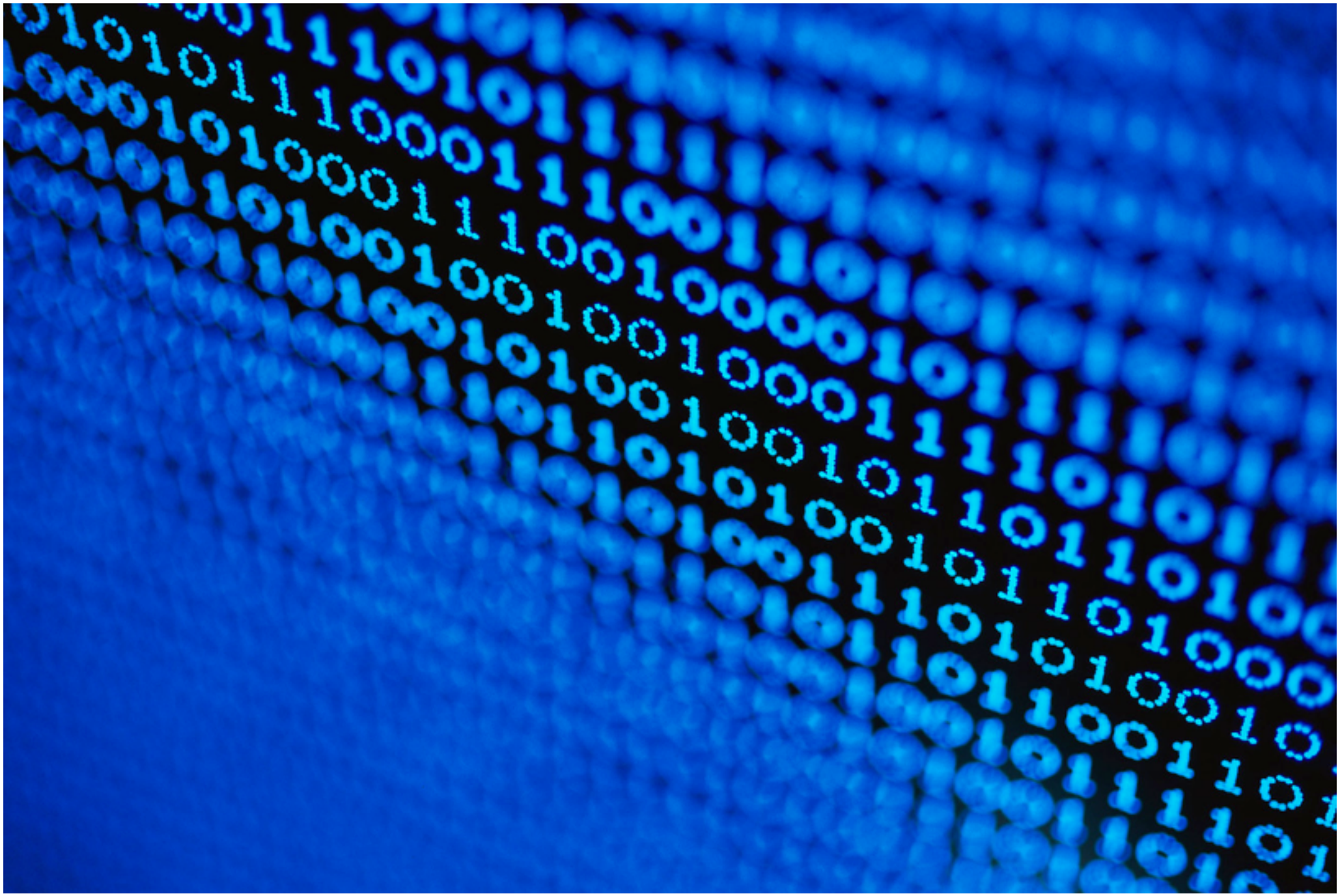
Format des données



GIF-1001 Ordinateurs : Structure et Applications, H2016

Jean-François Lalonde

Aujourd'hui

- Format des données
  - nombres entiers (positifs & négatifs)
  - chaînes de caractères
  - nombres rationnels
- Disponibilité
  - N'oubliez pas le doodle!
  - http://doodle.com/poll/ga8nphwfv68xfrag

Point hyper important à retenir™ #1

- Dans un ordinateur, *tout, absolument tout*, est stocké en format binaire
    - nombres entiers positifs, négatifs, nombre à virgule flottante, caractères, symboles, instructions, programmes, adresses, etc. etc. etc.
- Un caractère, en binaire, et nommé “bit”.
- Un bit peut avoir la valeur 0 ou 1.

