Codage de l'information

(entier/décimal/réel/caractères)

Rq : Pour ce TP vous aurez besoin de Visual Studio Code avec l'extension « Hex Editor »

Exercice 1 : Conversion – changement de base en Javascript

Deux fonctions utiles :

- (nb).toString(base) : convertit le nombre *nb* en une chaine de caractère représentant ce nombre dans une *base* donnée :
  exemples :
  (43).toString(2) = "101011" ➔changement de base 10 → 2
  (0x43).toString(2) = "1000011" ➔changement de base 16 → 2
  (0x43).toString(10) = "67" ➔changement de base 16 → 10
  !! Attention : ne fonctionne pas pour les nb négatifs, ne calcule pas le complément à deux

- parseInt("chaine", base) : convertit une *chaine* de caractères représentant un nombre dans une *base* donnée en un nombre entier en base 10.
  exemples :
  parseInt("1000",2) → 8

Convertir en binaire et en hexadécimal
a) $(3167)_{10}$
b) $(219)_{10}$
c) $(6560)_{10}$

Convertir en binaire et en décimal
a) $(3AE)_{16}$
b) $(FFF)_{16}$
c) $(6AF)_{16}$

Convertir en décimal et en hexadécimal
a) $(101)_2$
b) $(01011)_2$
c) $(11011101)_2$

Exercice 2 : représentation des nombres en Javascript

Les nombres en JS sont représentés sur 64 bits.

Si on considère les entiers relatifs le plus grand entier devrait être :
0111 1111 1111 1111 … 1111 = 2^{63}-1
Donc les calculs avec des nombres < 2^{63} devraient être corrects, et pourtant…

Calculer : n = Math.pow(2,62)
Afficher n :
Calculer : n = n + 1

Afficher n :
Calculer « n==n+1 »

Pourquoi de tels résultats ?

En fait, JS ne fait pas la différence entre nb entier et nb réel, tous les nombres sont codés en IEEE-764 sur 64 bits.

La doc dit que l'entier le plus grand géré de manière sure est 2^{53}-1 ; pourquoi ? l'explication se trouver du coté de la norme IEEE-764 … expliquer ?
Faisons maintenant quelques calculs sur des nbs réels :

Calculer n=0.1 + 0.2
Evaluer n== 0.3 ou encore 0.3 == 0.1 + 0.2
Pourquoi le résultat est false ?

Coder 0.1 en binaire avec une des 2 fonctions JS précédentes ; remarque ?
Coder 0.2 aussi binaire ; remarque ?
Conclusion ?

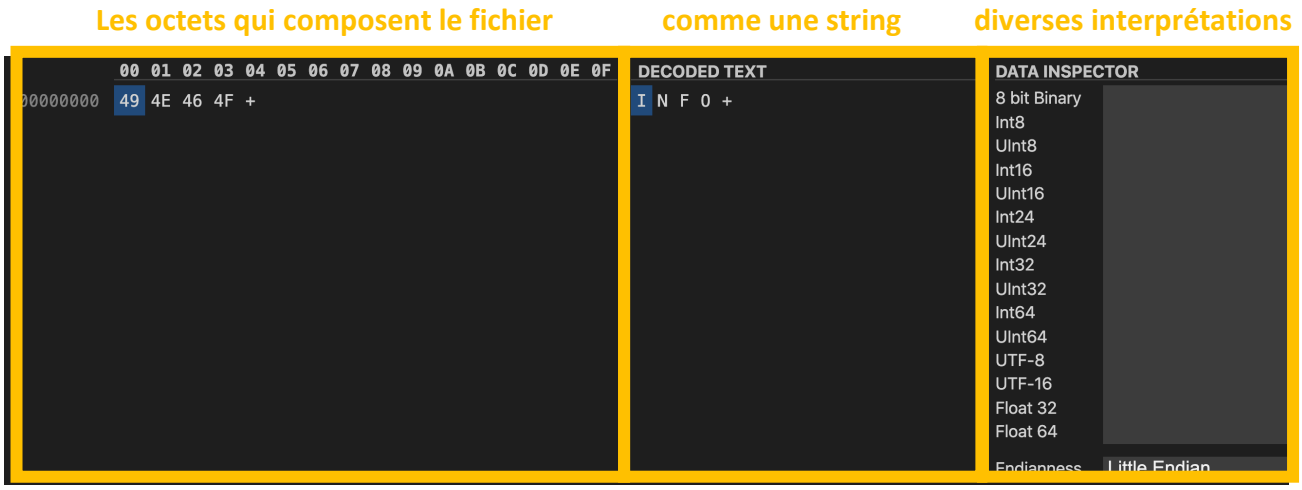
Exercice 3 : interprétation d'une chaine binaire

Considérons la chaîne de 32 bits : **0100 1001 0100 1110 0100 0110 0100 1111**
Qu'obtient-on si elle représente (détailler vos calculs) :

- 4 entiers sur 8 bits :
- 4 caractères :
- 2 entiers sur 16 bits :
- 1 réel sur 32 bits :

Pour vérifier vos calculs, avec VSC créer le fichier texte « info.txt » contenant le texte INFO ;

Ouvrir ce fichier avec l'extension « Hex editor » ; Il y a 3 colonnes :



En déplaçant le curseur dans la zone 1, vous devriez voir des valeurs dans « data inspector » et ainsi retrouver/vérifier vos calculs

