**2.1. Sintaxis del lenguaje. Operadores y palabras reservadas**

**La sintaxis de JavaScript es muy parecida a Java y C++. No obstante, es importante conocer los eventos, el control de los elementos HTML, etc.**

**2.1.1. Las 10 reglas básicas de la sintaxis del lenguaje**

**Regla 1. Las instrucciones en JavaScript terminan en un punto y coma. Ejemplo:**

****

**Regla 2. Decimales en JavaScript. Los números en JavaScript que tengan decimales utilizarán el punto. Ejemplos:**

****

**Regla 3. Los literales se pueden escribir entre comillas dobles o simples. Ejemplo:**

****

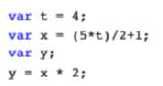
**Regla 4. Para declarar una variable, se utilizará la palabra reservada var.**

**Regla 5. El operador de asignación es el símbolo igual (=).**

**Regla 6. Se pueden utilizar los siguientes operadores aritméticos: (+-\*/). Ejemplo:**

****

**Regla 7. En las expresiones, también se pueden utilizar variables. Ejemplo:**

****

**Regla 8. Comentarios en JavaScript. Existen dos opciones:**

**a) // Para comentar el resto de la línea a partir de estas dos barras invertidas.**

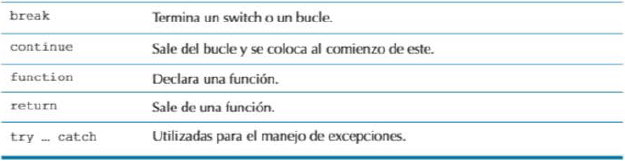
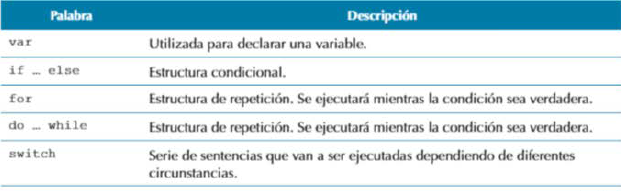
**b) /\* y\*/. todo lo contenido entre ambas etiquetas quedará comentado.**

**Regla 9. Los identificadores en JavaScript comienzan por una letra o \_ o $.**

**Regla 10. JavaScript es (case-sensitive). Ejemplo:**

****

**2.1.2. Las palabras reservadas de JavaScript**

****

**2.1.3. Uso de variables en JavaScript**

**Una variable es una posición de memoria donde se pueden alojar datos. El nombre de la variable permitirá a JavaScript ubicar y localizar ese espacio cuando interprete los scripts. Uso:**

**Constantes en JavaScript**

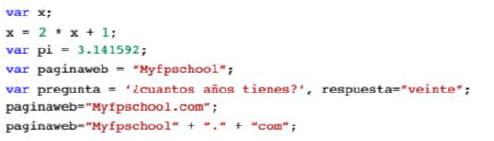
**En JavaScript, se puede utilizar la palabra reservada const. En vez de utilizar:**

****

**se aconseja emplear:**

****

**En JavaScript, se pueden ejecutar las siguientes órdenes:**

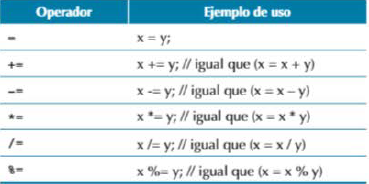
****

**2.1.4. Operadores en JavaScript**

**A) Operadores aritméticos**

****

**B) Operadores de asignación**

****

**C) Operadores de manejo de strings**

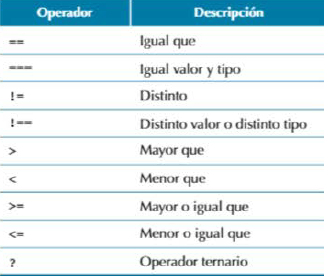
**En JavaScript, se utilizan los operadores + y += para concatenar strings. Ejemplo:**

