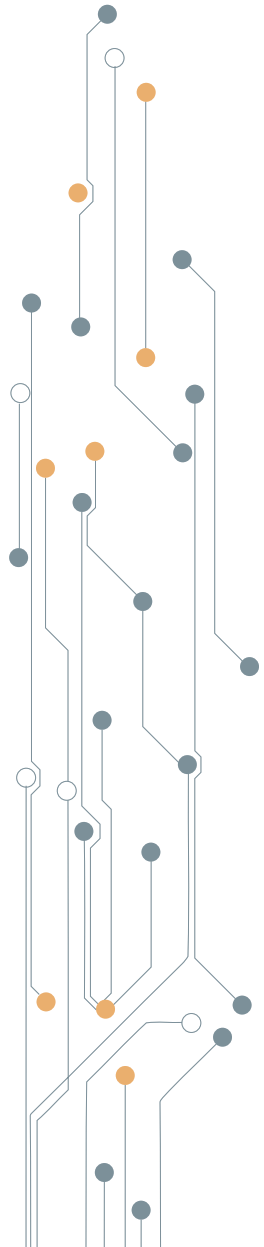
Objetos en javascript y la especificación JSON

Telefónica
EDUCACIÓN DIGITAL

Índice



1. Introducción

JSON (acrónimo de JavaScript Object Notation), es un formato para el intercambio de datos por la red, donde usualmente se utilizaba XML. Es un conjunto de datos, comprendidos entre los que puede medir JavaScript que son objetos, Arrays, cadenas, booleanos y números en Javascript.

Llegó sobre 2001 gracias al apoyo incondicional de Douglas Crockford. Yahoo! ayudó a su difusión gracias a la adición de este formato en algunos de sus servicios web más innovadores. Google comienza a realizar sus feeds en JSON para su protocolo web GData a finales del 2006.

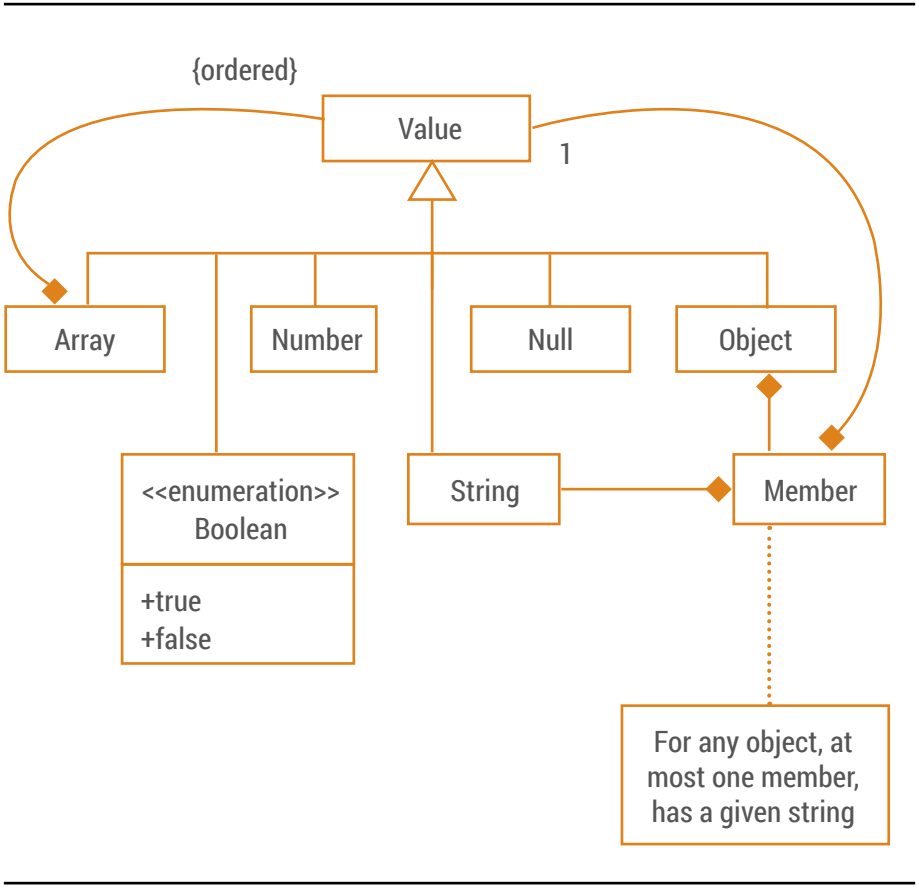
Es considerado como un lenguaje independiente de formato de los datos cuya especificación es descrita en RFC4627.

```
{
“id”  :  “0001”,
“type”  :  “donut”,
“name”  :  “Cake”,
“image”  :  “  {
     “url”  :  “images/0001.jpg”,
“width”  :  200,
“height”  :  200
},
“thumbnail”  : {
      “url”  :  “images/thumbnails/0001.jpg”,
      “width”  : 32,
      “height”  : 32
},
“dateEntry”  :  “2010-12-05”
}
```

