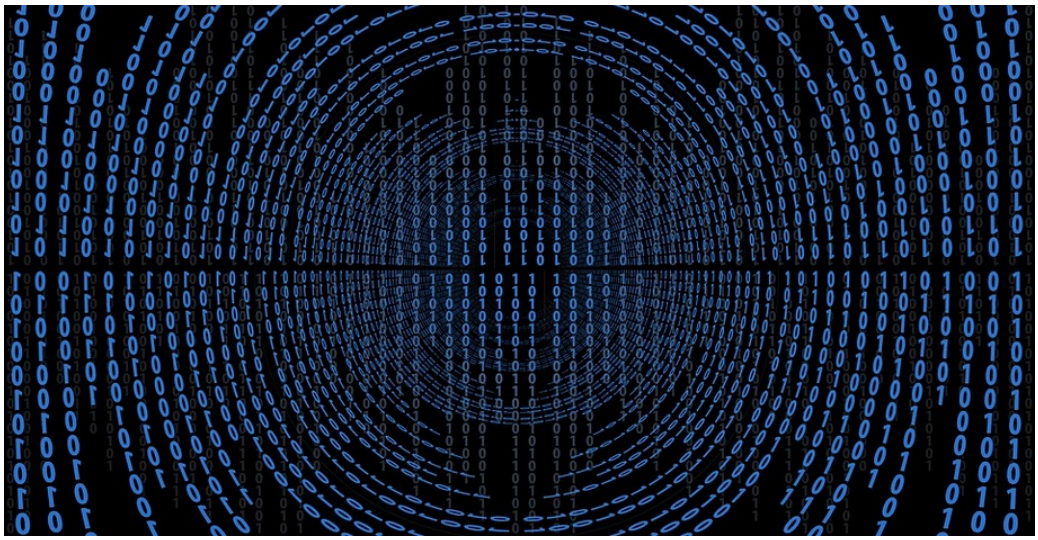
Bits y bytes. Cómo funciona la información digital.

Publicado el El Informatico - 10 de diciembre de 2021 -



¿Cuántas veces habrás oido que la información en un ordenador son en realidad "ceros y unos"? Seguro que muchas veces, pero quizás aún no sepas ni comprendas exáctamente a que se refiere o cómo funcionan éstos "ceros y unos" de los que se hablan. En éste artículo aprenderás cómo funciona la información digital y cómo se digitaliza dicha información.

¿Que es eso de "digital"?

Cuando hablamos de algo "digital" hablamos de una pieza de tecnología electrónica con componentes electrónicos cuyas entradas y salidas tienen dos estados claramente diferenciados. Probablemente estés familiarizado con los términos "*digital*" y "*analógico*". En electrónica, un circuito analógico es un circuito en el que el voltaje a la entrada y a la salida de los componentes puede tener cualquier valor. Por ejemplo, 15 voltios, o -13.5 voltios, o 35 voltios, que además pueden ser en corriente continua, o alterna. O incluso puede tratarse de una señal analógica como es el caso, por ejemplo, de las señales de radio o televisión. Lo importante es que **éstas señales pueden tener cualquier valor**.

En el caso de los *circuitos digitales*, los voltajes **siempre tienen dos valores claramente diferenciados**. En éste caso, distinguimos siempre entre **valor alto**

(H) y **valor bajo** (L). Ambos valores pueden ser cualquier cosa dependiendo del diseño del circuito y los valores que acepten los componentes del mísmo, pero por norma general suelen ser 5V de valor alto, y 0V de balor bajo.

Estos valores componen lo que en electrónica digital se denominan **unidades de información**.

La unidad básica de la información digital es el **bit**. Siendo el bit la representación de un valor en un circuito digital.

Numeración binaria

Como representamos la información mediante bits, que son los diferentes valores del circuito digital, y dichos bits pueden tener dos valores claramente diferenciados (y cuyos valores alto y bajo pueden variar entre circuitos), usamos un sistema de numeración **binaria** para representar los datos que se manejan en el circuito.

