4.- Generación de elementos HTML desde código JavaScript.

- Uno de los principales objetivos de JavaScript es convertir un documento HTML estático en una aplicación web dinámica.
- Por ejemplo, es posible ejecutar instrucciones que crean nuevas ventanas con contenido propio, en lugar de mostrar dicho contenido en la ventana activa.

4.- Generación de elementos HTML desde código JavaScript.

- Con JavaScript es posible manipular los objetos que representan el contenido de una página web con el fin de crear documentos dinámicos.
- Por ejemplo, es posible definir el título de una página web basándose en el SO utilizado:

```
<script type="text/javascript">
     var SO = navigator.platform;
     document.write("<h1>Documento abierto con: " + SO +
"</h1>");
</script>
```

4.- Generación de elementos HTML desde código JavaScript.

- Otro ejemplo es crear documentos en ventanas emergentes:

```
<script type="text/javascript">
   var texto = prompt(“Introduce un título para la nueva ventana: ");
   var ventanaNueva= window.open();
   ventanaNueva.document.write("<h1>" + texto + "</h1>");
</script>
```

4.- Generación de elementos HTML desde código JavaScript.

- La generación de código HTML a partir de JavaScript no se limita sólo a la creación de texto como en los ejemplos anteriores. Es posible crear y manipular todo tipo de objetos
- Por ejemplo, generar un formulario para modificar la propiedad del color de fondo de la página:

4.- Generación de elementos HTML desde código JavaScript.

- Es decir, obtener la siguiente página web dinámica:

4.- Generación de elementos HTML desde código JavaScript.

```
<script type="text/javascript">
  document.write("<form name=\"cambiacolor\">");
  document.write("<b>Selecciona un color para el fondo de página:</b><br>");
  document.write("<select name=\"color\">");
  document.write("<option value=\"red\">Rojo</option>");
  document.write("<option value=\"blue\">Azul</option>");
  document.write("<option value=\"yellow\">Amarillo</option>");
  document.write("<option value=\"green\">Verde</option>");
  document.write("</select>");
  document.write("<input type=\"button\" value=\"Modifica el color\"
  onclick=\"document.bgColor=document.cambiacolor.color.value\">");
  document.write("</form>");
</script>
```

Actividad 6

Modifica el ejemplo anterior agregando dos colores más al conjunto de colores del fondo de la página.

5.- Aplicaciones prácticas de los marcos.

- Es posible dividir la ventana de una aplicación web en dos o más partes independientes.
- Con JavaScript se puede interactuar entre estos sectores independientes.
- Dichos sectores se denominan marcos.

5.- Aplicaciones prácticas de los marcos.

Algunas páginas web presentan una estructura en la cual una parte permanece fija mientras que otra va cambiando.

Por ejemplo la página de la API de Java:

http://docs.oracle.com/javase/6/docs/api/

5.- Aplicaciones prácticas de los marcos.

- Los marcos se definen utilizando HTML mediante estas etiquetas:
  - `<frameset>` : para indicar al navegador el número de marcos, tamaño, etc. Los principales atributos son `cols` y `rows`.
  - `<frame>`: esta etiqueta define las características de cada marco. Sus principales atributos son:

Atributos	Descripción
frameborder	Define si mostrar o no el borde del marco.
marginheight	Permite cambiar los márgenes verticales del marco.
marginwidth	Permite cambiar los márgenes horizontales del marco.
name	Asigna un nombre al marco.
noresize	Evita que el usuario pueda modificar el tamaño del marco.
scrolling	Permite elegir si posiciona o no una barra de desplazamiento en el marco.
scr	Indica la URL de documento HTML que contendrá el marco.

5.- Aplicaciones prácticas de los marcos.

- JavaScript permite manipular los marcos mediante las propiedades frames, parent y top del objeto window.
- Por ejemplo, se define un documento HTML con dos marcos:

```
<html><head><title>Ejemplos de control de
marcos</title></head>
    <frameset cols="50%,50%">
      <frame src="Marco1.html" name="Marco1" noresize>
      <frame src="Marco2.html" name="Marco2" noresize>
    </frameset>
  <body></body>
</html>
```

5.- Aplicaciones prácticas de los marcos.