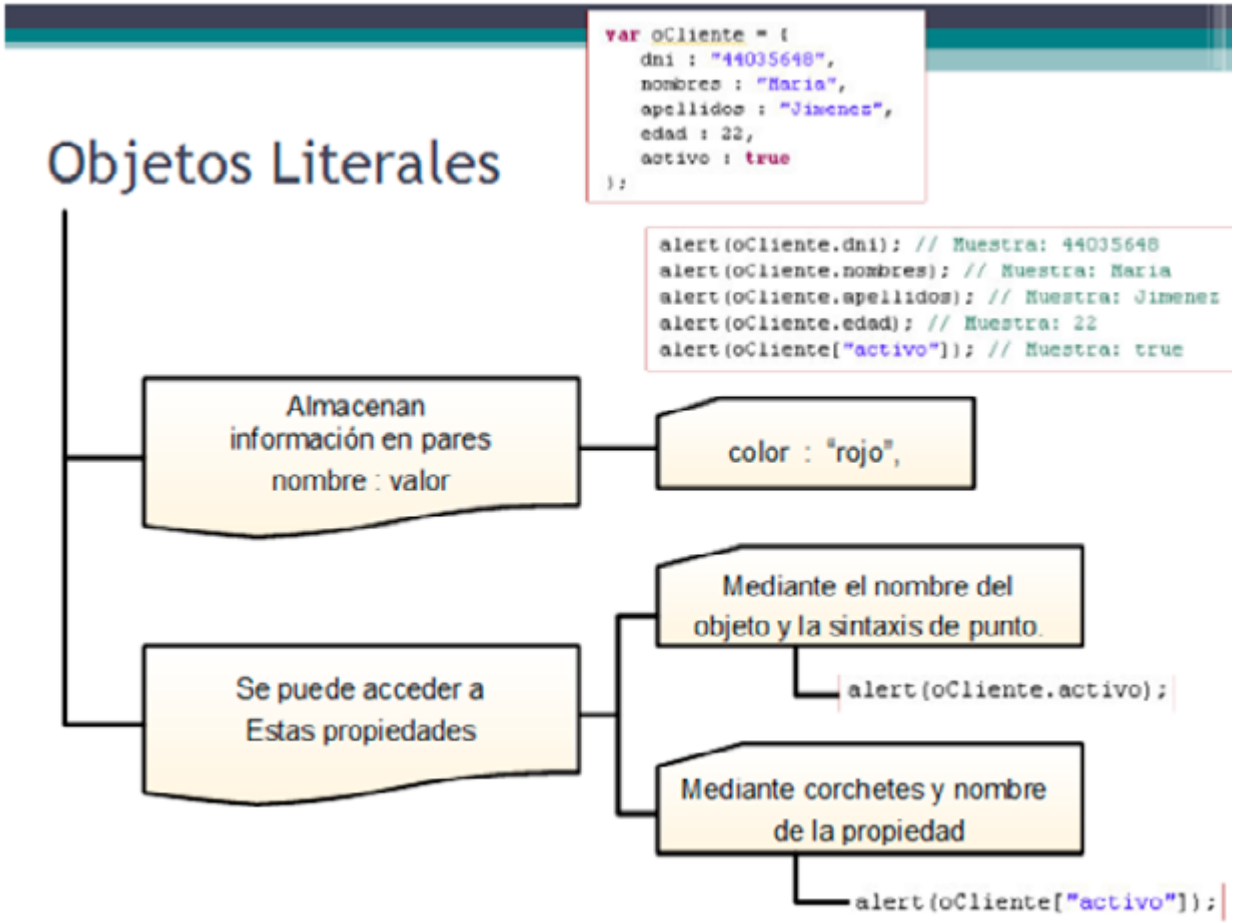
1.1 | Particularidades JSON sobre JavaScript