El sistema de numeración en binario es sencillo de comprender. De hecho, me atrevería a decir que ya sabes binario, aunque aún no lo sepas.

Para explicar lo que es el sistema binario, tomemos por ejemplo un sistema de numeración que nosotros conocemos muy bien y que usamos a diario, que es el sistema **decimal**. Cuando nosotros contamos en decimal, contamos con un único dígito desde el número 0 hasta el número 9. Pero **como es un sistema decimal, sólo tenemos diez dígitos para contar**. Así que cuando queremos pasar del 9 al siguiente valor, tenemos que añadir otro dígito a la izquierda. Que en éste caso sería el 10, y que representa cuántas vueltas hemos dado al primer dígito. Así volveríamos a contar desde el 10 al 19, y tendríamos que sumar 1 al segundo dígito para ir al siguiente número que es el 20. Repite ésto hasta el 99, y tendrás que añadir otro dígito a la izquierda para pasar al número 100. Y así constantemente.

El sistema de numeración binario es igual, pero con la diferencia de que contamos desde 0 hasta 1. Y como es un sistema binario, sólo tenemos dos dígitos para contar. Así que la siguiente cifra despues del 1, sería el 10. Despues contaríamos 10, 11, 100, 101, 110, 111, 1000, y así constantemente.

DECIMAL	BINARIO
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001
10	1010
11	1011
12	1100
13	1101
14	1110
15	1111
16	10000
17	10001
18	10010
19	10011
20	10100

Los primeros 20 números, representados en binario

Ahora bien, ¿Qué es un 1 y qué es un 0 en un circuito digital? Al igual que antes, todo depende del diseño y la interpretación del circuito. Pero **por norma general, decimos que un valor alto (H) representa un 1, y que un valor bajo (L) representa un 0**.

Hablaré más sobre éste y otros sistemas de numeración en otro artículo.

Unidades de información

La unidad básica de información es el bit, pero evidentemente con un bit sólo podemos representar un único digito, que puede ser un 0 o un 1. Si queremos representar valores numéricos más altos, necesitaremos agrupar más bits para hacer unidades más grandes.

La unidad más grande despues del bit es el *nibble*, compuesto de 4 bits. Pero en informática y en la mayoría de circuitos digitales usamos el **byte**. **Un *byte* equivale a 8 bits, y podemos representar cifras de hasta 256 valores distintos,**

desde el 0 hasta el 255 o desde el -127 hasta el 127.

UNIDADES DE BITS (SI)	
Bit (b)	1
Nibble	4 bits
Byte (B)	8 bits
Kilobit (Kb)	1Kb = 1000 bits
Megabit (Mb)	1Mb = 1000 Kb
Gigabit (Gb)	1Gb = 1000 Mb
Terabit (Tb)	1Tb = 1000 Gb
Petabit (Pb)	1Pb = 1000 Tb
Exabit (Eb)	1Eb = 1000 Pb

Unidades de bits en el sistema internacional (SI)

Aquí es cuando el tema de las unidades se complica un poco, así que presta atención si realmente quieres comprender ésto. Aunque un byte es una agrupación de 8 bits, **nosotros tratamos el byte como una unidad individual**. Esto quiere decir que **podemos agrupar bits individuales** (Se usa sobre todo para los anchos de banda en redes, por poner un ejemplo)**, o podemos agrupar bytes (conjuntos de 8 bits) de forma individual**.

Si seguimos agrupando bits, la siguiente unidad sería el **kilobit**. **Un kilobit son 1000 bits**. Despues el **megabit** (1000000 bits, o 1000 kilobits), el **gigabit** (1000 megabits), el **terabit** (1000 gigabits), el **petabit** (1000 terabits), etcétera.

Pero si agrupamos bytes, y aquí es donde viene la parte más confusa, **podemos hacer agrupaciones de 1000, o agrupaciones de 1024 de cada unidad**. Pero **todo depende del estandard que estemos utilizando**.

