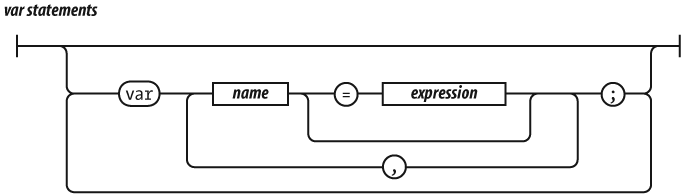
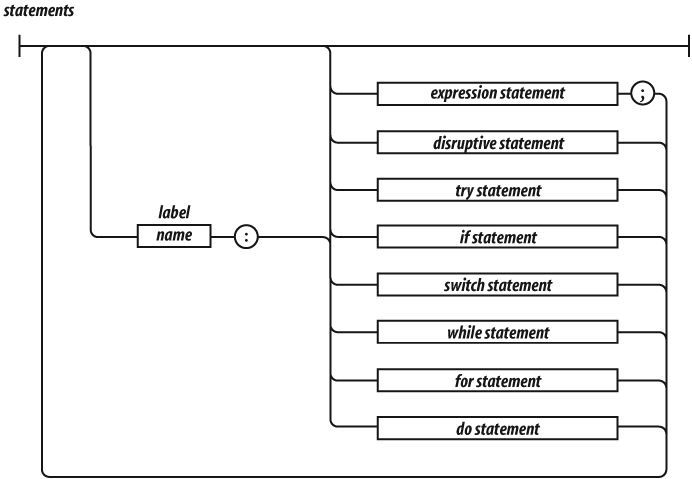
Statements

# Sentencias

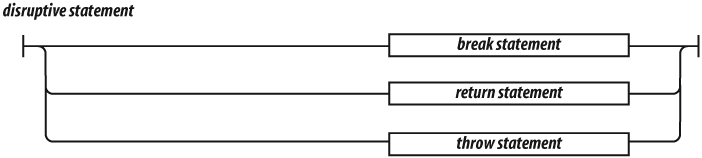


Una unidad de compilación contiene un conjunto de sentencias ejecutables. En los navegadores web, cada etiqueta <script> proporciona una unidad de compilación que se compila y ejecuta inmediatamente. Al carecer de enlazador, JavaScript las lanza todas juntas en un espacio de nombres global común. En [el Apéndice A](apa.html) encontrarás más información sobre las variables globales.

Cuando se utiliza dentro de una función, la sentencia vardefine las variables privadas de la función.



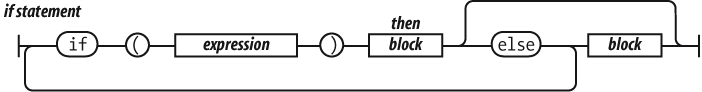
Las sentencias switch, while, for, y do pueden tener un prefijo opcional *label* que interactúa con la sentencia break.



Las sentencias tienden a ejecutarse en orden de arriba a abajo. La secuencia de ejecución puede ser alterada por las sentencias condicionales (if y switch), por las sentencias de bucle (while, for, y do), por las sentencias disruptivas (break, return, y throw), y por la invocación de funciones.

image with no caption

Un bloque es un conjunto de sentencias envueltas entre llaves. A diferencia de muchos otros lenguajes, los bloques en JavaScript no crean un nuevo ámbito, por lo que las variables deben definirse en la parte superior de la función, no en los bloques.

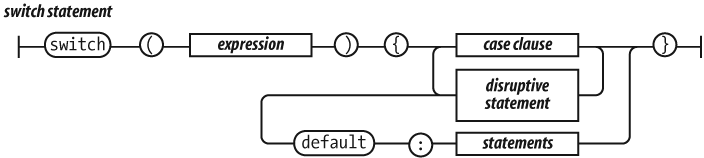


La sentencia if cambia el flujo del programa en función del valor de la expresión. El bloque *then* se ejecuta si la expresión es *truthy*; en caso contrario, se toma la rama opcional else.

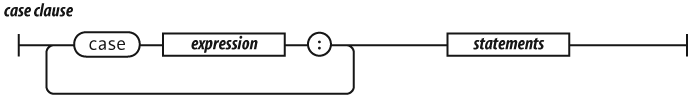
Estos son los valores de *falsy*:

* false
* null
* undefined
* La cadena vacía ''
* El número 0
* El número NaN

Todos los demás valores son verdaderos, incluidos true, la cadena 'false', y todos los objetos.



La sentencia switch realiza una bifurcación multidireccional. Compara la igualdad de la expresión con todos los casos especificados. La expresión puede producir un número o una cadena. Cuando se encuentra una coincidencia exacta, se ejecutan las sentencias de la cláusula del caso coincidente. Si no hay coincidencia, se ejecutan las sentencias opcionales de default.

