ANEXO

En el siguiente anexo se facilitan algunos de los métodos y propiedades de los objetos de JavaScript que pueden usarse para resolver los ejercicios.

OBJETO DATE

Se utiliza para trabajar con fechas y horas. Los objetos Date se crean con new Date().

Hay 4 formas de instanciar (crear un objeto de tipo `Date`):

```
var d = new Date();
var d = new Date(milisegundos);
var d = new Date(cadena de Fecha);
var d = new Date(año, mes, día, horas, minutos, segundos, milisegundos);
     // (el mes comienza en 0, Enero sería 0, Febrero 1, etc.)
```

Métodos:

`fecha.getDate()`	Retorna el día del mes almacenado en el objeto fecha según la hora local.
`fecha.getDay()`	Retorna el día de la semana almacenado en el objeto fecha según la hora local. El valor retornado es un entero de 0 a 6 correspondiendo a los días de la semana del domingo al sábado.
`fecha.getFullYear()`	Retorna el año almacenado en el objeto fecha según la hora local.
`fecha.getHours()`	Retorna un entero entre 0 y 23 indicando el número de horas desde la medianoche almacenadas en el objeto fecha según la hora local.
`fecha.getMilliseconds()`	Retorna un entero entre 0 y 999 indicando el número de milisegundos almacenados en el objeto fecha según la hora local.
`fecha.getMinutes()`	Retorna un entero entre 0 y 59 indicando el número de minutos almacenados en el objeto fecha según la hora local.
`fecha.getMonth()`	Retorna un entero entre 0 y 11 indicando el mes almacenado en el objeto fecha según la hora local.
`fecha.getSeconds()`	Retorna un entero entre 0 y 59 indicando los segundos almacenados en el objeto fecha según la hora local.
`fecha.getTime()`	Retorna un entero representando el número de milisegundos entre las 00:00:00 del 1 de enero de 1970 y la fecha y hora almacenada en el objeto fecha. El rango es aproximadamente de 285.616 años anteriores y posteriores al 1/1/1970. Los números negativos indican fechas anteriores a 1970.
`fecha.getTimezoneOffset()`	Retorna un entero representando el número de minutos entre la hora en la máquina en la que se ejecuta y la hora UTC.
`fecha.getUTCDate()`	Retorna el día del mes almacenado en el objeto fecha según la hora UTC
`fecha.getUTCDay()`	Retorna el día de la semana almacenado en el objeto fecha según la hora UTC.
`fecha.getUTCFullYear()`	Retorna el año almacenado en el objeto fecha según la hora UTC.
`fecha.getUTCHours()`	Retorna un entero entre 0 y 23 indicando el número de horas desde la medianoche almacenadas en el objeto fecha según hora UTC.
`fecha.getUTCMilliseconds()`	Retorna un entero entre 0 y 999 indicando el número de milisegundos almacenados en el objeto fecha según la hora UTC.
`fecha.getUTCMinutes()`	Retorna un entero entre 0 y 59 indicando el número de minutos

	almacenados en el objeto fecha según la hora UTC.
`fecha.getUTCMonth()`	Retorna un entero entre 0 y 11 indicando el mes almacenado en el objeto fecha según la hora UTC.
`fecha.getUTCSeconds()`	Retorna un entero entre 0 y 59 indicando los segundos almacenados en el objeto fecha según la hora UTC.
`fecha.setDate(día)`	Asigna un nuevo día del mes a la fecha almacenada en fecha según la hora local.
`fecha.setFullYear(año, [ mes, [ día]])`	Asigna un nuevo año a la fecha almacenada en fecha, opcionalmente se puede establecer un mes y un día del mes según la hora local.
`fecha.setHours(horas, [ minutos, [ segundos, [milsegundos]]])`	Asigna una hora a la fecha almacenada en fecha, según la hora local, opcionalmente se pueden establecer los minutos, segundos, y milisegundos.
`fecha.setMilliseconds(milsegundos)`	Asigna milisegundos a la hora almacenada en fecha, según la hora local.
`fecha.setMinutes( minutos, [ segundos, [milsegundos]])`	Asigna minutos a la hora almacenada en fecha, según la hora local, opcionalmente se pueden establecer los segundos, y milisegundos.
`fecha.setMonth( mes, [ valordefecha])`	Asigna el mes a la fecha almacenada en fecha, según la hora local, opcionalmente se pueden establecer un varor de fecha. El valor del mes estará comprendido entre 0 y 11.
`fecha.setSeconds(segundos, [milsegundos])`	Asigna el número de segundos a la hora almacenada en fecha, según la hora local, opcionalmente se pueden establecer los milisegundos.
`fecha.setTime(milsegundos)`	Asigna una fecha y hora al objeto fecha, milisegundos expresa el número de milisegundos desde medianoche del 1/1/1970. Es independiente de la zona horaria.
`fecha.toUTCString()`	Convierte la hora a un string en coordenadas UTC.