Según el estandard ISO/IEC 80000-13, si hacemos una agrupación de 1000 bytes, hablamos de un **kilobyte** (**KB**). Mientras que si la agrupación es de 1024, entonces hablamos de un **kibibyte** (**KiB**). El estandard ISO es el estandard que siguen casi todos los países, y por lo tanto deberían de seguir todos los circuitos electrónicos y cómo no, todos los fabricantes y desarrolladores de Software. Si usas Linux, éste es el estandard que se utiliza en Linux.

AGRUPACIONES DE 1000 BYTES		AGRUPACIONES DE 1024 BYTES	
Kilobyte (KB)	1 KB = 1000 B	Kibibyte (KiB)	1 KiB = 1024 B
Megabyte (MB)	1 MB = 1000 KB	Mebibyte (MiB)	1 MiB = 1024 KiB
Gigabyte (GB)	1 GB = 1000 MB	Gibibyte (GiB)	1 GiB = 1024 MiB
Terabyte (TB)	1 TB = 1000 GB	Tebibyte (TiB)	1 TiB = 1024 GiB
Petabyte (PB)	1 PB = 1000 TB	Pebibyte (PiB)	1 PiB = 1024 TiB
Exabyte (EB)	1 EB = 1000 PB	Exbibyte (EiB)	1 EiB = 1024 PiB

Valores de cada unidad de información según la norma ISO

Exíste otro estandard, que es el **JEDEC Standard 100B.01**. En éste estandard se usan sólo agrupaciones de 1024 bytes, y en éste caso decimos que **un kilobyte equivalen a 1024 bytes**. **Este es el estandard que se utiliza en Microsoft Windows, y es por lo que se genera tantísima confusión con las unidades de información**.

AGRUPACIONES DE BYTES (JEDEC)	
Byte (B)	1
Kilobyte (KB)	1 KB = 1024 B
Megabyte (MB)	1 MB = 1024 KB
Gigabyte (GB)	1 GB = 1024 MB
Terabyte (TB)	1 TB = 1024 GB
Petabyte (PB)	1 PB = 1024 TB
Exabyte (EB)	1 EB = 1024 PB

Valores de cada unidad de información según la norma JEDEC

Para simplificar, quiero dejar claro que en éste blog sigo la norma ISO, y no la JEDEC. Así que todas las unidades que veas de ahora en adelante seguirán la norma ISO.

En informática, en lo que al software se refiere al menos, **por norma general usamos agrupaciones de 1024 bytes**.

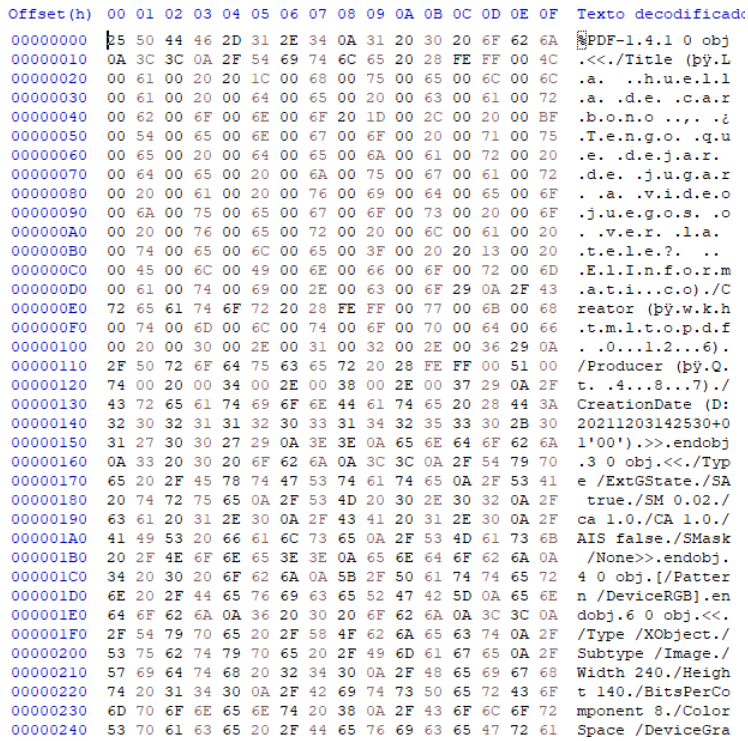
¿Por qué usar agrupaciones de 1024 bytes?

