HTML como <img>, <input>, <label>, etc.

No vamos a detenernos a estudiar el API Canvas, pero queremos mostrar brevemente algunos ejemplos que nos den una idea de qué es y para qué sirve. Escribe este código en en editor de textos y guárdalo con un nombre de archivo como ejemplo.html:

|  |
| --- |
| <!DOCTYPE html>  <html>   <head><title>Ejemplo canvas aprenderaprogramar.com</title>    <meta charset="utf-8"/>    <script type="text/javascript">      function dibujar() {        var canvas = document.getElementById("canvas");          var ctx = canvas.getContext("2d");          ctx.fillStyle = "rgb(200,0,0)";          ctx.fillRect (20, 20, 150, 100);          ctx.fillStyle = "rgba(0, 0, 200, 0.5)";          ctx.fillRect (30, 30, 55, 350);      }    </script>   </head>   <body onload="dibujar();">     <canvas id="canvas" width="200" height="400" style="border-style:solid;">     Su navegador no tiene soporte para mostrar el contenido     </canvas>   </body>  </html> |

 El resultado obtenido será algo similar a esto:



 Si investigas un poco sobre el código podrás comprender con facilidad qué es lo que hace. No vamos a entrar en detalles porque no es el objetivo de este curso.

Quizás pienses que esto es relativamente poco útil, sin embargo hay un gran potencial detrás del elemento canvas unido a la potencia de JavaScript. La creación anterior implica que generamos una imagen a partir de código, esto implica un potencial enorme, ya que la imagen puede estar ligada a datos (introducidos por el usuario, extraídos de una base de datos, tomados de otra página web, etc.), y también ligada a eventos (lo cual permite crear movimientos en respuesta a la acción del usuario, incluso crear juegos) o ligada al tiempo (animaciones, imágenes que van cambiando en el tiempo).

**¿PARA QUÉ SIRVE CANVAS?**

El api Canvas combinado con CSS, HTML y JavaScript nos permite hacer todo aquello que se nos pueda ocurrir. Vamos a citar algunas de las aplicaciones habituales:

a) Dibujar formas (líneas, puntos, círculos, etc.)

b) Dibujar texto

c) Dibujar encima de imágenes o manipular imágenes

d) Dibujar gráficos

e) Crear animaciones de todo tipo (matemáticas, escolares, publicidad, etc.).

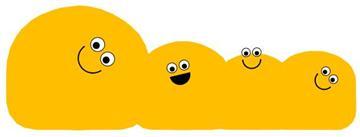
f) Crear espacios tridimensionales por donde el usuario puede moverse

g) Crear juegos

h) Manipular videos por ejemplo crear efectos gráficos sobre ellos

i) Hacer todo lo anterior en respuesta a eventos (por interacción con el usuario)

j) Hacer todo lo anterior ligado a datos variables



 Las bolas de la imagen anterior se pueden dibujar con relativa facilidad con Canvas. También se pueden dotar de animación, hacer que salten, se dividan, etc.

Si escribes canvas examples en un buscador como google o bing podrás acceder a numerosos ejemplos de uso de canvas. Muchos de ellos facilitan el código fuente de forma ordenada para que te sea fácil reproducirlo en tu ordenador o incorporarlo a tus páginas web.

**VENTAJAS E INCONVENIENTES DE CANVAS**

La principal ventaja de Canvas es permitir la creación y manipulación de todo tipo de formas e imágenes sin dependencias de plugins o tecnologías externas a los propios lenguajes de desarrollo web y navegadores. Igualmente es una gran ventaja la facilidad que brinda para la interacción gráfica con el usuario o para ligar el aspecto gráfico a datos que cambian dinámicamente.

Canvas es una herramienta que, al igual que todo, bien usado resulta muy útil y mal usado puede generarnos problemas. Un inconveniente ligado a canvas es pretender hacer páginas webs muy animadas y con esto sobrecargar el computador del usuario. Ten en cuenta que crear animaciones consume recursos y puede ralentizar la navegación por las páginas web o crear molestias al usuario.

**6.2 Aplicación Web**

**6.2.1 ¿Qué es una aplicación Web?**

Las aplicaciones Web son aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor Web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores Web en la que se confía la ejecución al navegador.

Las aplicaciones Web son populares debido a lo práctico del navegador Web como cliente ligero, a la independencia del sistema operativo, así como a la facilidad para actualizar y mantener aplicaciones Web sin distribuir e instalar software a miles de usuarios potenciales. Existen aplicaciones como los webmails, wikis, weblogs, tiendas en línea y la propia Wikipedia que son ejemplos bien conocidos de aplicaciones Web.

Es importante mencionar que una página Web puede contener elementos que permiten una comunicación activa entre el usuario y la información. Esto permite que el usuario acceda a los datos de modo interactivo, gracias a que la página responderá a cada una de sus acciones, como por ejemplo rellenar y enviar formularios, participar en juegos diversos y acceder a gestores de base de datos de todo tipo.

A diferencia de los primeros tiempos de la computación cliente servidor, donde cada aplicación tenía que ser instalado independientemente en cada ordenador personal y los programas clientes servían como interfaz de usuario, las aplicaciones Web generan dinámicamente una serie de páginas en un formato estándar, como HTML o XHTML, soportados por los navegadores Web comunes. Se utilizan lenguajes interpretados en el lado del cliente, directamente o a través de plugins tales como JavaScript, Java, Flash, etc., para añadir elementos dinámicos a la interfaz de usuario. Generalmente cada página Web en particular se envía al cliente como un documento estático, pero la secuencia de páginas ofrece al usuario una experiencia interactiva. Durante la sesión, el navegador Web interpreta y muestra en pantalla las páginas, actuando como cliente para cualquier aplicación Web.