Algunas de las particularidades o reglas del formato JSON a tener en cuenta son:

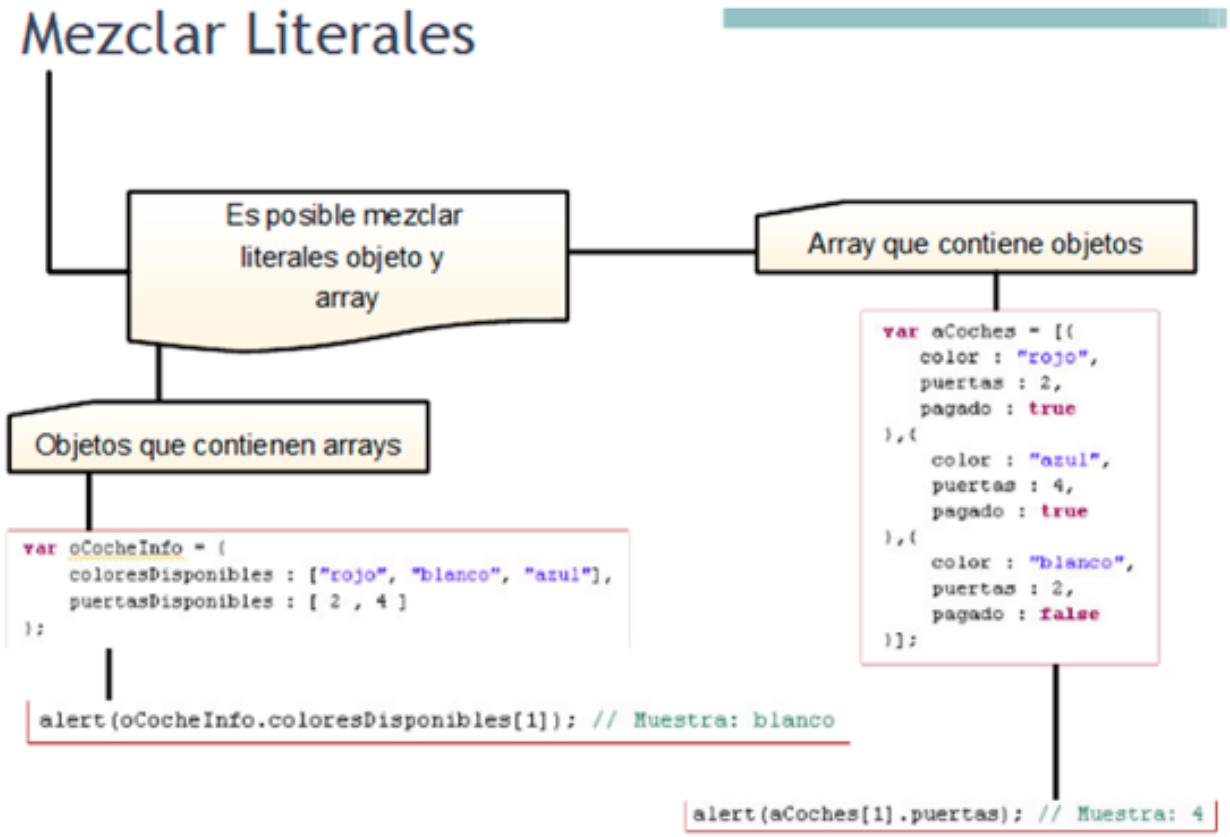
- Son **duplas nombre-valor** y los nombres van delimitados por comillas, tanto simples como dobles, aunque pueden aparecer sin ellas.
- **JSON** puede representar los seis tipos de valores de JavaScript: objetos, Arrays, números, cadenas, booleanos y null.