La respuesta es sencilla. **Dado que los sistemas digitales están basados en el sistema binario, las unidades de memoria se representan en potencias de**

2. Y 1024 es una potencia de 2 (2 elevado a 10).

Sin embargo, cuando no hablamos de unidades de memoria, podemos usar unidades de 1000 bytes, siguiendo el sistema internacional. Por éste motivo, la norma ISO contempla las dos posibilidades. Una siguiendo la nomenclatura estandard para las unidades de 1000 (kilo, mega, giga...), y la otra creando otra nomenclatura para las de 1024 (kibi, mebi, gibi...).

Generalmente cuando hablamos de memoria, siempre usamos el byte como unidad base para las memorias, y cada "celda" de memoria es 1 byte, siendo 16 bytes una línea de memoria (una vez más, en potencias de 2).



Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Texto decodificad
00000000	25	50	44	46	2D	31	2E	34	0A	31	20	30	20	6F	62	6A	%PDF-1.4.1 0 obj
00000010	0A	3C	3C	0A	2F	54	69	74	6C	65	20	28	FE	FF	00	4C	.<<./Title (þÿ.L
00000020	00	61	00	20	20	1C	00	68	00	75	00	65	00	6C	00	6C	.a. ..h.u.e.l.l
00000030	00	61	00	20	00	64	00	65	00	20	00	63	00	61	00	72	.a. .d.e. .c.a.r
00000040	00	62	00	6F	00	6E	00	6F	20	1D	00	2C	00	20	00	BF	.b.o.n.o ..,. .¿
00000050	00	54	00	65	00	6E	00	67	00	6F	00	20	00	71	00	75	.T.e.n.g.o. .q.u
00000060	00	65	00	20	00	64	00	65	00	6A	00	61	00	72	00	20	.e. .d.e.j.a.r.
00000070	00	64	00	65	00	20	00	6A	00	75	00	67	00	61	00	72	.d.e. .j.u.g.a.r
00000080	00	20	00	61	00	20	00	76	00	69	00	64	00	65	00	6F	. .a. .v.i.d.e.o
00000090	00	6A	00	75	00	65	00	67	00	6F	00	73	00	20	00	6F	.j.u.e.g.o.s. .o
000000A0	00	20	00	76	00	65	00	72	00	20	00	6C	00	61	00	20	. .v.e.r. .l.a.
000000B0	00	74	00	65	00	6C	00	65	00	3F	00	20	20	13	00	20	.t.e.l.e.?. ..
000000C0	00	45	00	6C	00	49	00	6E	00	66	00	6F	00	72	00	6D	.E.l.I.n.f.o.r.m
000000D0	00	61	00	74	00	69	00	2E	00	63	00	6F	29	0A	2F	43	.a.t.i...c.o)./C
000000E0	72	65	61	74	6F	72	20	28	FE	FF	00	77	00	6B	00	68	reator (þÿ.w.k.h
000000F0	00	74	00	6D	00	6C	00	74	00	6F	00	70	00	64	00	66	.t.m.l.t.o.p.d.f
00000100	00	20	00	30	00	2E	00	31	00	32	00	2E	00	36	29	0A	. .0...1.2...6).
00000110	2F	50	72	6F	64	75	63	65	72	20	28	FE	FF	00	51	00	/Producer (þÿ.Q.
00000120	74	00	20	00	34	00	2E	00	38	00	2E	00	37	29	0A	2F	t. .4...8...7)./
00000130	43	72	65	61	74	69	6F	6E	44	61	74	65	20	28	44	3A	CreationDate (D:
00000140	32	30	32	31	31	32	30	33	31	34	32	35	33	30	2B	30	20211203142530+0
00000150	31	27	30	30	27	29	0A	3E	3E	0A	65	6E	64	6F	62	6A	1'00').>>.endobj
00000160	0A	33	20	30	20	6F	62	6A	0A	3C	3C	0A	2F	54	79	70	.3 0 obj.<<./Typ
00000170	65	20	2F	45	78	74	47	53	74	61	74	65	0A	2F	53	41	e /ExtGState./SA
00000180	20	74	72	75	65	0A	2F	53	4D	20	30	2E	30	32	0A	2F	true./SM 0.02./
00000190	63	61	20	31	2E	30	0A	2F	43	41	20	31	2E	30	0A	2F	ca 1.0./CA 1.0./
000001A0	41	49	53	20	66	61	6C	73	65	0A	2F	53	4D	61	73	6B	AIS false./SMask
000001B0	20	2F	4E	6F	6E	65	3E	3E	0A	65	6E	64	6F	62	6A	0A	/None>>.endobj.
000001C0	34	20	30	20	6F	62	6A	0A	5B	2F	50	61	74	74	65	72	4 0 obj.[/Patter
000001D0	6E	20	2F	44	65	76	69	63	65	52	47	42	5D	0A	65	6E	n /DeviceRGB].en
000001E0	64	6F	62	6A	0A	36	20	30	20	6F	62	6A	0A	3C	3C	0A	dobj.6 0 obj.<<.
000001F0	2F	54	79	70	65	20	2F	58	4F	62	6A	65	63	74	0A	2F	/Type /XObject./
00000200	53	75	62	74	79	70	65	20	2F	49	6D	61	67	65	0A	2F	Subtype /Image./
00000210	57	69	64	74	68	20	32	34	30	0A	2F	48	65	69	67	68	Width 240./Heigh
00000220	74	20	31	34	30	0A	2F	42	69	74	73	50	65	72	43	6F	t 140./BitsPerCo
00000230	6D	70	6F	6E	65	6E	74	20	38	0A	2F	43	6F	6C	6F	72	mponent 8./Color
00000240	53	70	61	63	65	20	2F	44	65	76	69	63	65	47	72	61	Space /DeviceGra

