Les bases du codage binaire et hexadécimal

1) Rappels sur le codage Décimal des nombres.

Les humains comptent sur leurs doigts en '**base 10' décimale**.

Les nombres sont composés de plusieurs chiffres qui vont de **0 à 9**.
Exemples :
9 est un nombre composé d'un seul chiffre
123 est un nombre composé de trois chiffres

Prenons comme autre exemple le nombre **1967**.
Ce nombre décimal peut se décomposer ainsi :

7 x 10^0 = 7 *(10 puissance 0 = 1) La puissance 0 d'un nombre est toujours égal à 1.*
6 x 10^1 = 60 *(10 puissance 1 = 10) La puissance 1 d'un nombre est toujours égal à lui-même.*
9 x 10^2 = 900 *(10 puissance 2 = 10 x 10 = 100)*
1 x 10^3 = 1000 *(10 puissance 3 = 10 x 10 x 10 = 1000)*

L'ensemble est bien égal à : *1000 + 900 + 60 + 7 soit 1967.*

Les nombres sont toujours construits ainsi quel que soit leur "base", décimale binaire hexa. ou autre.

2) Quelques bases de Binaire.

Les machines elles n'utilisent que du **binaire** (**0 ou 1**)
Comme pour les nombres en décimal,
ceux en binaire sont composés de plusieurs chiffres.

Les chiffres de base en binaires s'appellent des **bits** ou des digits
On les réunit en ensembles de **8 bits** appelés **octets** (ou bytes avec un Y en anglais)
Avec **un octet** on peut coder les nombres allant de **0 à 255**

Le **bit** le **plus à droite** d'un nombre binaire est appelé : '***bit de poids faible***',
Le **bit** le **plus à gauche** d'un nombre binaire est dit : '***bit de poids fort'***

Prenons comme exemple l'octet **1100 0111**.
Ce nombre se convertit en décimal en commençant par le bit de poids faible, celui le plus à droite :

1 x 2^0 = 1 *(2 puissance 0 = 1) La puissance 0 d'un nombre est toujours égal à 1.*
1 x 2^1 = 2 *(2 puissance 1 = 2) La puissance 1 d'un nombre est toujours égal à lui-même.*
1 x 2^2 = 4 *(2 puissance 2 = 2 x 2 = 4)*
0 x 2^3 = 0 *(2 puissance 3 = 2 x 2 x 2 = 8)*

0 x 2^4 = 0 *(2 puissance 4 = 2 x 2 x 2 x 2 = 16)*
0 x 2^5 = 0 *(2 puissance 5 = 2 x 2 x 2 x 2 x 2 = 32)*
1 x 2^6 = 64 *(2 puissance 6 = 2 x 2 x 2 x 2 x 2 x 2 = 64)*
1 x 2^7 = 128 *(2 puissance 7 = 2 x 2 x 2 x 2 x 2 x 2 x 2= 128)*

L'ensemble est égal à *(128 + 64 + 0 + 0) + (0 + 4 + 2 + 1)* = **199** en décimal

On peut aussi faire un **tableau** pour faciliter la conversion :

Valeur de chaque bit à 1	128	64	32	16	8	4	2	1
Nombre binaire	1	1	0	0	0	1	1	1

Il suffit d'additionner toutes les valeurs en décimal pour les bits à 1
et on obtient la valeur en décimal du nombre binaire…toujours **199.**

3) ***Quelques bases d'Hexadécimal.***

Pour faciliter la vie des programmeurs et utiliser une notation plus courte que le binaire, tout en restant facile à convertir, on utilise l'**hexadécimal**.

En **hexadécimal** chaque chiffre vas de **0 à F** ,
A étant l'équivalent de **10** ; et **F** étant l'équivalent de **15** en décimal classique.

Chaque chiffre peut donc être : **0 ; 1 ; 2 ; 3 ; 4 ; 5 ; 6 ; 7 ; 8 ; 9 ; A ; B ; C ; D ; E ; F**.

L'avantage est que l'on peut représenter **un octet** avec **deux chiffres en hexadécimal**
FF en hexa est égal à **1111 1111 en binaire**

Si nous reprenons notre exemple d'octet en binaire précèdent : **1100 0111**
En hexadécimal nous prenons **chaque bloc de 4 bits** pour le convertir séparément.

Commençons par **0111** :

1 x 2^0 = 1 *(2 puissance 0 = 1) La puissance 0 d'un nombre est toujours égal à 1.*
1 x 2^1 = 2 *(2 puissance 1 = 2) La puissance 1 d'un nombre est toujours égal à lui-même*.
1 x 2^2 = 4 (*2 puissance 2 = 2 x 2 = 4)*
0 x 2^3 = 0 *(2 puissance 3 = 2 x 2 x 2 = 8)*

Ça donne (*1+2+4+0)* = **7...** qui s'écrit évidemment en décimal comme en hexadécimal.

Prenons le deuxième bloc **1100** comme si c'était un nombre séparé

0 x 2^0 = 0 *(2 puissance 0 = 1) La puissance 0 d'un nombre est toujours égal à 1.*
0 x 2^1 = 0 *(2 puissance 1 = 2) La puissance 1 d'un nombre est toujours égal à lui-même.*
1 x 2^2 = 4 (*2 puissance 2 = 2 x 2 = 4)*
1 x 2^3 = 8 (*2 puissance 3 = 2 x 2 x 2 = 8*)

Ça donne (*0+0+4+8)* = 12... qui s'écrit **C** en **hexadécimal**

Notre nombre binaire **1100 0111** s'écrit 197 en décimal et **C7** en **hexadécimal**.
Une notation bien plus courte donc.

Les **adresses Mac** des cartes réseau sont toujours notées en **hexadécimal**,
avec une suite de **6 octets.**
Ca permet une notation courte et de les distinguer facilement des adresses **IP V4**
qui sont une suite de **4 octets** présentés en **décimal.**

En voici un exemple d'adresse **Mac** :
08-2E-5F-2C-00-19 *(affichage système **Windows** avec ' **–** 'de séparation)*
ou **08:2E:5F:2C:00:19**

4) ***Pour aller plus loin.***

Dans Windows 10, la calculatrice en mode 'Programmeur,' permet la conversion Décimale/Hexa./Binaire

Si vous voulez aller plus loin ou si la **vidéo** vous inspire plus que la lecture.
Je vous recommande cette excellente vidéo **YouTube** qui reprend ce que je présente ici et plus encore :
https://www.youtube.com/watch?v=pPK_C7DKyJo