- Las **fechas** no son un tipo de objeto propio.
- Los **números** no pueden ir precedidos de ceros a no ser en el caso de notación decimal (Ejemplo: 0.001).
- **JSON** es considerado un lenguaje independiente
- Sus objetos deben ser considerados como **cadenas Javascript**, no como objetos nativos.



Es posible mezclar objetos y arrays, arrays y objetos, de esta manera:



1.2 | Transformar JSON a String y String a JSON

Muchos de los códigos JavaScript que utilizan JSON para su funcionamiento o transmisión de información necesitan transformar en numerosas ocasiones a **String** (cadenas de texto) y viceversa.

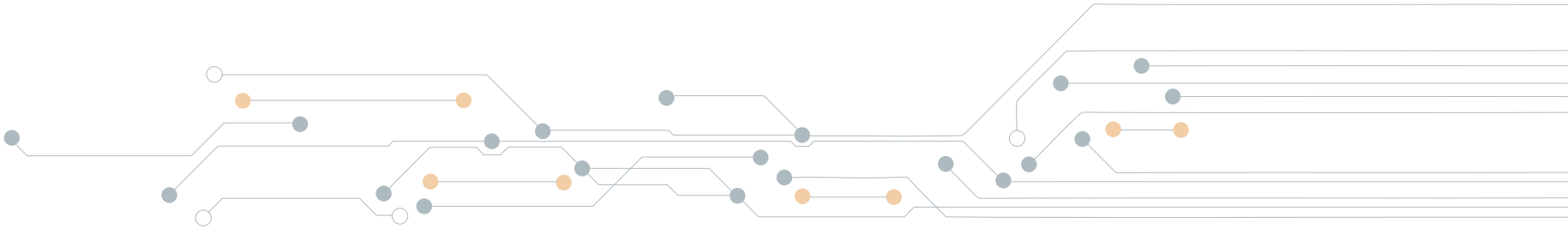
Hay muchas maneras de hacer eso, antiguamente se utilizaba una función denominada "eval" que realizaba esta transformación (no sin muchos fallos y desventajas)

También se pueden realizar estas transformaciones con bibliotecas de terceros como pueden ser las de **JQuery** y **Mootools** (entre otras).

Sin embargo, con la llegada del **ECMAScript 5,** se ha implementado un nuevo objeto **JSON** basado en la **API** programada por el propio Douglas Crockford. Sus métodos más interesantes son **parse()** y **stringify().**

parse() → transforma de string a json.

stringify() →transforma de json a string.



2. El estándar DOM

En el momento en el que se desarrolló el lenguaje XML, apareció la imprescindible necesidad de procesamiento y manipulación del contenido de los archivos XML mediante los lenguajes de desarrollo.

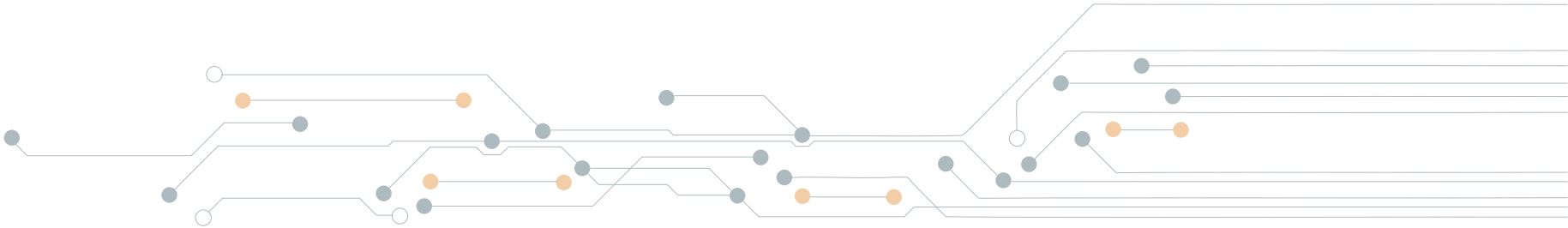
XML es sencillo de redactar pero complejo para su procesamiento y manipulación. Por ello, surgen algunas técnicas entre las que se encuentra *DOM*.