Los métodos setUTCDate, setUTCFullYear, setUTCHours, setUTCMilliseconds, setUTCMinutes, setUTCMonth, setUTCSeconds son equivalentes a los vistos anteriormente pero utilizan la hora UTC en vez de la hora local.

Algunos ejemplos de uso:

```
var d = new Date();
document.write(d.getFullYear());
document.write(d.getMonth());
document.write(d.getUTCDay());
var d2 = new Date(5,28,2011,22,58,00);
d2.setMonth(0);
d.setFullYear(2020);
```

OBJETO STRING

El objeto String se utiliza para manipular una cadena almacenada de texto.
Los objetos String se crean con new String().

Sintaxis

```
var txt = new String("cadena");
```

o simplemente:

```
var txt = "cadena";
```

Propiedades:

`string.length`	Devuelve la longitud de la cadena de texto string

Métodos:

Algunos de los métodos del objeto String facilitan la inclusión de etiquetas HTML en las cadenas de texto. No tendremos en cuenta estos métodos, ya que la creación de páginas Web no es el objeto de este documento. En los métodos siguientes string puede ser un objeto String o un literal.

`string.charAt(índice)`	Retorna el carácter en la posición especificada por índice. El índice está en el rango de 0 a string.length
`string.charCodeAt(índice)`	Retorna el código Unicode correspondiente al carácter que se encuentra en la posición especificada por índice.
`string1.concat(string2)`	Retorna un objeto String igual a la concatenación de string1 y string2.
`String.fromCharCode (code1, code2, code2,...)`	Crea un string desde los códigos Unicode que recibe
`string.indexOf(cadena, [índice])`	Encuentra la primera ocurrencia de la cadena en string comenzando la búsqueda en la posición índice. Si no se especifica un índice se comenzará la búsqueda en el primer carácter. Retorna un entero indicando la posición de comienzo de la cadena dentro del objeto string. Si la cadena no se encuentra retorna -1
`string.lastindexOf(cadena, [índice])`	Es idéntico al método indexOf exceptuando que la búsqueda se realiza de derecha a izquierda en vez de izquierda a derecha, retornando la posición de la última ocurrencia de la cadena en string.
`String.search(expReg)`	Busca una subcadena en la cadena y devuelve la posición dónde se encontró.
`string.slice(inicio, [fin])`	Crea un objeto String conteniendo los caracteres de string comprendidos entre las posiciones inicio y fin. Si fin es negativo indica un desplazamiento desde el final de string.
`string.split(separador)`	Crea un array en que cada elemento es un string conteniendo los caracteres que resultan de dividir string en cada punto donde contiene el carácter separador.
`string.substr(inicio, [longitud])`	Retorna un string conteniendo los caracteres de string desde la posición inicio y con la longitud especificada. Si no se especifica la longitud retornará hasta el último carácter.
`string.substring(inicio, [fin])`	Retorna un substring de string conteniendo los caracteres entre las posiciones inicio y fin.
`string.toLowerCase()`	Convierte los caracteres de string a minúsculas.
`string.toUpperCase()`	Convierte los caracteres de string a mayúsculas.
`string.toString()`	Retorna el valor de string.
`string.valueOf()`	Retorna el valor de string.

Ejemplos de uso:

```
var cadena="El parapente es un deporte de riesgo medio";
document.write("La longitud de la cadena es: "+ cadena.length + "<br/>");
document.write(cadena.toLowerCase()+ "<br/>");
document.write(cadena.charAt(3)+ "<br/>");
document.write(cadena.indexOf('pente')+ "<br/>");
document.write(cadena.substring(3,16)+ "<br/>");
```

OBJETO DOCUMENT:

Se refiere a los documentos que se cargan en la ventana del navegador.

Permite manipular las propiedades y el contenido de los principales elementos de las páginas web (accede a todos los elementos html).

Cuenta con una serie de sub-objetos como los vínculos, puntos de anclaje, imágenes o formularios.

Colección del objeto Document:

Colección	Descripción
anchors[]	Es un array que contiene todos los hiperenlaces del documento.
applets[]	Es un array que contiene todos los applets del documento.
forms[]	Es un array que contiene todos los formularios del documento.
images[]	Es un array que contiene todas las imágenes del documento.
links[]	Es un array que contiene todos los enlaces del documento.