Imágen de un editor hexadecimal con la información de la memoria codificada, mostrando cada byte en sistema hexadecimal. Hablaré del sistema hexadecimal en otro artículo.

Interpretando la información

Ahora ya conocemos lo que es un sistema digital y cómo funciona la información digital. Pero hasta ahora, sólo hemos hablado de números. Sin embargo, cuando estás leyendo éste artículo, en la pantalla de tu dispositivo movil, tablet, ordenador, o desde un dispositivo lector de texto, estarás recibiendo un **texto**. Y un texto no son sólo números. No sólo hay texto, también hay imágenes, datos de audio, de video... **¿Cómo traducimos esos ceros y unos en información tangible?**

La explicación es un poco compleja, y es que **todo depende de la manera en la que se interpreten**. Y es que el mundo digital es un mundo **abstracto**. Todo lo que ves en la pantalla, lo que escuchas, y lo que sientas en ellos, es en realidad un montón de números interpretados de una manera muy especifica.

Por ejemplo, a la hora de renderizar texto, se usa lo que se denomina una **codificación de caracteres**.

Una **codificación de caracteres consiste en una tabla de caracteres, donde cada valor numérico (code points) se transforma en un símbolo, siguiendo una tabla (code page)**. Principalmente exísten dos codificaciones estandard: la ASCII (de 8 bits, o 1 byte) y la UNICODE (de 16 bits, o 2 bytes).

