

# El juego de caracteres UTF-8

En esta lección se explica qué es un juego de caracteres informático y, concretamente, qué es el juego de caracteres UTF-8 y su uso en HTML.

Esta lección es la primera lección de un grupo de tres lecciones que se recomienda leer en el siguiente orden:

- [El juego de caracteres UTF-8](#), que explica qué es un juego de caracteres informático y, concretamente, qué es el juego de caracteres UTF-8.
- [Entidades de carácter y entidades numéricas](#), que explica cómo incluir caracteres Unicode en una página web
- [Secuencias](#), que muestra cómo se definen las secuencias de caracteres Unicode que corresponden a dibujos (banderas, emojis, etc.)

Como complemento a las lecciones anteriores, las páginas siguientes muestran diferentes símbolos Unicode agrupados por temas:

- [Símbolos gráficos](#), que muestra los caracteres Unicode que corresponden a símbolos gráficos (incluye tanto los símbolos en blanco y negro como los emojis que tiene versión en modo texto)
- [Emojis](#), que muestra únicamente los caracteres Unicode que corresponden a los emojis (en color)
- [Géneros](#), que muestra las secuencias Unicode que corresponden a personas con un género (masculino o femenino) determinado
- [Colores de piel](#), que muestra las secuencias Unicode que corresponden a personas con un color de piel determinado
- [Parejas](#), que muestra las secuencias Unicode que corresponden a parejas y familias
- [Banderas](#), que muestra las secuencias Unicode que corresponden a banderas
- [Otras](#), que muestra las secuencias Unicode no incluidas en las páginas anteriores
- [Twemoji](#), que muestra los dibujos de emojis creados por el proyecto [Twemoji](#) con licencia libre [CC-BY 4.0](#) (es un fichero grande que puede tardar un poco en cargarse)

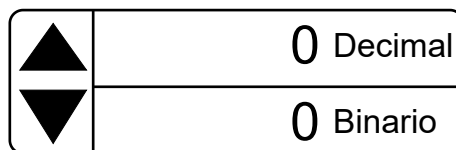
## Bits

En informática, los números no se guardan en notación decimal (con las cifras del 0 al 9), sino en notación binaria, utilizando únicamente el 0 y el 1. Como en la notación decimal, para escribir números en la notación binaria se utilizan varias cifras seguidas, pero en este caso utilizando únicamente ceros y unos.

Estos son los primeros números naturales escritos en notación decimal y binaria:

<b>Decimal</b>	0	1	2	3	4	5	6	7	8	...
<b>Binaria</b>	0	1	10	11	100	101	110	111	1000	...

La figura siguiente permite ver la notación decimal y binaria de los enteros entre 0 y 255. Para subir o bajar el valor, haga clic en los triángulos:



En notación decimal, con  $n$  cifras podemos escribir  $10^n$  números. Es decir, con 1 cifra podemos escribir 10 números (del 0 al 9), con 2 cifras podemos escribir 100 números (del 00 al 99), con 3 cifras podemos escribir 1000 números (del 000 al 999), etc.

En notación binaria, con  $n$  cifras podemos escribir  $2^n$  números. Es decir, con 1 cifra podemos escribir 2 números (0 y 1), con 2 cifras podemos escribir 4 números (del 00 al 11, es decir, del 0 al 3), con 3 cifras podemos escribir 8 números (del 000 al 111, es decir, del 0 al 7), etc.

A las cifras binarias se les llama *bit*, es decir el número binario 1001 es un número de 4 bits, de la misma manera que el número decimal 329 es un número de 3 cifras decimales.

## Bytes

Para realizar su trabajo más rápido, los ordenadores no trabajan con cifras individuales (*bits*), sino con varias cifras a la vez. La agrupación más utilizada es el *byte*, que es la agrupación de 8 bits. Un byte puede representar por tanto 256 valores distintos ( $2^8$ , del 00000000 al 11111111, es decir, del 0 al 255).

A medida que ha avanzado la informática, los ordenadores han ido trabajando con grupos de bits cada vez más grandes. En 1981, el primer PC tenía un procesador que trabajaba con grupos de 8 bits (1 byte). En 1984 los PCs empezaron a utilizar procesadores de 16 bits (2 bytes). En 1985 los PCs empezaron a utilizar procesadores de 32 bits (4 bytes). En 2003 los PCs empezaron a utilizar procesadores de 64 bits (8 bytes). Actualmente (noviembre de 2022) no existen PCs con procesadores de 128 bits (16 bytes), pero algunos componentes de los PCs como las procesadores gráficos ya trabajan con grupos de 256 o incluso 512 bits.