****

**O lo que sería igual:**

****

**D) Operadores lógicos y de comparación**

****

**E) Operadores de tipo**

**Permiten conocer si un objeto es una instancia de un tipo concreto o conocer el tipo de una variable:**

**typeof. Devuelve el tipo de una variable.**

**instanceof. Devuelve true si un objeto es una instancia de un tipo de objeto.**

**2.2. Tipos de datos. Asignaciones y expresiones**

**Solamente tiene cinco tipos de datos:**

**1. String.**

**2. Number.**

**3. Boolean.**

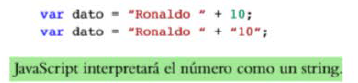
**4. Array.**

**5. Object.**

**Ejemplo:**

****

**"Dimas" o "lengua" son literales y, por eso, aparecen entrecomillados. Para JavaScript, las siguientes dos líneas de código son iguales:**

****

**2.2.1. El valor null**

**Null es nada. Cuando se asigna null a una variable, es porque es un objeto.**

**ejemplo:**

****

**2.2.2. Conversiones entre tipos de datos en JavaScript**

**JavaScript es un lenguaje interpretado, la conversión entre un tipo de dato y otro es transparente al programador. Asignará el tipo de datos más acertado a la variable que el programador esté utilizando.**

**Pero existen funciones de conversión como String() y Number() para convertir a cadenas de caracteres y números.**

**También existen las funciones:**

**parseInt(). Que convierte una cadena a un número entero.**

**parseFloat(). Que convierte una cadena a un número decimal.**

**ejemplo:**

****

**Es posible también convertir variables de tipo fecha a string con la función toDateString(). Ejemplo:**

****

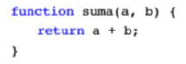
**Para convertir la hora en formato UTC, se puede utilizar la función toUTCString(). Ejemplo:**

****

**2.3. Introducción a las funciones**

**Reutilizar el código es algo básico en cualquier lenguaje de programación. La variación del resultado de las funciones dependerá de sus argumentos.**

**ejemplo:**

****

**Se tiene la palabra reservada function seguida de su nombre. Esta función aceptará dos parámetros a y b y devolverá (return) la suma de ambos.**

**ejemplo:**

****

**2.4. Introducción a los objetos en JavaScript**

**cómo se definiría un objeto en JavaScript:**

****

**un objeto, al igual que en la vida real, puede tener muchos atributos, y cada atributo se verá reflejado en una variable dentro del objeto. Un objeto se diferenciará de otros por el valor de sus atributos.**

**definiciones de objetos en una sola línea:**

****

**Para acceder a los atributos de un objeto:**

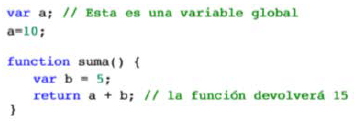
****

**O bien:**

****

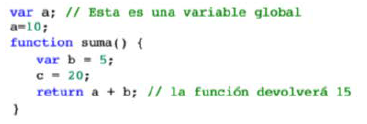
**2.5. Variables y ámbitos de utilización**

**diferencia que existe entre variables globales y locales en JavaScript. Ejemplo:**

****

**La diferencia entre a y b es que a puede utilizarse en otras funciones, y b solamente existirá dentro de la función suma.**

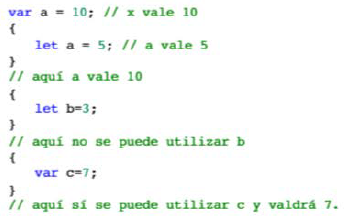
**¿Qué es lo que ocurre cuando no se declara una variable como la variable c en el siguiente código?**

****

**Que será global a todo el JavaScript de esa página.**

**2.5.1. Las diferencias entre var y let**

**La diferencia entre ambas palabras reservadas es que, cuando se declara una variable con var, la variable puede utilizarse fuera del bloque donde fue declarada. Ejemplo:**