Exercice 3 : représentation des caractères

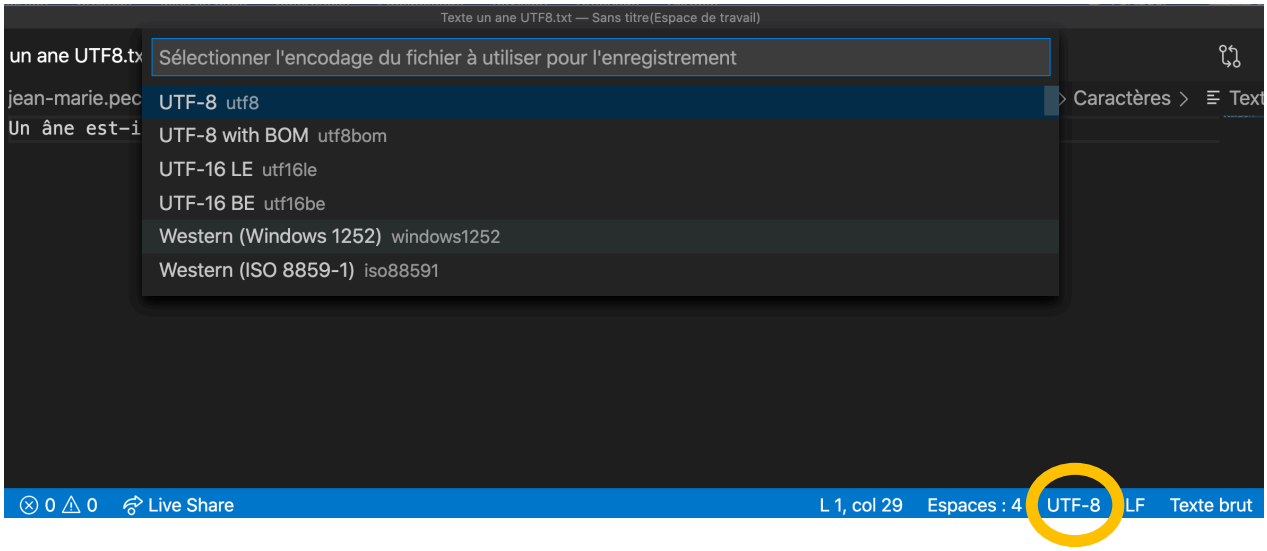
Soit la chaîne de caractères : **Un ane est-il passe par la ?**

Si elle est codée en ASCII, combien d'octets occupe-t-elle ? Donner la suite de codes en hexa.
Si elle est codée en UTF8, combien d'octets occupe-t-elle ?
https://www.utf8-chartable.de/

Soit la chaîne de caractères : **Un âne est-il passé par là ?**

Si elle est codée en ASCII Etendu, combien d'octets occupe-t-elle ?
Si elle est codée en Windows1252, combien d'octets occupe-t-elle ?
Si elle est codée en UTF32, combien d'octets occupe-t-elle ?
Si elle est codée en UTF8, combien d'octets occupe-t-elle ?
Si elle est codée en « UTF8 avec BOM », combien d'octets occupe-t-elle ?
(rechercher ce qu'est le BOM et l'influence sur la taille du fichier ; quel est son « code » ?)

Vérifier tous ces calculs en créant un fichier différent contenant ce texte avec un des codages.
Par défaut dans VSC le texte est codé en UTF8 ; pour changer le codage cliquer sur le codage indiqué dans la barre d'état puis choisir le nouveau codage.



Exercice 4 : fichier texte ou fichier traitement de texte ?

Nous venons de manipuler des fichiers textes avec un editeur de textes, normal donc ; mais quel est le format d'un fichier produit par un traitement de textes ?

Avec Word (ou équivalent), créer deux fichiers, un au format « .doc » et un au format « .docx » contenant le texte « Un âne est-il passé par là ? »
A priori on ne pourra modifier ces fichiers qu'avec Word (ou équivalent) et pourtant …

Essayer d'ouvrir le fichier « .doc » avec VSC ; il indique que c'est un fichier binaire (donc pas un fichier texte) mais il vous laisse la possibilité de l'ouvrir ; ouvrez le avec « hex editor » ;
Rien de vraiment clair à lire….

Mais vous devriez trouver dans la colonne « decode text » le texte saisi.
Caractère par caractère, remplacer le texte « Un âne est-il passé par là « par « Une vache est passée par la ».
Enregistrer puis rouvrir le fichier avec Word ; résultat ?

Essayer d'ouvrir le fichier « .docx » avec VSC ; il indique aussi que c'est un fichier binaire ; ouvrez le quand même avec « hex editor » ;
Mais cette fois rien de lisible, impossible de retrouver le texte initial…
En fait les fichiers « docx » sont des archives ;
Renommer le fichier « .docx » en « .zip » puis décompresser le fichier ; on voit alors qu'il contient plusieurs dossiers et plusieurs fichiers ; repérer le fichier « document.xml » ; puis chercher le texte « âne » et remplacer le par « vache ». Compresser à nouveau tous les docs dans un nouveau fichier « .zip » que vous renommerez en « .docx » ; puis ouvrez le avec Word ; normalement vous devriez voir le texte modifié.
Conclusion ?