- El primer marco (`Marco1`) contiene la página `Marco1.html`:

```
<html><body>
    <form name="form1">
      <select name="color">
        <option value="green">Verde
        <option value="blue">Azul
      </select><br><br>
      <select name="marcos">
        <option value="0">Izquierda
        <option value="1">Derecha
      </select>
    </form>
</body></html>
```

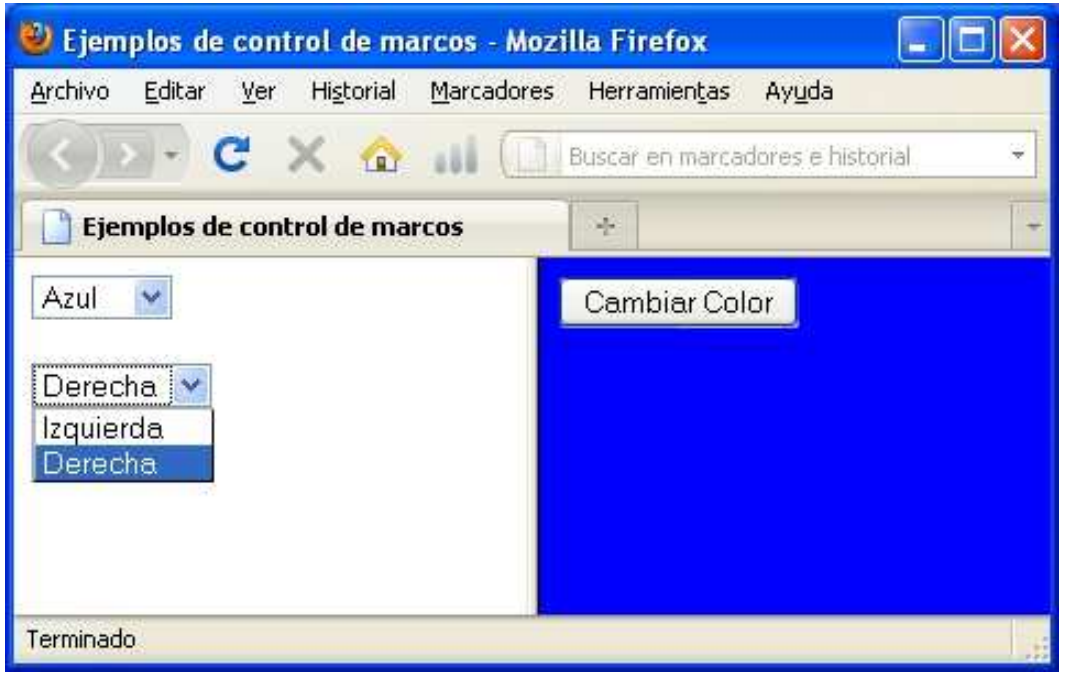
5.- Aplicaciones prácticas de los marcos.

- El segundo marco (`Marco2`) contiene la página `Marco2.html`:

```
<html><body><form>
      <input type="Button" value="Cambiar Color" onclick="
        campoColor = parent.Marco1.document.form1.color;
        if(campoColor.selectedIndex==0){colorin = 'green';}
        else{colorin = 'blue';}
        campoFrame = parent.Marco1.document.form1.marcos
        if(campoFrame.selectedIndex==0){
          window.parent.Marco1.document.bgColor = colorin
        }else{
          window.parent.Marco2.document.bgColor = colorin
        }">
</form></body></html>
```

5.- Aplicaciones prácticas de los marcos.

- El resultado se puede ver en esta imagen:



Actividad 7

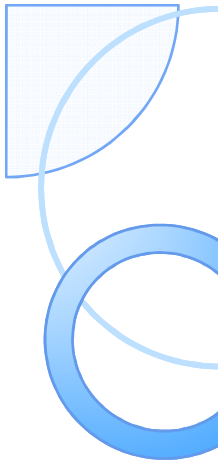
Modifica el código del ejemplo de marcos utilizado anteriormente, con el fin de ocultar el borde y que no se note la separación entre ellos.

6.- Gestión de las ventanas.

- JavaScript permite gestionar diferentes aspectos relacionados con las ventanas como por ejemplo abrir nuevas ventanas al presionar un botón.
- Cada una de estas ventanas tiene un tamaño, posición y estilo diferente.
- Estas ventanas emergentes suelen tener un contenido dinámico.

6.- Gestión de las ventanas.

- Abrir y cerrar nuevas ventanas:
  - Es una operación muy común en las páginas web.
  - En algunas ocasiones se abren sin que el usuario haga algo.
  - HTML permite abrir nuevas ventanas pero no permite ningún control posterior sobre ellas.



6.- Gestión de las ventanas.

- Abrir y cerrar nuevas ventanas:
  - Con JavaScript es posible abrir una ventana vacía mediante el método `open()`:

```
nuevaVentana = window.open();
```

De este modo la variable llamada `nuevaVentana` contendrá una referencia a la ventana creada.

6.- Gestión de las ventanas.