Point hyper important à retenir™ #2

- On utilise un nombre *fini* et *pré-déterminé* de bits pour représenter de l'information.
  - 8 bits forment un octet (byte ou char) pouvant représenter $2^8 = 256$ valeurs.
  - 1 Kilo-octet (1 ko) = 1024 octets ou 8192 bits.
  - 1 Mega-octet (1 Mo) = 1024 Ko, 1 048 576 octets, ou …
  - Les mots « short, integer et long » expriment une valeur entière contenue sur un nombre croissant d'octets (1,2,4 ou 2,4,8 ou ...).

Nombres entiers positifs et bases

- Bases d'importance:
  - décimal (base 10): $255_d = 2x10^2 + 5x10^1 + 5x10^0$
  - binaire (base 2): $11111111_b = 1x2^7 + 1x2^6 + \ldots + 1x2^0$
  - hexadécimal (base 16):
    - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
    - $FF_h = Fx16^1 + Fx16^0$

De décimal vers les autres bases

- $10_d = ?_b$

10	2		
-10	5	2	
0	-4	2	2
	1	-2	**1**
		0	

- $9d = 1010_b$

De décimal vers les autres bases

- $147_d = ?_2$

147	2						
-146	73	2					
1	-72	36	2				
	1	-36	18	2			
		0	-18	9	2		
			0	-8	4	2	
				1	-4	2	2
					0	-2	**1**
						0	

- $23147_d = ?_h$

23147	16		
-23136	1446	16	
11 (B)	-1440	90	16
	6	-80	**5**
		10 (A)	

Point hyper important à retenir™ #3

- L'hexadécimal est une *façon plus compacte* de représenter du binaire.
  - 1 "symbole" en hexadécimal = 4 bits.
  - exemples:
    - $0_h = 0000_b$
    - $2_h = 0010_b$
    - $9_h = 1001_b$
    - $A_h = 1010_b$
    - $F_h = 1111_b$
    - $029AF_h = 00000010100110101111_b$

Nombres entiers négatifs

- Représentation « signe et magnitude »
  - Le premier bit est le signe: 0 = positif, 1 = négatif
  - Le reste est la magnitude
  - Exemple: $1101_b = (-1) \times (4 + 1) = -5_d$
  - Combien de nombres peut-on représenter?

- Problèmes?
  - Représentation de 0?
    - $0_d = 0000_b = 1000_b$
  - Opérations arithmétiques:
    - $-3_d + 4_d = 1011_b + 0100_b = 1111_b = -7_d$!

Représentation “complément 2”

- Le premier bit est $-2^{(\text{nombre de bits-1})}$, le reste est additionné
- Exemple: $1101_b = ?$
  - $1101_b = -2^3_d + 5_d = -3_d$.
- Pour changer le signe d’un nombre, il faut:
  - le soustraire de 2^N (d’où le nom “complément 2”)
  - plus direct (et plus facile): inverser tous les bits et ajouter 1.
- Exemple: $3_d = 0011_b$. $-3_d = ?_b$
  - Si on prend le complément + 1, on obtient: $1100_b + 1_b = 1101_b$.
- Combien de nombres peut-on représenter?

Nombres entiers négatifs

- Problèmes de tout à l'heure?
  - Représentation de 0?
    - 0000_b (maintenant, $1000_b = -8_d$)
  - Opérations arithmétiques:
    - $-3_d + 4_d = 1101_b + 0100_b = 0001_b = 1_d$ (beaucoup mieux)
- En pratique, soustraction = addition avec complément 2

Débordement (“overflow”)

- Quels nombres peut-on représenter avec N bits en complément 2?
  - -2^{N-1} à $2^{N-1}-1$
- Qu’arrive-t-il si on additionne
  - $5_d + 4_d = ?$
  - on obtient -7_d!
- De façon similaire, $-5_d + -4_d = 7_d$
- Comment faire pour détecter un débordement?
  - le bit de signe change