****

**2.6. Eventos**

**son cualquier suceso que le pueda ocurrir a un elemento HTML.**

**Algunos eventos que pueden ocurrir en una página web son:**

**Pulsar un botón.**

**Modificar un campo de texto.**

**Pulsar una tecla.**

**La página ha terminado de cargarse.**

**Hacer clic sobre un elemento HTML.**

**El formato de programar un evento en JavaScript:**

****

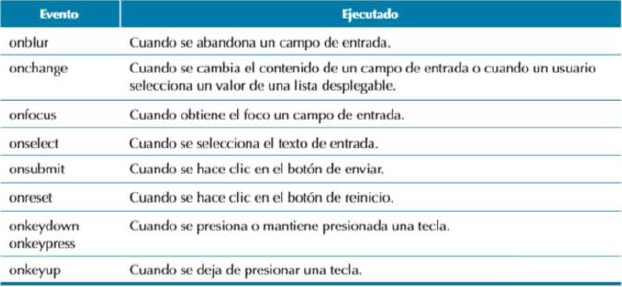
**También es posible cambiar las comillas simples por comillas dobles.**

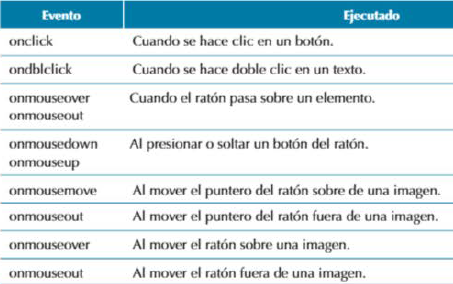
**ejemplo de evento en javascript programado para un botón:**

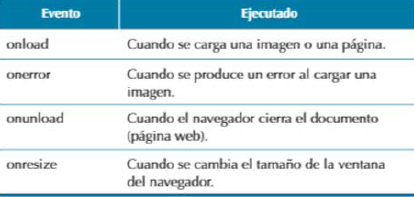
****

**2.6.1. Tipos de eventos en JavaScript**

**En un navegador, hay muchos tipos que pueden ser controlados por JavaScript:**

****

****

****

**2.6.2. Los listener**

**Otra forma de añadir los eventos a los elementos u objetos HTML del DOM (listeners):**

****

**en el código anterior se utiliza el método addEventListener para vincular un evento a una función JavaScript.**

**Los parámetros de la función addEventListener():**

**1. El evento (sin el "on").**

**2. La función por ejecutar.**

**3. useCapture. Se utilizará si el usuario quiere iniciar la captura para que todos los eventos de ese tipo se lancen en el mismo listener. Generalmente, se utiliza false.**

**2.7. Objeto string o cadena de caracteres**

**ejemplos de funcionamiento de strings:**

****

**podían utilizar comillas dobles y simples**

**Para utilizar las comillas dentro de un string, se necesitará utilizar secuencias de escape. Un ejemplo:**

****

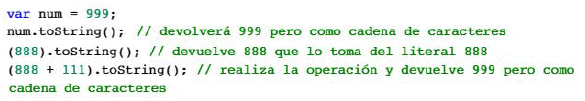
**El contenido sería: la mejor "web" de tecnología.**

**2.8. Números**

**2.8.1. Convertir un número en un string**

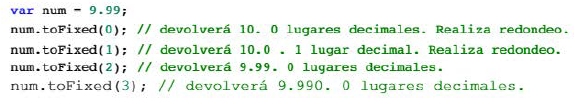
**En JavaScript, existe el método toString(), que convierte un número en una cadena.**

**ejemplo:**

****

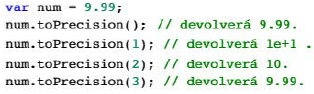
**2.8.2. Ajuste de decimales. Número de decimales de un número**