DOM (Document Object Model) es un agregado de utilidades diseñadas para la manipulación de XML. Además, DOM también se usar para manipulación de documentos *XHTML y HTML*.

DOM es una *API* de funciones que se pueden usar para la manipulación de las páginas *XHTML* de forma eficiente y rápida.

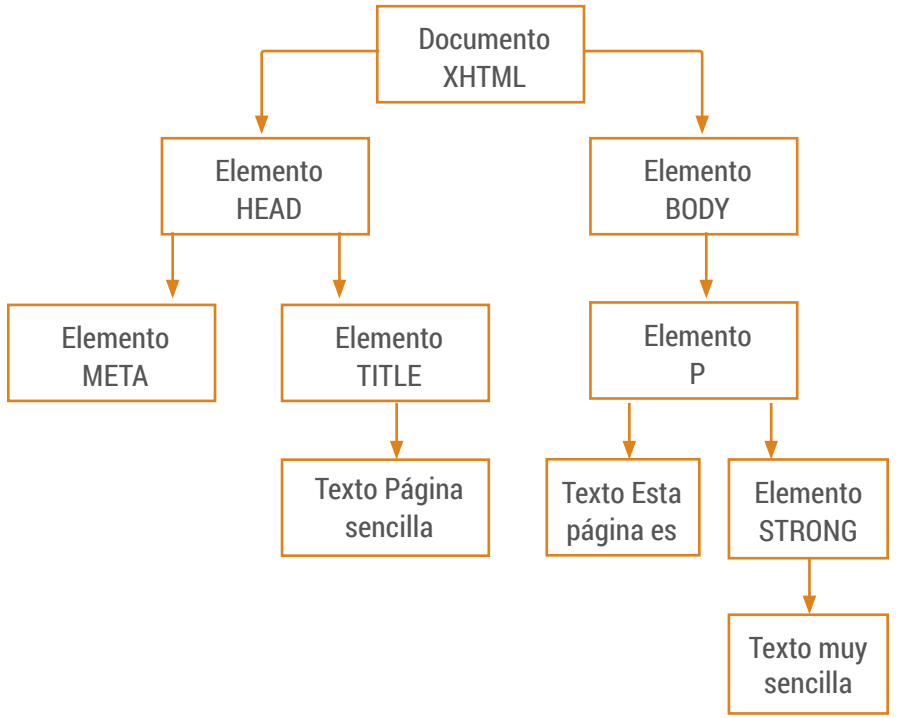
Antes de usar las funciones, *DOM* convierte internamente el *XML* original en una estructura fácilmente manejable formada por una jerarquía de nodos. De esta manera, *DOM* transforma el *XML* en una serie de nodos interconectados en árbol.

```
<!DOCTYPE  html PUBLIC ”-//W3C//DTD XHTML 1.0
Transitional//EN” “http://www.w3.org/TR/xhtml1/DTD/
xhtml1-transitional.dtd”>
<html xmlns=”http://www.w3.org/1999/xhtml”>
  <head>
     <meta http-equiv=”Content-Type” content=”text/html;
charset=iso-8859-1” />
      <title>Página sencilla</title>
   </head>
    <body>
       <p>Esta página es <strong>muy sencilla</strong></
p>
    </body>
</html>
```



El árbol que se genera no representa únicamente los contenidos del fichero origen (mediante los nodos del árbol) sino que representa sus relaciones (mediante las ramas del árbol que conectan los nodos).

En ocasiones DOM se asocia con el desarrollo web y con JavaScript, la API de DOM es independiente de cualquier lenguaje de desarrollo. DOM está disponible en la mayoría de lenguajes de desarrollo empleados comúnmente.



2.1 | Manipulación de un documento HTML con DOM JavaScript

Una de las principales ventajas del uso del DOM es que permite a los desarrolladores web disponer de un control preciso sobre la estructura o forma del documento HTML o XML que están controlando. Las funciones que usa DOM permiten añadir, eliminar, modificar y reemplazar cualquier nodo de cualquier documento sencillamente.