El byte se utiliza en Informática como unidad de medida en el almacenamiento de información (capacidad de los discos duros o memorias). Como las capacidades de los dispositivos son ahora tan grandes, normalmente se expresan en múltiplos que corresponden a potencias de 2. Habitualmente e históricamente estos múltiplos se suelen llamar Kilobytes, Megabytes, Gigabytes, Terabytes, etc. aunque realmente estos nombres no son correctos ya que, en este contexto, Kilo no significa mil (1000), sino 1024, ya que 1024 es una potencia de dos ( $2^{10} = 1024$ ), Mega no significa millón sino 1.048.576 ( $1024 \times 1024$ ), etc. Los nombres correctos para estas unidades son kibibyte (1 KiB = 1024 bytes), mebibyte (1 MiB = 1024 KiB = 1.048.576 bytes), gibibyte (1 GiB = 1024 MiB), tebibyte (1 TiB = 1024 GiB), etc. Estas unidades se definieron en 1998, pero raramente se usan.

## Notación hexadecimal

El inconveniente de la notación binaria es que para escribir un número grande se necesitan muchos bits. Una manera de acortar los números en binario es utilizar la notación hexadecimal, que utiliza 16 dígitos distintos (del 0 al 9, A, B, C, D, E y F).

En notación hexadecimal, con  $n$  cifras podemos escribir  $16^n$  números. Es decir, con 1 cifra podemos escribir 16 números (del 0 a F), con 2 cifras podemos escribir 256 números (del 00 al FF, es decir, del 0 al 255), con 3 cifras podemos escribir 4096 números (del 000 al FFF, es decir, del 0 al 4095), etc.

Pasar de la notación binaria a la hexadecimal (o viceversa) es muy sencillo, porque cada cuatro bits de la notación binaria corresponden a un único carácter hexadecimal.


La figura siguiente permite ver la notación binaria, hexadecimal y decimal de los enteros entre 0 y 255. Para subir o bajar cada bit, haga clic en los triángulos:

	
Binario	0
Hexadecimal	0
Decimal	0

## El juego de caracteres ASCII

En Informática toda la información se guarda en forma numérica, por lo que para guardar caracteres (letras, números, signos de puntuación, etc.) se necesita establecer una correspondencia entre los códigos numéricos y los caracteres. Al conjunto de correspondencias se le llama juego de caracteres. Los primeros ordenadores usaban cada uno su propio juego de caracteres, pero en los años 50 empezaron a proponerse estándares que facilitasen la comunicación entre diferentes sistemas. En los años 60 se creó el [código ASCII](#), que definía 128 caracteres (33 caracteres de control y 95 imprimibles -letras minúsculas, mayúsculas, cifras, puntuación, etc.-) y que se convirtió en el estándar en informática.


La figura siguiente permite ver el carácter correspondiente a cada valor del código ASCII (algunos valores no son visibles).

	80 Decimal P ASCII
--	-----------------------

En el código ASCII cada carácter ocupaba precisamente 1 byte, es decir, 8 bits: 7 bits para codificar los 128 caracteres y 1 bit de paridad para comprobar errores. Ese juego de caracteres era suficiente para el idioma inglés, pero era insuficiente para otros idiomas, por lo que a medida que la informática se extendía por el mundo se tuvieron que crear otros juegos de caracteres.

Una forma de ampliar los caracteres disponibles manteniendo el tamaño de un byte por carácter fue abandonar el bit de paridad y utilizar los ocho bits para codificar caracteres. De esa manera el juego de caracteres podía contener 256 caracteres distintos. Los primeros 128 eran los de ASCII, pero los siguientes se podían utilizar para incluir otros caracteres. El más utilizado de esos juegos es el [ISO 8859-1](#), que incluye los caracteres que necesitan los idiomas de Europa occidental (acentos agudos, graves, diéresis, etc.). Windows utiliza un juego de caracteres, el [Windows-1252](#), que tiene algunos caracteres distintos, pero que se suele considerar equivalente.

La figura siguiente permite ver el carácter correspondiente a cada valor del juego de caracteres Windows-1252 (algunos valores no son visibles).