**el método tofixed() servirá para ajustar el número de decimales de un número.**

****

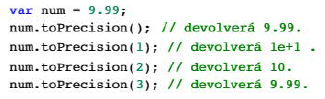
**2.8.3. Precisión de un número**

**el método toPrecision() servirá para ajustar la precisión de un número:**

****

**2.8.4. Convertir cualquier variable en un número**

**La función Number() permite convertir cualquier variable en un número. Ejemplo:**

****

**2.9. Fechas**

**Las fechas se pueden visualizar como un número o un string. Ejemplo:**

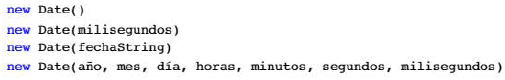
****

**El resultado de visualizar la fecha actual con el script anterior sería algo similar a:**

****

**2.9.1. Creación de un objeto de tipo fecha**

**Existen cuatro formas diferentes de inicializar un objeto de tipo fecha:**

****

**Véanse algunas formas de utilizar estos constructores:**

****

**2.9.2. Formatos de tipo fecha**

**Existen distintos formatos de tipo fecha:**

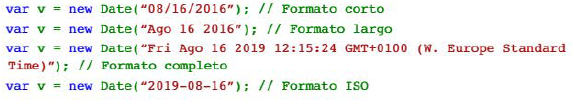
**Formato corto.**

**Formato largo.**

**Formato completo.**

**Formato ISO.**

**ejemplo:**

****

**2.9.3. Métodos del tipo fecha**

**En JavaScript, los objetos tipo fecha tienen los siguientes métodos:**

**getDate(). Obtiene el día del mes (1-31).**

**getDay(). Obtiene el día de la semana en formato numérico (0-6).**

**getFull Year(). Obtiene el año (2016, por ejemplo).**

**getHours().**

**getMilliseconds().**

**getMinutes().**

**getMonth().**

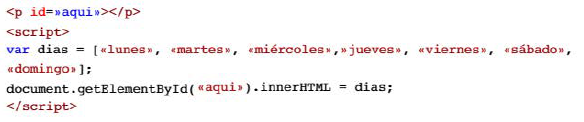
**get Seconds().**

**getTime(). Obtiene la hora en milisegundos desde el 1 de enero de 1970.**

**2.10. Arrays**

**Los arrays son una serie de elementos u objetos contiguos, a los cuales pueden accederse utilizando un subíndice.**

**ejemplo de utilización de arrays dentro de una página web:**

****

**También se podría haber creado el array asi:**

****

**2.10.1. Operaciones con arrays**

**Existen múltiples operaciones en JavaScript para manipular arrays:**

**Conocer su longitud:**

****

**Ordenarlo en alfabéticamente:**

****

**Añadir un nuevo elemento:**

****

**Extraer el último elemento:**

****

**El último elemento se eliminará del array. Extraer significa "retirar".**

**Eliminar el primer elemento:**

****

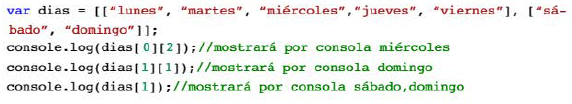
**Eliminar un elemento concreto:**

****

**2.10.2. Arrays multidimensionales**

**Consisten en tener en una posición de un array otro array de elementos. Se pueden tener arrays bidimensionales, tridimensionales, etc.**

**un ejemplo de creación y acceso a un array bidimensional. Como se puede ver, se han separado los días laborables del fin de semana en los dos elementos del array principal:**

****

**2.11. Sentencias condicionales**

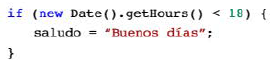
**If. Se ejecutará el bloque si la condición es verdadera.**

**Else. Se ejecutará el bloque si la condición es falsa.**

**Else if. Es combinación de las anteriores y se utilizará cuando la primera condición sea falsa.**

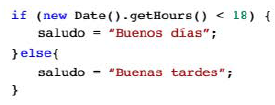
**Switch. Se utilizará para especificar una serie de códigos que se ejecutarán de forma alternativa.**