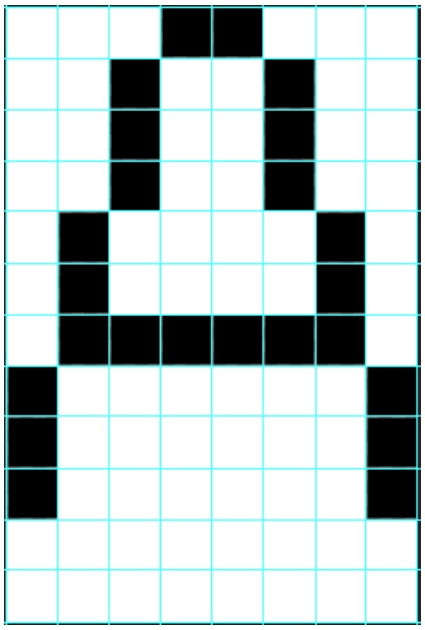
ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Tabla ASCII. Fuente: Wikimedia commons

Siguiendo ésta tabla, es muy sencillo de codificar cualquier texto en un texto digital. La palabra “EJEMPLO” en mayúsculas quedaría codificada con los valores, 69-74-69-77-80-76-79 respectivamente (en decimal).

¿Y cómo se codifica la imágen de cada letra (caracter) del texto? Muy sencillo. En el ámbito de los gráficos generados por ordenador, la unidad base es el **pixel**. Las imágenes están compuestas por “tablas”, siendo cada “celda” un pixel. La información contenida en cada píxel varía en función del formato. En el caso de una fuente de texto, podemos usar una imágen monocroma, de por ejemplo imágenes de 8 x 12 pixels (8 de largo y 12 de alto). Cada pixel podría ser 1 bit, siendo cada 1 un pixel negro, y cada 0 un pixel en blanco.



Letra A digitalizada en una imágen de 8×12 pixels. Cada cuadro representa un píxel, y cada pixel representa un bit de información. De modo que cada línea de la cuadrícula (llamada "mapa de bits") representa 1 byte.

Siguiendo éste formato, la letra A de la imágen quedaría codificada con los siguientes valores en la memoria (siendo cada valor un único byte): 24-36-36-36-66-66-126-129-129-129-0-0.

FILA	BINARIO	DECIMAL
1	00011000	24
2	00100100	36
3	00100100	36
4	00100100	36
5	01000010	66
6	01000010	66
7	01111110	126
8	10000001	129
9	10000001	129
10	10000001	129
11	00000000	0
12	00000000	0

Representación binaria y decimal de cada pixel de la imágen mostrada anteriormente.

Cuando la rutina para mostrar texto en la pantalla encuentre un byte con valor

decimal 65, buscará en la tabla ASCII (posición 65) la dirección en la memoria donde se encuentre el gráfico de la letra A, y dibujar en la pantalla la letra en base a la información contenida en el mapa de bits representado anteriormente, donde cada bit es un pixel blanco (0) o negro (1). Y así sucesivamente con el resto de caracteres del texto.

Share on Facebook Tweet Follow us

ANTERIOR

Desinformación, Bulos, Fake News y manipulación de la información

Buscar ...

Entradas Recientes

- Bits y bytes. Cómo funciona la información digital.
- Desinformación, Bulos, Fake News y manipulación de la información
- Cómo reinstalar o bajar de versión los drivers de la tarjeta gráfica con DDU en Windows 10 y 11
- La “huella de carbono”, ¿Tengo que dejar de jugar a videojuegos o ver la tele?
- ¿Qué es un NFT (Non Fungible Token)?
- El Metaverso: Nada nuevo en el horizonte

Categorías

Actualidad Android Básicos Ciberseguridad Clima Criptografía Emulación / Virtualización FOSS Hacking Hardware Informática Internet Juegos Opinion Otros Personal Privacidad Programación Tecnología Time Machine Tutoriales

RSS

Subscribirse al feed RSS

Inicio
Catálogo
PDFs
Política de privacidad
Política de Cookies
Acerca de mi
Acerca de ElInformati.co