Las interfaces Web tienen ciertas limitaciones en las funcionalidades que se ofrecen al usuario. Hay funcionalidades comunes en las aplicaciones de escritorio como dibujar en la pantalla o arrastrar-y-soltar que no están soportadas por las tecnologías Web estándar. Los desarrolladores Web generalmente utilizan lenguajes interpretados (scripts) en el lado del cliente para añadir más funcionalidades, especialmente para ofrecer una experiencia interactiva que no requiera recargar la página cada vez (lo que suele resultar molesto a los usuarios). Recientemente se han desarrollado tecnologías para coordinar estos lenguajes con las tecnologías en el lado del servidor. Como ejemplo, el mencionado anteriormente AJAX es una técnica de desarrollo Web que usa una combinación de varias tecnologías.

***Ventajas***

 **Ahorra tiempo:** Se pueden realizar tareas sencillas sin necesidad de descargar ni instalar ningún programa.

 **No hay problemas de compatibilidad:** No hace falta crear diferentes clientes el función de cada sistema operativo. Basta tener un navegador actualizado para poder utilizarlas.

 **No ocupan espacio en nuestro disco duro.**

 **Actualizaciones inmediatas:** Como el software lo gestiona el propio desarrollador, cuando nos conectamos estamos usando siempre la última versión que haya lanzado.

 **Consumo de recursos bajo:** Dado que toda (o gran parte) de la aplicación no se encuentra en nuestro ordenador, muchas de las tareas que realiza el software no consumen recursos nuestros porque se realizan desde otro ordenador.

 **Multiplataforma:** Se pueden usar desde cualquier sistema operativo porque sólo es necesario tener un navegador.

 **Portables:** Es independiente del ordenador donde se utilice (un PC de sobremesa, un portátil...) porque se accede a través de una página Web (sólo es necesario disponer de acceso a Internet). La reciente tendencia al acceso a las aplicaciones Web a través de teléfonos móviles requiere sin embargo un diseño específico de los ficheros CSS para no dificultar el acceso de estos usuarios.

 **La disponibilidad** suele ser alta porque el servicio se ofrece desde múltiples localizaciones para asegurar la continuidad del mismo.

 **Los virus no dañan los datos** porque éstos están guardados en el servidor de la aplicación.

 **Colaboración:** Gracias a que el acceso al servicio se realiza desde una única ubicación es sencillo el acceso y compartición de datos por parte de varios usuarios. Tiene mucho sentido, por ejemplo, en aplicaciones online de calendarios u oficina.

***Inconvenientes***

 Habitualmente ofrecen menos funcionalidades que las aplicaciones de escritorio. Se debe a que las funcionalidades que se pueden realizar desde un navegador son más limitadas que las que se pueden realizar desde el sistema operativo. Pero cada vez los navegadores están más preparados para mejorar en este aspecto.

 Al igual que lo que mencionábamos previamente cuando estábamos tratando las pegas del Cloud Computing, aquí ocurre algo análogo. La disponibilidad depende de un tercero, el proveedor de la conexión a Internet o el que provee el enlace entre el servidor de la aplicación y el cliente. Así que la disponibilidad del servicio está supeditada al proveedor.

Los lenguajes de programación empleados para el desarrollo de apliaciones Web son diversos, pero destacan, entre otros, los siguientes: PHP, JavaScript, Perl, Ruby, Phyton, XML o ASP.NET. Aunque este último no es un lenguaje de programación como tal, sino una arquitectura de desarrollo Web en la que se pueden usar por debajo diferentes lenguajes.

**Agradecimientos (Times New Roman, 10, negrita)**

Si existiera, mencionarlos en forma concisa. Será escrito en fuente (Times New Roman, 10).

**Referencias (Times New Roman, 10, negrita).**

Documentación y bibliografía utilizada. Todas las publicaciones citadas deberán incluirse en la lista de referencias. La numeración será secuencial y estará entre corchetes: [1]. Será escrita en fuente (Times New Roman, 10)

**Notas a pie de página**

Las Notas de pie de página serán 9pto Times y aparecen al pie de la página correspondiente. La referencia numérica será en superscript.

**Datos de Contacto: (Times New Roman, 10, negrita)**

*Nombre y Apellido. Institución. Dirección postal. E-mail. Serán escritos en fuente (Times New Roman, 10, Cursiva)*

Tablas: Las Tablas serán numeradas en forma secuencial (Tabla 1, 2, 3, etc.,) con un titulo descriptivo, ambos escritos en Times New Roman, 10, cursiva. Las tablas estarán centradas en la columna o en caso contrario en toda la página. Los títulos de las columnas de la Tabla estarían en caracteres Times New Roman, 10 negrita.

Figuras: Las figuras serán numeradas secuencialmente: Figura 1, 2, 3, etc.).

Aquellas figuras que abarquen toda la página serán ubicadas al comienzo o al final de la página.

**Notas generales respecto a la presentación**:

* Cada Paper debería contener un máximo de 10 páginas y un mínimo de 5.
* El tamaño de la página debería ser de 21mm x 297mm (Tamaño A4).
* Los márgenes deben establecerse en 2,5 cm.
* Las páginas no deben numerarse.
* No agregue otros encabezados o notas al pié.
* El espaciado entre columnas debe ser de 1 cm.
* Las columnas deberían ser de igual ancho
* El texto debe ser color negro.