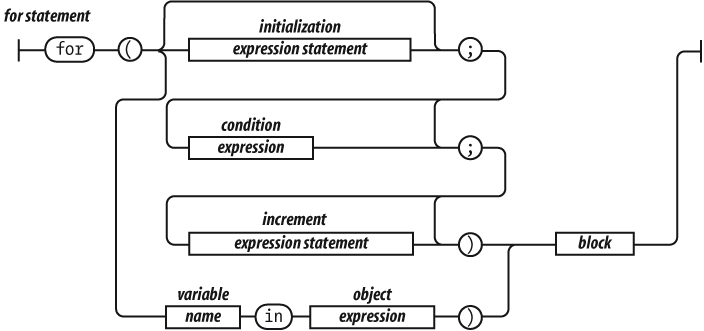


Una cláusula case contiene una o varias expresiones case. No es necesario que las expresiones case sean constantes. La sentencia que sigue a una cláusula debe ser una sentencia disruptiva para evitar que se pase a la siguiente case. La sentencia break puede utilizarse para salir de un conmutador.

image with no caption

La sentencia while realiza un bucle simple. Si la expresión es falsa, el bucle se romperá. Mientras la expresión sea verdadera, se ejecutará el bloque.

La sentencia for es una sentencia de bucle más complicada. Tiene dos formas.



La forma convencional está controlada por tres cláusulas opcionales: la cláusula *initialization*la *condition*y la *increment*. Primero se realiza la inicialización, que suele inicializar la variable del bucle. A continuación, se evalúa la cláusula *condition* se evalúa. Normalmente, esto comprueba la variable del bucle con un criterio de finalización. Si se omite el parámetro *condition* se omite, se asume un *condition* de true. Si la variable *condition* es falso, el bucle se rompe. En caso contrario, se ejecuta el bloque, luego se ejecuta el *increment* y el bucle se repite con el bloque *condition*.

La otra forma (llamada for in) enumera los nombres de las propiedades (o claves) de un objeto. En cada iteración, se asigna otra cadena de nombre de propiedad del bloque *object* se asigna a la cadena *variable*.

Suele ser necesario comprobar *object*.hasOwnProperty(*variable*) para determinar si el nombre de la propiedad es realmente un miembro del objeto o se encontró en su lugar en la cadena de prototipos.

for (myvar in obj) {  
 if (obj.hasOwnProperty(myvar)) {  
 ...  
 }  
}

image with no caption

La sentencia do es como la sentencia while, salvo que la expresión se comprueba después de que se ejecute el bloque, en lugar de antes. Esto significa que el bloque siempre se ejecutará al menos una vez.

image with no caption

La sentencia try ejecuta un bloque y captura las excepciones lanzadas por el bloque. La cláusula catch define un nuevo *variable* que recibirá el objeto de excepción.

image with no caption

La sentencia throw lanza una excepción. Si la sentencia throw está en un bloque try, el control pasa a la cláusula catch. En caso contrario, se abandona la invocación a la función y el control pasa a la cláusula catch de la try de la función invocadora.

La expresión suele ser un literal de objeto que contiene una propiedad name y una propiedad message. El receptor de la excepción puede utilizar esa información para determinar qué hacer.

image with no caption

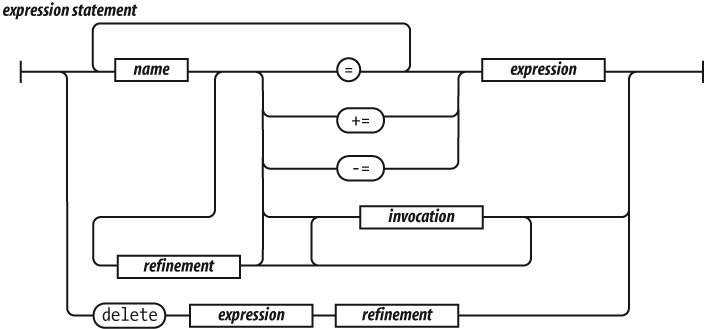
La expresión return provoca el retorno anticipado de una función. También puede especificar el valor a devolver. Si no se especifica una expresión de retorno, el valor devuelto será undefined.

JavaScript no permite un final de línea entre el retorno y la expresión.

image with no caption

La expresión break provoca la salida de una expresión de bucle o de una expresión switch. Puede tener opcionalmente un *label* que provocará la salida de la sentencia etiquetada.

JavaScript no permite un final de línea entre el break y la etiqueta.



Una sentencia expression puede asignar valores a una o varias variables o miembros, invocar un método o eliminar una propiedad de un objeto. El operador = se utiliza para asignar. No lo confundas con el operador de igualdad ===. El operador += puede sumar o concatenar.