Propiedades del objeto Document:

Propiedad	Descripción
alinkColor	Corresponde al color de los vínculos activos de la página.
anchors	Corresponde a los puntos de anclaje (etiquetas <a name>) del documento.
applets	Corresponde a los applets (etiquetas <applet>) Java del documento.
bgColor	Corresponde al color del fondo del documento.
cookie	Corresponde a un fichero guardado en el equipo del cliente del navegador con información sobre el usuario.
domain	Corresponde al nombre del dominio por defecto para el documento.
embeds	Corresponde a los objetos embebidos (etiqueta <embed> en el documento.
fgColor	Corresponde al color del texto del documento.
forms	Corresponde a los formularios (etiqueta <form>) del documento.
images	Corresponde a las imágenes (etiqueta <images>) del documento.
lastModified	Corresponde a la última fecha en la que se modificó el documento.
layers	Corresponde a las capas (etiqueta <layer>) del documento.
linkColor	Corresponde al color de los enlaces aun no visitados.
links	Corresponde a los vínculos (etiqueta <a href>) del documento.
plugins	Corresponde a las referencias y llamadas de los plugins del documento.
readyState	Devuelve el estado de carga del documento actual.
referrer	Corresponde a la dirección del documento usado para ir al documento actual.
title	Corresponde al título (etiqueta <tittle>) del documento.
URL	Corresponde a la dirección del documento.
vlinkColor	Corresponde al color de los enlaces visitados.

Métodos del objeto Document:

Métodos	Descripción
`captureEvents()`	Intercepta un evento para que pueda ser manipulado por el documento.
`close()`	Cierra el documento activo, abierto previamente con document.open().
`getElementById()`	Para acceder a un elemento identificado por el **id** escrito en paréntesis.
`getElementsByName()`	Para acceder a los elementos identificados por el atributo **name** escrito entre paréntesis.
`getsElementsByTagName()`	Para acceder a los elementos identificados por el tag o la etiqueta escrita entre paréntesis.
`getSelection()`	Devuelve el texto seleccionado en el documento.
`handleEvent()`	Activa el manejador del evento especificado.
`home()`	Carga la página de inicio.
`open()`	Activa el documento.
`releaseEvents()`	Libera los eventos que han sido interceptados.
`routeEvents()`	Intercepta un evento y lo pasa a lo largo de la jerarquía del objeto que lo lanzó.
`write()`	Escribe datos en el documento.
`writeln()`	Escribe datos además de un salto de línea en el documento.

OBJETO WINDOW:

Permite gestionar las ventanas del navegador:

El método que genera una nueva ventana es window.open(). Este método contiene hasta tres parámetros: la URL del documento a abrir, el nombre de esa ventana y su apariencia física (tamaño, color, etc.)

```
var subVentana=window.open("nueva.html","nueva","height=800,width=600");
```

Para cerrar la nueva ventana desde nuestro script utilizamos el método close().

```
subVentana.close();
```

Propiedades del objeto Window:

Propiedad	Descripción
`closed`	Corresponde al valor booleano que indica si la ventana está cerrada o no.
`defaultStatus`	Corresponde a la cadena de texto de la barra de estado del navegador.
`document`	Corresponde al documento actual de la ventana.
`frames`	Corresponde al conjunto de marcos de la ventana.
`history`	Corresponde al conjunto de elementos que representan las URL visitadas.
`innerHeight`	Corresponde a la altura utilizable de la ventana.
`innerWidth`	Corresponde al ancho utilizable de la ventana.
`length`	Corresponde al número de frames de la ventana.

location	Corresponde a la URL de la barra de direcciones.
locationbar	Corresponde a la barra de direcciones del navegador.
menubar	Corresponde a la barra del menú del navegador.
name	Corresponde al nombre de la ventana.
opener	Corresponde a la referencia del objeto window que haya abierto una ventana nueva.
outerHeight	Corresponde a la altura exterior de la página.
outerWidth	Corresponde al ancho exterior de la página.
pageXoffset	Corresponde a la posición horizontal de la ventana.
pageYoffset	Corresponde a la posición vertical de la ventana.
parent	Corresponde a la referencia del objeto window que contiene los marcos de una página con marcos
personalbar	Corresponde a la barra de herramientas personal.
scrollbars	Corresponde a las barras de desplazamiento horizontal y vertical.
self	Corresponde a la ventana actual.
status	Corresponde a la cadena con el mensaje que contiene la barra de estado.
toolbar	Corresponde a la barra de herramientas del navegador.
top	Corresponde a la ventana de nivel superior.