**La sentencia IF:**

****

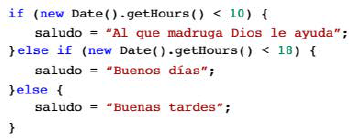
**El código anterior hace que la variable saludo valga "Buenos días" si es antes de las 18:00.**

**La sentencia ELSE:**

****

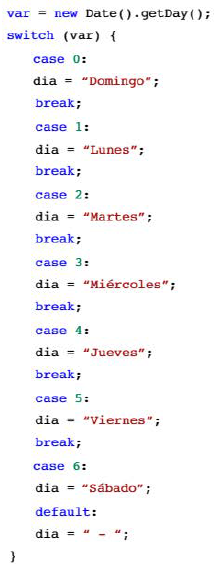
**hace que la variable saludo valga "Buenos días" si es antes de las 18:00 y "Buenas tardes" en caso contrario.**

**La sentencia ELSE IF:**

****

**hace que la variable saludo valga "Al que madruga Dios le ayuda" si la hora es antes de las 10 de la mañana, "Buenos días" si es antes de las 18:00 y "Buenas tardes" en caso contrario.**

**La sentencia SWITCH:**

****

**dependiendo del valor de la variable var, la variable dia tendrá un valor u otro en función del día de la semana en el que se esté.**

**2.12. Bucles**

**A veces se necesita ejecutar un trozo de código repetidas veces. Para ello, se usan los bucles y los hay de distintos tipos:**

**a) Bucle for. Se utiliza esta sentencia cuando se quiere ejecutar un bloque una serie de veces determinada**

**b) Bucle while. Se emplea esta sentencia cuando se desea ejecutar un bloque de código una serie de veces sin conocer cuántas. El bloque se ejecutará mientras se cumpla una condición.**

**bucle FOR:**

****

**se inicializa la variable i a 0 (i=0). Dicho bucle se ejecutará mientras que la variable i sea menor o igual que 10 (i<=10) y la variable i a su vez se irá incrementando en una unidad (i++) en cada pasada del bucle.**

**Existe una variante del bucle for que es el bucle for in:**

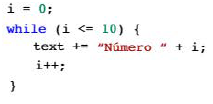
****

**Es posible crear un bucle decreciente:**

****

**el valor de i configurado a 10, en un principio, irá decreciendo y el bucle se ejecutará siempre y cuando el valor de la variable i sea mayor o igual que 0.**

**bucle WHILE:**

****

**Como puede observarse, es el anterior ejemplo del bucle for, pero ahora reconvertido a while.**

**2.12.1. Recorrido de un array**

**Existen dos formas de recorrer un array:**

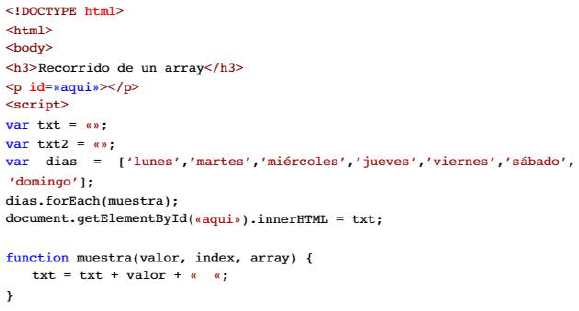
**1. Mediante el recorrido clásico. Se utilizará un bucle para recorrer el array. Ejemplo:**

