

```
size_t strlcpy(char *dst, const char *src, size_t dstsize);
```

Copie `dstsize - 1` caractères de la chaîne `src` vers la chaîne `dst` et ensuite ajoute une terminaison NUL (`'\0'`) si `dstsize` n'est pas zéro.

La fonction prends la taille entière de la chaîne `dst` comme paramètre de taille. Elle ne peut pas remplir `dst` avec des `'\0'` sauf le `'\0'` de terminaison!

La fonction retourne la taille totale de la chaîne qu'elle essaie de créer - dans ce cas c'est simplement la taille de `src` (jusqu'à le premier `'\0'` rencontré, p.x. pour `src[16] = "Bonjour!"` `strlcpy = 8`).

```
char *src = "Bonjour, Mariya!";  
char dst[16];
```

```
strlcpy (dst, src, 16) retourne 16  dst = "Bonjour, Mariya"  
strlcpy (dst, src, 7)  retourne 16  dst = "Bonjou"  
strlcpy (dst, src, 0)  retourne 16  dst = "Bonjou"
```

Donc à chaque fois il y a bien `dstsize - 1` caractères copiés.

```
size_t strlcat(char *dst, const char *src, size_t dstsize);
```

La fonction concatène **au plus** `dstsize - strlen(dst) - 1` caractères de la chaîne `src` à la fin de la chaîne `dst`. Elle ajoute ensuite la terminaison `'\0'` sauf si `dstsize = 0` ou si la chaîne originale `dst` est plus longue que `dstsize`.

Si `src` et `dst` se chevauchent, le comportement est indéfini.

La fonction retourne la taille totale de chaîne qu'elle essaie de créer (`strlen(dst) + strlen(src)` ou `dstsize + strlen(src)`).

---

#### DETAILS:

---

`dstsize` est la taille maximale de la chaîne `dst` et `strlen(dst)` représente le nombre de caractères dans la chaîne `dst`.

p.ex.

```
dst[10] = "HELLO"
```

Alors:

```
dstsize = 10;
```

```
strlen(dst) = 5;
```

*Dans cet exemple, les éléments de `dst[5]` jusqu'à `dst[10]` sont remplis avec des `'\0'`;*

Donc la place disponible pour la concaténation est :

**`dstsize - strlen(dst) - 1`**

**Où « -1 » représente le dernier élément de la chaîne `dst` qui doit absolument être un `'\0'`. La fonction `strlcat` garantit cette terminaison!**

Dans l'exemple en haut, la fonction peut concaténer au maximum 4 caractères et ajoutera ensuite la terminaison nulle.

#### **- CAS `dstsize < strlen(dst)`**

Si nous sommes dans un cas où `dstsize < strlen(dst)` alors cela veut dire que la chaîne `dst` n'a pas suffisamment de place disponible pour accepter des caractères -> la concaténation n'est pas possible.

La fonction doit retourner la taille de la chaîne qu'elle essaie de créer, ce qui est **`dstsize + strlen(src)`**

- CAS `dstsize == strlen(dst)` et `dstsize == strlen(dst) + 1`

Dans ces cas, la concaténations n'est pas possible parce qu'il n'y a pas suffisamment de place disponible. Cependant la fonction retourne **`strlen(dst) + strlen(src)`**.

- CAS `dstsize > strlen(dst) + 1`

La condition pour que la concaténation ait lieu est satisfaite!

Prenons l'exemple suivant:

```
dst[10] = "HELLO";  
src[6] = "123456";  
dstsize = 10;  
strlen(dst) = 5;
```

Nous pourrions concaténer au maximum :  $dstsize - strlen(dst) - 1 = 4$  éléments de la chaîne src à la chaîne dst.

La chaîne dst deviendra "HELLO1234". La fonction retourne 11.

**La concaténation s'arrêtera lorsque `strlen(dst) == dstsize - 1` et ajoutera ensuite la terminaison nulle!**

**La fonction retournera `strlen(src) + strlen(dst)`;**