Chaînes de caractères — ASCII

- **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange

- Table reliant un caractère d'imprimerie à une valeur de 00_h à FF_h

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

http://www.asciitable.com

Chaînes de caractères — ASCII

- Exemple: “Bonjour!” (sans les “”) en ASCII?
  - attention aux majuscules…
- Bonjour! = 42 6F 6E 6A 6F 75 72 21 (en hex)

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Point hyper important à retenir™ #4

- A priori, nous ne pouvons pas savoir ce qu'une chaîne binaire signifie.
  - Ex: que veut dire 0x416C6C6F (sur 32 bits)?
  - La bonne réponse est: ça dépend!

entier non-signé	1097624687
entier signé	278356550
caractères ASCII	Allo

- Il nous *faut* donc savoir quel format utiliser pour bien interpréter les données

Nombres rationnels

- Une possibilité (16 bits):
  - mettons la virgule au milieu:

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	,	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}
b_{16}	b_{15}	b_{14}	b_{13}	b_{12}	b_{11}	b_{10}	b_9	,	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

    - 8 premiers bits: 2^0 à 2^7 (0 à 255)
    - 8 derniers bits: 2^{-1} à 2^{-8} (1/256 à 255/256, 0.00390625 à 0.99609375)

- Problèmes?
  - très limité! valeur maximale = 255, précision = 1/256

Nombres rationnels, en décimal

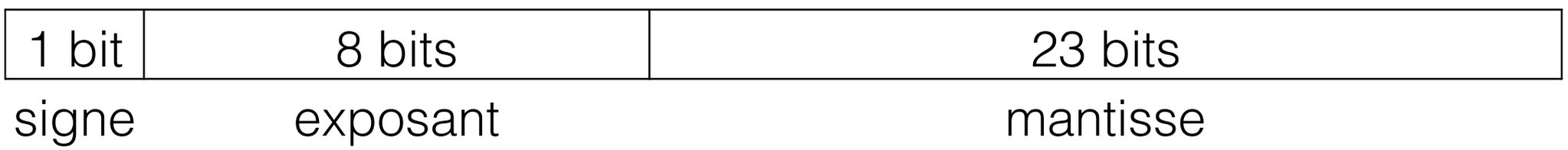
- Rappelez vous la notation scientifique (en décimal)
  - 6500 = (+) 6,5x10^3 (en décimale), ou 6,5E3
- Composantes:
  - signe (+): indique si le nombre est positif ou négatif
  - base (10): décimale
  - exposant (3): indique l'ordre de grandeur
  - mantisse (6,5): détermine la précision de la valeur représentée

Nombres rationnels, en binaire

- La norme IEEE 754 a été adoptée universellement (2008) pour les fractions sur 32, 64, et 128 bits

- Très similaire à la notation scientifique:
  - (signe) 1,mantisse x 2^{exposant}

- Par exemple, sur 32 bits (simple précision):
  - signe: un bit
  - base: 2, donc binaire. Comme cette base est toujours 2, on n'a pas besoin de la stocker (c'est implicite)
  - exposant (décalé): 8 bits (donc de 0 à 255), mais on soustrait 127, donc de -127 à +127
  - mantisse: 23 bits



Nombres rationnels (ou virgule flottante—“floating point”)

- (signe) 1,mantisse x $2^{(\text{exposant-127})}$
- Quel nombre décimal est représenté par:
  - 01000000011010000…?
- Sur 64 bits (double précision)?
  - Plus de bits pour l’exposant (11) et la mantisse (52)

Outil pratique

http://www.binaryconvert.com

Point hyper important à retenir™ #4

- A priori, nous ne pouvons pas savoir ce qu'une chaîne binaire signifie.
  - Ex: que veut dire 0x416C6C6F (sur 32 bits)?
  - La bonne réponse est: ça dépend!

entier non-signé	1097624687
entier signé	278356550
rationnel	14.47764
caractères ASCII	Allo

  - Il nous *faut* donc savoir quel format utiliser pour bien interpréter les données

Chaînes de caractères — Unicode

- Table reliant un caractère d'imprimerie à une valeur entre 0000_h et $FFFF_h$ (65536_d)

- En fait, il existe plusieurs (16) de ces tables

- Couvre même
  - l'écriture cunéiforme
  - hiéroglyphes
  - cartes à jouer