	126 Decimal ~ CP1252
---	-------------------------

Pero pronto los juegos de caracteres de un byte se encontraron con barreras importantes:

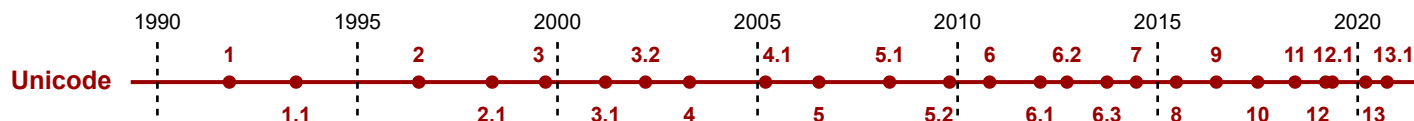
- por un lado, 256 caracteres son insuficientes para cubrir todos los idiomas, por lo que se utilizaban juegos de caracteres distintos para idiomas que necesitaban otros caracteres (por ejemplo, para el alfabeto cirílico). El problema es que escribir sistemas operativos y programas capaces de gestionar tantos juegos de caracteres diferentes es muy complicado.
- por otro lado, muchos lenguajes escritos (sobre todo los asiáticos) utilizan más de 256 caracteres distintos, por lo que 1 byte era en cualquier caso insuficiente para codificar su juego de caracteres

## El juego de caracteres universal: Unicode

La llegada de Internet y el aumento consiguiente del intercambio de información entre diferentes países hizo aún más patente la necesidad de creación de un juego de caracteres universal, que abarcara las necesidades de toda la humanidad. La idea se empezó a trabajar en 1987 y en 1991 se publicó la primera versión de Unicode,

La principal ventaja de Unicode es que permite que la información guardada pueda ser interpretada correctamente por cualquier sistema informático. Por esa razón, cualquier comunidad tiene interés en incluir su sistema de escritura en Unicode, no solamente las lenguas en uso actual sino también las lenguas muertas, símbolos y pictogramas. Por ello Unicode no ha dejado de crecer y la última versión publicada en septiembre de 2022, [Unicode 15.0](#), contiene casi 150.000 caracteres.

Las primeras versiones de Unicode se publicaron más o menos cada dos años pero desde 2009 Unicode se publica anualmente. La versión 14.0 se publicó con seis meses de retraso debido a la epidemia del coronavirus Covid-19, y a partir de esta versión Unicode se publicará en el tercer trimestre de cada año (en principio, en septiembre).



La primera versión de Unicode utilizaba dos bytes para codificar cada carácter, pero con el paso del tiempo los dos bytes se quedaron cortos y la norma actual utiliza cuatro bytes para codificar los caracteres. El inconveniente de utilizar 4 bytes para codificar los caracteres es que se ocupa cuatro veces más espacio que codificando con un byte y se necesita cuatro veces más tiempo para transmitirlos. Para optimizar, se decidió que los primeros valores de Unicode coincidieran con los de ASCII, de manera que los textos ingleses pudieran seguir utilizando un byte por carácter. Aunque el tema es bastante complejo, se puede resumir en que existen tres opciones:

- UTF-32: cada carácter ocupa cuatro bytes.
- UTF-16: cada carácter ocupa dos bytes si está entre los primeros 65.536 (256x256) caracteres de Unicode y cuatro bytes para el resto de caracteres.
- UTF-8: cada carácter ocupa un byte si está entre los primeros 128 caracteres de Unicode, 2 bytes si está entre los primeros 2048 caracteres, 3 bytes si está entre los primeros 65536 (256x256) o cuatro bytes para el resto de caracteres.

#### Notas:

- Unicode no utiliza todos los bits de cada carácter. UTF-8 de 1 byte utiliza 7 bits (incluye 128 caracteres), UTF-16 de 2 bytes utiliza 11 bits (incluye 1920 caracteres de los 2048 posibles), etc.
- Existen dos variantes de UTF-16 y de UTF-32: Big Endian, en la que el byte más significativo se pone en primera posición, y Little Endian, en la que el byte más significativo se pone en segunda posición.