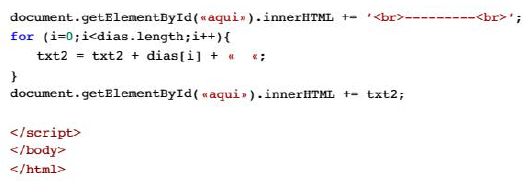
****

**2. Mediante el método forEach(). Ejemplo:**

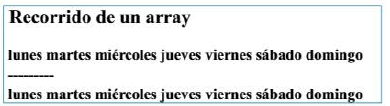
****

**Donde muestra es una función que será invocada por cada uno de los elementos del array dias. Ejemplo:**

****

****

**El aspecto del navegador del ejemplo anterior será el que se muestra**

****

**2.13. practica guiada**

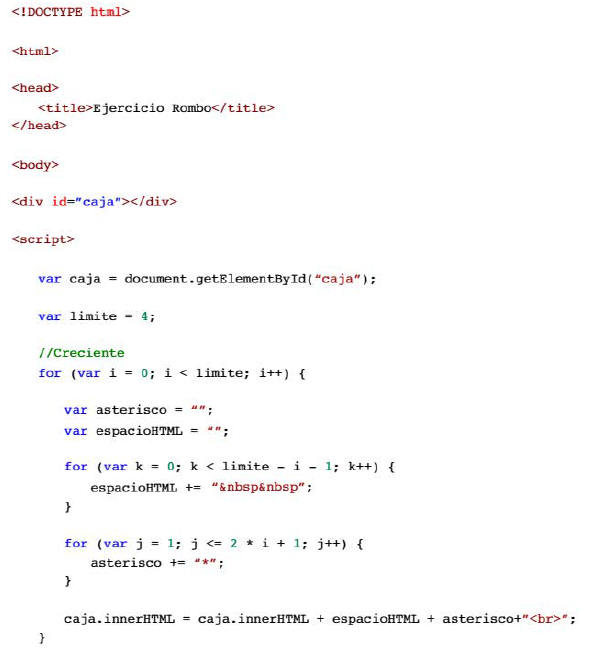
**crear un rombo respetando los espacios y la forma en lo posible. Para el espacio, hay que utilizar el símbolo HTML &nbsp y, para el salto de línea, la etiqueta <br>.**

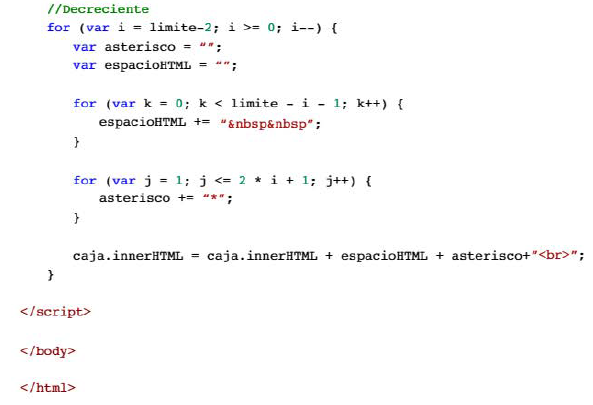
****

**Se ha dividido el rombo en dos partes: una parte superior en la que va creciendo el número de asteriscos y otra parte inferior en la que el número de asteriscos es decreciente.**

**En las variables asterisco y espacioHTML, se van a almacenar respectivamente los asteriscos y los espacios HTML que han de mostrarse respectivamente.**

**En una variable de nombre límite, se registra el número de filas (4) que va a tener la parte creciente (la decreciente tendrá una menos).**

****

****

**Parte creciente. En esta parte, se ejecuta un bucle creciente y se observa que el código más complejo son los bucles que almacenan los espacios HTML y los asteriscos en las variables espacioHTML y asterisco.**

**En el caso de los espacios, estos van a ir decreciendo, dado que se establece en el bucle que la variable índice k sea menor que limite - i - 1 (siendo i el índice del bucle principal).**

****

**En el caso de los asteriscos, es todo lo contrario. Al decrecer los espacios, crecen los asteriscos y, por tanto, el índice j del bucle será menor o igual a 2\* i + 1.**

****

**Parte decreciente. Es igual a la parte creciente, pero se observa que, en este caso, el bucle principal tiene que ser decreciente. El bucle comienza en límite-2, dado que se va a ejecutar solamente tres veces con valores del índice i igual a 2, 1 y 0.**

****