

```
// cette fonction met la première lettre de chaque mot en majuscule
// (si c'est une lettre minuscule)
// et toutes les autres lettres du mot en minuscule
// (si ce sont des lettres majuscules)
// un mot est une chaîne de caractères alphanumériques
// (si la première lettre d'un mot est un caractère numérique, elle sera donc
// traitée comme la première lettre d'un mot, mais ne sera pas mise en majuscule)
```

```
// l'unique paramètre de cette fonction est la chaîne de caractères (str)
```

```
char    *ft_strcapitalize(char *str)
{
```

```
    // on déclare une variable i de type int
    // puis on l'initialise à 0
    // elle servira d'indice pour parcourir un à un
    // les caractères de la chaîne str
    int    i;
```

```
    // on déclare une variable is_first_letter de type int
    // puis on l'initialise à 1
    // elle permettra d'indiquer si le caractère courant
    // est potentiellement le premier caractère d'un mot
    // (1 : potentiellement le 1er caractère d'un mot)
    // ou non
    // (0 : pas le 1er caractère d'un mot)
    // on appelle ce genre de variable un flag
    int    is_first_letter;
```

```
    i = 0;
    is_first_letter = 1;
```

```
    // tant que la chaîne de caractère n'est pas terminée
    // (tant que le caractère de fin de chaîne n'a pas encore été atteint)
    // (ou si la chaîne de caractères n'est pas vide !)
```

```
    while (str[i] != '\0')
    {
```

```
        // si le caractère en cours est une lettre minuscule
```

```

// (entre 'a' et 'z')
if (str[i] >= 'a' && str[i] <= 'z')
{
    // et que ce caractère est potentiellement
    // la première lettre d'un mot
    if (is_first_letter)
    {
        // il est converti en majuscule
        // en soustrayant 32 à sa valeur ASCII
        // (voir ft_strupcase)
        str[i] -= 32;

        // le flag is_first_letter est mis à 0
        // car la première lettre du mot a été traitée
        // et que le caractère suivant ne sera pas
        // la première lettre d'un mot
        is_first_letter = 0;
    }
}
// sinon (si le caractère en cours n'était pas une minuscule
// et la première lettre d'un mot)
// si c'est une lettre majuscule
// et que ce n'est pas la première lettre d'un mot
// elle est convertie en minuscule
// en ajoutant 32 à sa valeur ASCII
// (voir ft_lowercase)

// ATTENTION :
// il faut bien indiquer else
// car sinon, si la lettre précédente était une lettre minuscule
// et la première lettre d'un mot
// (si on est passé dans le if précédent)
// et donc qu'elle a été convertie en majuscule
// et que le flag is_first_letter est passé à 0
// cette première lettre du mot
// sera à nouveau convertie en minuscule !

// avec else, on évite donc de passer dans ce if

```

```

// si on est déjà passé dans le premier if !

else if (str[i] >= 'A' && str[i] <= 'Z')
{
    if (!is_first_letter)
        str[i] += 32;

    // sinon (si c'est une lettre majuscule et le
    // premier caractère d'un mot)
    else
        // le flag is_first_letter est mis à 0
        // car la première lettre du mot a été traitée
        // (mais n'a pas été convertie en majuscule
        // car c'est déjà une majuscule !)
        // et que le caractère suivant ne sera pas
        // la première lettre d'un mot
        is_first_letter = 0;
}

// sinon (si le caractère en cours n'est ni une minuscule,
// ni une majuscule)
// si c'est un caractère numérique

// REMARQUE :
// on pourrait se passer de ce else, et écrire
// if (str[i] >= '0' && str[i] <= '9')
// la fonction se comporterait de la même manière
// mais serait moins efficace
// car, sans ce else, on vérifierai que le caractère
// est un chiffre même dans le cas
// ou il a déjà été identifié et traité
// comme un caractère minuscule ou majuscule !
// cela induirait une vérification supplémentaire
// dans la fonction alors que c'est inutile
// (car un caractère ne peut pas être simultanément
// une lettre minuscule, une lettre majuscule, et un chiffre)
else if (str[i] >= '0' && str[i] <= '9')

```

```

        // le flag is_first_letter est mis à 0
        // car la première lettre du mot a été traitée
        // (mais n'a pas été convertie en majuscule
        // car c'est un chiffre !)
        // et que le caractère suivant ne sera pas
        // la première lettre d'un mot
        is_first_letter = 0;

    // sinon (si le caractère en cours n'est ni une minuscule,
    // ni une majuscule, ni un chiffre)
    // donc si le caractère en cours est un caractère spécial
    // (ne faisant pas partie d'un mot)
    else
        // le flag is_first_letter est mis à 1
        // car ce n'est PAS la première lettre d'un mot
        // et que le caractère suivant sera POTENTIELLEMENT
        // la première lettre d'un mot
        is_first_letter = 1;

    // on incrémente i
    // pour passer au caractère suivant de la chaîne de caractères
    i++;
}

// une fois sorti de la boucle while
// (une fois le caractère de fin de chaîne de caractères '\0' trouvé)
// donc une fois la chaîne de caractères parcourue entièrement
// on retourne la chaîne modifiée
return (str);
}

#include "ft_strcpy.h"
#include "ft_putstr.h"
#include "ft_strcapitalize.h"
#include <unistd.h>

int    main(void)
{

```

```

// tableau de 62 caractères pour stocker la chaine de caractères
// + le caractère de fin de chaine de caractères
char    str[62];

// on utilise ft_strcpy pour copier la chaine de caractères
// en deuxième paramètre dans str (premier paramètre)
// cette fonction ajoute le caractère nul en fin de chaine

ft_strcpy(str, "salut, comment tu vas ? 42mots quarante-deux; cinquante+et+un");

// on affiche la chaine de caractères originale
ft_putstr(str);

// on saute une ligne
write(1, "\n", 1);

// on capitalise la chaine de caractères, et on l'affiche
ft_putstr(ft_strcapitalize(str));

// on saute une ligne
write(1, "\n", 1);
return (0);

```

```

}

```

```

// RESULTAT :
// salut, comment tu vas ? 42mots quarante-deux; cinquante+et+un
// Salut, Comment Tu Vas ? 42mots Quarante-Deux; Cinquante+Et+Un

```