Métodos del objeto Window:

Métodos	Descripción
alert(texto)	Presenta una ventana de alerta donde se puede leer el texto que recibe por parámetro.
back()	Ir una página atrás en el historial de páginas visitadas. Funciona como el botón de volver de la barra de herramientas
blur()	Desactiva la ventana.
close()	Cierra la ventana.
confirm(texto)	Muestra una ventana de confirmación y permite aceptar o rechazar.
find()	Muestra una ventanita de búsqueda.
focus()	Coloca el foco de la aplicación en la ventana.
forward()	Ir una página adelante en el historial de páginas visitadas. Como si pulsásemos el botón de adelante del navegador.
home()	Ir a la página de inicio que haya configurada en el explorador.
moveBy(pixelsX, pixelsY)	Mueve la ventana del navegador los pixels que se indican por parámetro hacia la derecha y abajo.
moveTo(pixelsX, pixelsY)	Mueve la ventana del navegador a la posición indicada en las coordenadas que recibe por parámetro.
open()	Abre una ventana secundaria del navegador.
print()	Abre una ventana secundaria del navegador.
prompt()	Muestra una caja de diálogo para pedir un dato. Devuelve el

	dato que se ha escrito.
`resizeBy(pixelsAncho,pixelsAlto)`	Redimensiona el tamaño de la ventana, añadiendo a su tamaño actual los valores indicados en los parámetros. El primero para la altura y el segundo para la anchura. Admite valores negativos si se desea reducir la ventana.
`resizeTo(pixelsAncho,pixelsAlto)`	Redimensiona la ventana del navegador para que ocupe el espacio en pixels que se indica por parámetro.
`scroll(pixelsX, pixelsY)`	Hace un scroll de la ventana hacia la coordenada indicada por parámetro. Este método está desaconsejado, pues ahora se debería utilizar scrollTo()
`scrollBy(pixelsX,pixelsY)`	Hace un scroll del contenido de la ventana relativo a la posición actual.
`scrollTo(pixelsX,pixelsY)`	Hace un scroll de la ventana a la posición indicada por el parámetro. Este método se tiene que utilizar en lugar de scroll.
`setInterval()`	Define un script para que sea ejecutado indefinidamente en cada intervalo de tiempo.
`setTimeout(sentencia,milisegundos)`	Define un script para que sea ejecutado una vez después de un tiempo de espera determinado
`stop()`	Detiene una página.

TIMERS

Los timers son funciones predefinidas del objeto window (por tanto se pueden invocar usando window.nombreDeLaFuncion(…) o simplemente usando nombreDeLaFuncion(…).

SINTAXIS TIMER	UTILIDAD	EJEMPLOS

`varreferenciaTimer = setTimeout(nombreFuncion, tiempoMiliseg);`
 	
*** Para función sin parámetros**
	La función nombreFuncion se ejecutará con un retraso de tiempoMiliseg respecto a lo que sería su ejecución natural.	varejecT = setTimeout(mostrarAlerta, 5000);
		
		//mostrarAlerta se ejecuta con 5 segundos de retraso

`varreferenciaTimer = setTimeout(function() {nombreFuncion (par1, par2, …, parN)},tiempoMiliseg);`
 	
*** Para función con parámetros**
	La función nombreFuncion se ejecutará con un retraso de tiempoMiliseg respecto a lo que sería su ejecución natural.	var control = setTimeout(function() {ejemploAccion(nodosTituloCurso, nodosCambiados, contador, tocaCambiar)}, 1500);
		
		//La función se ejecuta con 1.5 s de retraso

`clearTimeOut(referenciaTimer)`	Detiene la ejecución programada por referenciaFuncionADetener mediante setTimeOut (si se ejecuta antes del tiempo programado)	clearTimeOut(ejecT)

SINTAXIS TIMER	UTILIDAD	EJEMPLOS

`var referenciaTimer = setInterval(nombreFuncion(par1, par2, … parN), tiempoMiliseg);`
 	
*** Para función sin parámetros**
	Ejecuta periódicamente la función nombreFuncion con un intervalo entre ejecuciones de tiempoMiliseg.	setInterval(mostrarAlerta, 5000);
		
		//mostrarAlerta se ejecuta periódicamente cada 5 segundos

`var referenciaTimer = setInterval(function() {nombreFuncion(par1, par2, … parN)}, tiempoMiliseg);`
 	
*** Para función con parámetros**
	Ejecuta periódicamente la función nombreFuncion con un intervalo entre ejecuciones de tiempoMiliseg.	var t = setInterval(function(){reloj('Chile') },2000);
		