La tabla siguiente muestra algunos caracteres codificados en Windows-1252, UTF-8, UTF-16 o UTF-32 (big y little endian):

Carácter	a	á	e	é	€	♫
CP1252	61	E1	65	E9	80	
UTF-8	61	C3 A1	65	C3 A9	E2 82 AC	F0 9D 84 9E
UTF-16 (BE)	00 61	00 E1	00 65	00 E9	20 AC	D8 34 DD 1E
UTF-16 (LE)	61 00	E1 00	65 00	E9 00	AC 20	34 D8 1E DD
UTF-32 (BE)	00 00 00 61	00 00 00 E1	00 00 00 65	00 00 00 E9	00 00 20 AC	00 01 D1 1E
UTF-32 (LE)	61 00 00 00	E1 00 00 00	65 00 00 00	E9 00 00 00	AC 20 00 00	1E D1 01 00

Se puede consultar el [juego de caracteres Unicode completo](#), dividido en secciones correspondientes a sistemas de escritura y conjuntos de símbolos.

Anuncio de nuevas versiones Unicode: [5.2 \(2009\)](#) - [6.0 \(2010\)](#) - [6.1 \(2012\)](#) - [6.2 \(2012\)](#) - [6.3 \(2013\)](#) - [7.0 \(2014\)](#) - [8.0 \(2015\)](#) - [9.0 \(2016\)](#) - [10.0 \(2017\)](#) - [11.0 \(2018\)](#) - [12.0 \(2019\)](#) - [12.1 \(2019\)](#) - [13.0 \(2020\)](#) - [13.1 \(2020\)](#) - [14.0 \(2021\)](#) - [15.0 \(2022\)](#).

## Juegos de caracteres en páginas web

En las primeras versiones de HTML se especificaba que el juego de caracteres utilizado por las páginas web debía ser el juego de caracteres ISO 8859-1, más conocido como Latin-1.

En 1997 con la norma [RFC 2070](#) se permitió el juego de caracteres ISO 10646 (equivalente a Unicode).

En 2008 UTF-8 ya era el formato más utilizado en la web.

Desde 2014 la recomendación HTML 5 especifica que UTF-8 es el juego caracteres predeterminado.

## Tipos de letra Unicode

Para poder mostrar los caracteres Unicode, se necesita disponer de tipos de letra que los incluyan. Cada sistema operativo incluye sus propios tipos de letra predeterminados, que poco a poco van incluyendo más caracteres Unicode, con más o menos retraso con respecto a la norma.

- En Windows, las fuentes que incluyen los caracteres Unicode son las fuentes [Segoe UI Symbol](#) y [Segoe UI Emoji](#), que incluyen tanto la representación en modo texto, en blanco y negro, como la representación en modo emoji, en color. Hay que tener en cuenta que Windows 10 y Windows 11 tienen versiones distintas de estas fuentes, que sólo se van actualizando en Windows 11.
- Las distribuciones GNU/Linux solían incluir la fuente [Symbola](#), a veces bajo un nombre distinto. Es una fuente en blanco y negro, que hasta enero de 2018 se distribuía con una licencia permisiva, pero a partir de la versión 11.0 (publicada en junio de 2018), pasó a ser solo gratuita para usos no comerciales ni educativos. [\[hilo sobre Symbola en Fedora \(2018\)\]](#).
- **falta hacer referencia a las fuentes usadas en otros sistemas operativos**

Existen algunos proyectos de creación de tipos de letra libres que incluyan al menos los caracteres de Unicode que corresponden a pictogramas. El problema es que las fuentes en color pueden usar hasta cinco formatos distintos (SBIX, COLRV0, COLRV1, SVG-in-OpenType, CBDT/CBLC) cuyo soporte es bastante desigual [\[ChromaCheck](#) permite comprobar el soporte de esos formatos en el navegador].

- [Noto Emoji](#), un proyecto creado por Google en 2014. Es una fuente en color (la versión en blanco y negro se considera obsoleta), disponible en los formatos CBDT/CBLC o COLRV1. Actualmente (noviembre de 2022) Chrome admite ambos formatos y Firefox admite únicamente el formato COLRV1 si se activa la preferencia `gfx.font_rendering.colr_v1.enabled`, pero en Firefox 107 [está previsto](#) que admita COLRV1 sin necesidad de activar esa preferencia.