La primera especificación de DOM (DOM Level 1) se definió en el año 1998 y homogeneizó el desarrollo del DHTML o HTML dinámico en todos los navegadores, ya que permitía variar el contenido de las páginas sin necesidad de volver a cargar toda la página entera.

Los documentos *XML* y *HTML* son convertidos por DOM en una jerarquía de nodos. Los nodos pueden ser de diferentes tipos.

- **Document:** nodo raíz de los documentos HTML y XML. Todos los demás salen de él.
- **DocumentType:** nodo que contiene la representación del *DTD* empleado en la página (indicado mediante el *DOCTYPE*).
- **Element:** contenido definido por un par de etiquetas (o tags) de apertura y cierre *(<etiqueta>...</etiqueta>)* o de una etiqueta abreviada que se autocierra *(<etiqueta/>)*. Es el único nodo que puede tener tanto nodos hijos como atributos.
- **Attr:** el par nombre-de-atributo/valor.
- **Text:** el contenido del texto que se halla entre una etiqueta de apertura y una de cierre. También guarda el contenido de una sección de tipo *CDATA.*
- **CDataSection:** nodo que muestra una sección tipo *<![CDATA[ ]]>.*
- **Comment:** un comentario de XML.
- Y otros menos usuales: DocumentFragment, Entity, EntityReference, ProcessingInstruction y Notation.

2.2 | Selectores y DOM

Una vez DOM ha formado automáticamente el árbol completo de nodos de la página, ya es posible usar sus funciones para obtener la información sobre los nodos o controlar su contenido.

JavaScript crea el objeto "Node" para tratar las propiedades y métodos necesarios para el procesamiento y manipulación de los documentos.

Primeramente, el objeto Node crea las siguientes constantes para la identificación de los tipos de nodos:

- Node.ELEMENT_NODE =1
- Node.ATTRIBUTE_NODE = 2
- Node.TEXT_NODE = 3
- Node.CDATA_SECTION_NODE = 4
- Node.ENTITY_REFERENCE_NODE= 5
- Node.ENTITY_NODE = 6
- Node.PROCESSING_INSTRUCTION_NODE = 7
- Node.COMMENT_NODE= 8
- Node.DOCUMENT_NODE = 9
- Node.DOCUMENT_TYPE_NODE= 10
- Node.DOCUMENT_FRAGMENT_NODE = 11
- Node.NOTATION_NODE=12

Y a partir de este momento, podemos usar cualquier función o propiedad de DOM para nuestro código JavaScript.

Propiedad / Método	Valor devuelto	Descripción
nodeName	String	El nombre del nodo (no está definido para algunos tipos de nodo)
nodeValue	String	El valor del nodo (no está definido para algunos tipos de nodo)
nodeType	Number	Una de las 12 constantes definidas anteriormente
ownerDocument	Document	Referencia al documento al que pertenece el nodo
firstChild	Node	Referencia del primer nodo de la lista childNodes
lastChild	Node	Referencia del último nodo la lista childNodes
childNodes	Nodelist	Lita todos los nodos hijo del nodo actual
previousSibling	Node	Referencia del nodo hermano anterior o null si este nodo es el primer hermano
hasChildNodes()	Bolean	Devuelve true si el nodo actual tiene uno o más hijos
Attributes	NameNodeMap	Se emplea con nodos de tipo Element. Contiene objetos tipo Attr que definen todos los atributos del elemento
appendChild(nodo)	Node	Añade un nuevo nodo al final de la lista childNodes
removeChild(nodo)	Node	Elimina un nodo de la lista childNodes
replaceChild(nuevoNodo, anteriorNodo)	Node	Reemplaza el nodo anteriorNodo por el nodo nuevoNodo
insertBefore(nuevoNodo, anteriorNodo)	Node	Inserta el nodo nuevoNodo antes que la posición del nodo anteriorNodo dentro de la lista childNodes

Telefónica

EDUCACIÓN DIGITAL