		//reloj se ejecuta periódicamente cada 2 s

`clear`**`Interval`**`(referenciaTimer)`	Detiene la ejecución programada por referenciaFuncionADetener mediante setInterval	clearInterval(ejecT)
`var referenciaTimer = requestAnimationFrame(nombreFuncion)` *** Para función sin parámetros**	Crea un bucle de repintado delegando el control del mismo en el navegador.	globalID = requestAnimationFrame(animacionRepetimos);
`var referenciaTimer = requestAnimationFrame(function(){nombreFuncion(par1, par2, …, parN)});` *** Para función con parámetros**	Crea un bucle de repintado delegando el control del mismo en el navegador.	globalID = requestAnimationFrame(function(){animacionRepetimos('estiloDivertido') });
`cancelAnimationFrame(referenciaTimer)`	Detiene la ejecución programada por referenciaFuncionADetener mediante RequestAnimationFrame.	

EVENTOS: Algunos de los eventos más importantes definidos por JavaScript:

Evento	Descripción	Elementos para los que está definido
onblur	Deseleccionar el elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onchange	Deseleccionar un elemento que se ha modificado	<input>, <select>, <textarea>
onclick	Pinchar y soltar el ratón	Todos los elementos
ondblclick	Pinchar dos veces seguidas con el ratón	Todos los elementos
onfocus	Seleccionar un elemento	<button>, <input>, <label>, <select>, <textarea>, <body>
onkeydown	Pulsar una tecla (sin soltar)	Elementos de formulario y <body>
onkeypress	Pulsar una tecla	Elementos de formulario y <body>
onkeyup	Soltar una tecla pulsada	Elementos de formulario y <body>
onload	La página se ha cargado completamente	<body>
onmousedown	Pulsar (sin soltar) un botón del ratón	Todos los elementos
onmousemove	Mover el ratón	Todos los elementos
onmouseout	El ratón *"sale"* del elemento (pasa por encima de otro elemento)	Todos los elementos
onmouseover	El ratón *"entra"* en el elemento (pasa por encima del elemento)	Todos los elementos
onmouseup	Soltar el botón que estaba pulsado en el ratón	Todos los elementos
onreset	Inicializar el formulario (borrar todos sus datos)	<form>
onresize	Se ha modificado el tamaño de la ventana del navegador	<body>
onselect	Seleccionar un texto	<input>, <textarea>
onsubmit	Enviar el formulario	<form>
onunload	Se abandona la página (por ejemplo al cerrar el navegador)	<body>

Atributos para <frame>

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.

Propiedades de ventana.

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.

Métodos y propiedades del objeto XMLHttpRequest

Propiedad	Descripción
`readyState`	Valor numérico (entero) que almacena el estado de la petición
`responseText`	El contenido de la respuesta del servidor en forma de cadena de texto
`responseXML`	El contenido de la respuesta del servidor en formato XML. El objeto devuelto se puede procesar como un objeto DOM
`status`	El código de estado HTTP devuelto por el servidor (**200** para una respuesta correcta, **404** para "No encontrado", **500** para un error de servidor, etc.)
`statusText`	El código de estado HTTP devuelto por el servidor en forma de cadena de texto: "OK", "Not Found", "Internal Server Error", etc.

Los valores definidos para la propiedad `readyState` son los siguientes:

Valor	Descripción
`0`	No inicializado (objeto creado, pero no se ha invocado el método **open**)
`1`	Cargando (objeto creado, pero no se ha invocado el método **send**)
2	Cargado (se ha invocado el método **send**, pero el servidor aún no ha respondido)
3	Interactivo (se han recibido algunos datos, aunque no se puede emplear la propiedad `responseText`)
4	Completo (se han recibido todos los datos de la respuesta del servidor)

Los métodos disponibles para el objeto `XMLHttpRequest` son los siguientes:

Método	Descripción
`abort()`	Detiene la petición actual
`getAllResponseHeaders()`	Devuelve una cadena de texto con todas las cabeceras de la respuesta del servidor
`getResponseHeader("cabecera")`	Devuelve una cadena de texto con el contenido de la cabecera solicitada
`onreadystatechange`	Responsable de manejar los eventos que se producen. Se invoca cada vez que se produce un cambio en el estado de la petición HTTP. Normalmente es una referencia a una función JavaScript
`open("metodo", "url")`	Establece los parámetros de la petición que se realiza al servidor. Los parámetros necesarios son el método HTTP empleado y la URL destino (puede indicarse de forma absoluta o relativa)
`send(contenido)`	Realiza la petición HTTP al servidor
`setRequestHeader("cabecera", "valor")`	Permite establecer cabeceras personalizadas en la petición HTTP. Se debe invocar el método `open()` antes que `setRequestHeader()`