- [Twemoji](#), un proyecto creado por Twitter en 2014. Es una fuente en color, en formato SVG-in-OpenType, que Google Chrome actualmente (noviembre de 2022) no es capaz de mostrar. Actualmente (noviembre de 2022), Twemoji solamente incluye hasta Unicode 14 y aunque internamente tenían disponible ya Unicode 15, los despidos posteriores a la compra de Twitter por parte de Elon Musk han paralizado su publicación.
- [twemoji-colr](#), un proyecto de Mozilla que rehace la fuente Twemoji en formato COLR. Es una fuente en color, en formato COLR, que admiten tanto Firefox como Google Chrome. Firefox la utiliza para mostrar los emojis no incluidos en Segoe en Windows. La versión que Firefox utiliza actualmente (noviembre de 2022) es la versión 0.6, que solamente incluye hasta Unicode 13.

Estas fuentes se pueden emplear en una página web utilizando el mecanismo de las fuentes web, que se comenta en la [lección CSS: Fuentes web](#)

Otros proyectos parecen prácticamente abandonados o se han convertido en proyectos comerciales:

- [EmojiOne](#), un proyecto creado en 2014 que se centraba principalmente en los caracteres emoji, aunque incluía muchos otros símbolos, con un diseño bastante trabajado y a todo color. Por desgracia en abril de 2017 el proyecto se convirtió en un proyecto comercial.
- [EmojiTwo](#), un *fork* de EmojiOne creado en 2017 tras el cambio de licencia de EmojiOne, pero que no ha publicado ninguna nueva versión hasta ahora.

---

## Pictogramas en Unicode

Unicode no solamente contiene caracteres para todas las lenguas del mundo, sino que también incluye una gran cantidad de pictogramas. Una gran parte de ellos son símbolos gráficos e iconos de amplia difusión que ya estaban incluidos en juegos de caracteres anteriores, pero desde 2009 (Unicode 5.2) se están incorporando también imágenes más sofisticadas, como los [emoticonos](#) o los [emojis](#), creados en Japón a finales de los 90 para los dispositivos móviles. Además, mediante el mecanismo de las [secuencias de caracteres](#), introducido en Unicode 4.1 (2005), se han ido añadiendo nuevos dibujos, como las banderas, y especialmente las variantes de género, color de piel o grupos de individuos

Unicode contiene así miles de pictogramas, que se irán ampliando en las próximas versiones (véase [Unicode roadmap](#)).

Estos son, por ejemplo, los emojis añadidos en las últimas versiones de Unicode: [15.0 \(2022\)](#) - [14.0 \(2021\)](#) - [13.1 \(2020\)](#) - [13.0 \(2020\)](#) - [12.1 \(2019\)](#) - [12.0 \(2019\)](#) - [11.0 \(2018\)](#)

Estos son algunas de las tablas de caracteres de Unicode que contienen pictogramas:

- [PDF](#) - Controles y Latín básico
- [PDF](#) - Controles y Latín 1 (suplemento)
- [PDF](#) - Puntuación
- [PDF](#) - Símbolos de monedas
- [PDF](#) - Símbolos con letras
- [PDF](#) - Flechas
- [PDF](#) - Símbolos técnicos misceláneos
- [PDF](#) - Símbolos alfanuméricos con círculo alrededor
- [PDF](#) - Cajas
- [PDF](#) - Formas geométricas
- [PDF](#) - Símbolos misceláneos
- [PDF](#) - Dingbats
- [PDF](#) - Flechas suplementarias B
- [PDF](#) - Símbolos y flechas misceláneos
- [PDF](#) - Símbolos y puntuación CJK
- [PDF](#) - Símbolos CJK con círculo alrededor
- [PDF](#) - Símbolos musicales
- [PDF](#) - Fichas de Mahjong
- [PDF](#) - Fichas de dominó
- [PDF](#) - Cartas
- [PDF](#) - Alfanuméricos con círculo alrededor (suplemento)
- [PDF](#) - Ideográficos con círculo alrededor (suplemento)
- [PDF](#) - Símbolos y pictogramas misceláneos
- [PDF](#) - Emoticonos
- [PDF](#) - Dingbats decorativos
- [PDF](#) - Símbolos de transporte y mapas
- [PDF](#) - Símbolos alquímicos
- [PDF](#) - Formas geométricas extendidas
- [PDF](#) - Símbolos y pictogramas misceláneos suplementarios
- [PDF](#) - Fichas de ajedrez
- [PDF](#) - Símbolos y pictogramas extendidos A
- [PDF](#) - Símbolos informáticos antiguos

---

Última modificación de esta página: 7 de noviembre de 2022



Esta página forma parte del curso [Páginas web HTML y hojas de estilo CSS](#) por [Bartolomé Sintes Marco](#)