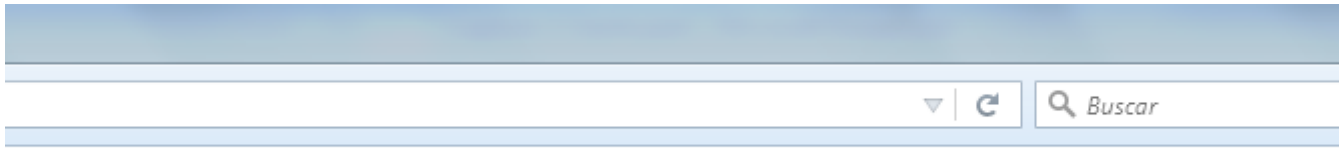
- Abrir y cerrar nuevas ventanas:
  - El método `open( )` cuenta con tres parámetros:
    - URL.
    - Nombre de la ventana.
    - Colección de atributos que definen la apariencia de la ventana.
  - Ejemplo:

```
nuevaVentana = window.open(“http://www.misitioWeb.com/ads”,
“Publicidad”, “height=100, widht=100”);
```

Actividad 8

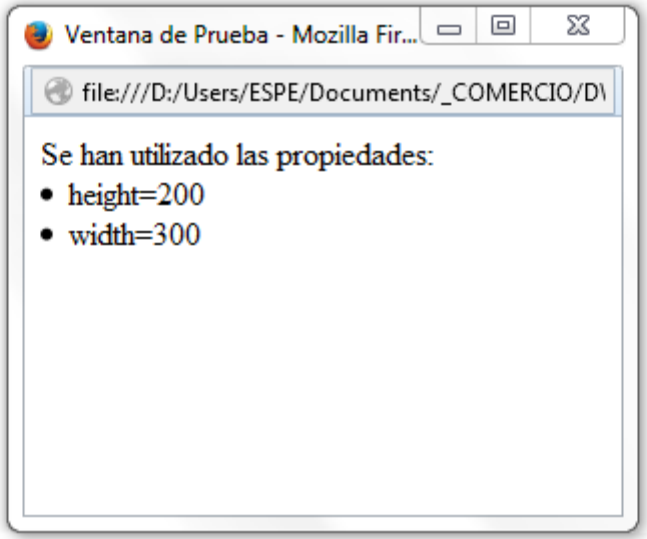
Crea un botón en una pagina web que abra una nueva ventana al hacer clic sobre él. En la nueva ventana, establecemos los atributos de altura y anchura, además de crear etiquetas HTML para generar el título de la ventana y un texto en el que especifiquemos las propiedades modificadas.

Actividad 8



Ejemplo de Apariencia de una Ventana

Abre una Ventana



6.- Gestión de las ventanas.

- Para cerrar una ventana se puede invocar el método close():

```
myWindow1.document.write('<input type=button
value=Cerrar onClick=window.close()>');
```

Realiza esta modificación a la Actividad 8 para añadir un botón de cerrar en la ventana nueva.

Actividad 9

Apertura de múltiples ventanas:

Crea un botón en una página web que abra cinco ventanas a la vez y que presenten a su vez un botón para poder cerrarlas.

6.- Gestión de las ventanas.

- Apariencia de las ventanas:
  - La ventanas cuentan con propiedades que permiten decidir su tamaño, ubicación o los elementos que contendrá.

Propiedad	Descripción
directories	Corresponde a los botones del directorio estándar del navegador.
height	Corresponde a la altura de la ventana.
menubar	Corresponde a la barra del menú.
resizable	Corresponde a la opción de cambiar o no el tamaño de la ventana.
scrollbar	Corresponde a las barras de desplazamiento.
status	Corresponde a la barra de estado.
toolbar	Corresponde a la barra de herramientas.
width	Corresponde a la anchura de la ventana.

6.- Gestión de las ventanas.

- Comunicación entre ventanas:
  - Desde una ventana se pueden abrir o cerrar nuevas ventanas.
  - La primera se denomina ventana principal, mientras que las segundas se denominan ventanas secundarias.
  - Desde la ventana principal se puede acceder a las ventanas secundarias.

6.- Gestión de las ventanas.

- Comunicación entre ventanas:
  - En el siguiente ejemplo se muestra cómo acceder a una ventana secundaria:

```
<html><head></head><body>
  <script type="text/javascript">
   var ventanaSecundaria = window.open("", "ventanaSec", "width=500,
   height=500");
  </script>
  <center><h1> Comunicaci&oacute;n entre ventanas </h1><br>
  <form name=formulario>
   <input type=text name=url size=50 value="http://www.">
   <input type=button value="Mostrar URL en ventana secundaria"
  onclick="ventanaSecundaria.location = document.formulario.url.value;">
  </form></center></body></html>
```

6.- Gestión de las ventanas.

- Comunicación entre ventanas:
  - En el ejemplo anterior, se está enviando la información de la ventana principal a la ventana secundaria.
  - Cuando necesitemos hacer referencia al objeto window que haya abierto una ventana nueva, es decir, a la ventana padre, utilizaremos la propiedad opener del objeto window.

Actividad 10

Modifica el ejemplo anterior para que la comunicación entre las dos ventanas sea bidireccional. Es decir, que la url que se escriba en la ventana principal, se muestre en la ventana secundaria y viceversa.