

## Affichages formatés à l'écran avec `printf`

Ce document propose un aperçu des possibilités fournies par l'instruction `printf` pour afficher des valeurs à l'écran. Après un rappel de la syntaxe du `printf`, nous verrons comment préciser la longueur et l'alignement des valeurs de différents types lors de leur affichage.

### Table des matières

<a href="#">L'instruction <code>printf</code></a>	2
<a href="#">Syntaxe générale</a>	2
<a href="#">Points d'injection</a>	2
<a href="#">Résultat de la fonction <code>printf()</code></a>	2
<a href="#">Typage des points d'injection</a>	3
<a href="#">Format des types principaux</a>	3
<a href="#">Format des autres types</a>	3
<a href="#">Formatage</a>	4
<a href="#">Les chaînes de caractères</a>	4
<a href="#">Chaîne de caractères alignée à droite</a>	4
<a href="#">Chaîne de caractères alignée à gauche</a>	4
<a href="#">Chaîne de caractères plus longue que la longueur indiquée</a>	5
<a href="#">Chaîne de caractères tronquée</a>	5
<a href="#">Les entiers</a>	6
<a href="#">Valeur entière formatée et alignée à gauche</a>	6
<a href="#">Valeur entière formatée et alignée à droite</a>	6
<a href="#">Affichage des zéros non significatifs</a>	6
<a href="#">Forcer l'affichage du signe</a>	6
<a href="#">Les nombres réels</a>	7
<a href="#">Longueur imposée avec nombre de décimales</a>	7
<a href="#">Longueur quelconque avec nombre de décimales</a>	7
<a href="#">Alignement de la valeur à gauche</a>	7
<a href="#">Affichage des zéros non significatifs</a>	8
<a href="#">Partie entière plus longue que la longueur souhaitée</a>	8
<a href="#">Affichage du signe</a>	8
<a href="#">Écriture scientifique d'un nombre réel</a>	8
<a href="#">Les caractères</a>	9
<a href="#">Format de base pour les caractères</a>	9
<a href="#">Caractères spéciaux</a>	9
<a href="#">Afficher les caractères réservés</a>	9
<a href="#">Afficher le code d'un caractère</a>	9
<a href="#">Afficher un caractère par son code</a>	9
<a href="#">Les booléens</a>	10
<a href="#">Afficher un booléen comme un entier</a>	10
<a href="#">Afficher un booléen comme une chaîne de caractères</a>	10
<a href="#">Les autres types de données</a>	10

## 1. L'instruction printf

### 1.1 Syntaxe générale

L'instruction `printf` est une fonction variadique, c'est-à-dire qu'elle admet un nombre variable de paramètres (au moins un). Le premier (et éventuellement unique) paramètre constitue le "message" à afficher à l'écran.

Exemple : `printf("Bonjour");`


### 1.2 Points d'injection

Dès lors que le message contient des éléments variables, il faut lui ajouter des "*points d'injection*" en indiquant le format d'affichage souhaité pour chaque élément. Ce format commence toujours par le caractère `%`.

Il faut bien sûr fournir, à la suite du message, autant de valeurs qu'il y a de points d'injection.

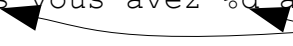
Exemple avec un point d'injection de type chaîne de caractères :

```
printf("Bonjour %s", nomDeLaPersonne);
```



Exemple avec deux points d'injection, le premier de type chaîne de caractères et le deuxième de type entier :

```
printf("Bonjour %s vous avez %d ans", prenom, age);
```



À l'exécution, la valeur de la variable `prenom` viendra remplacer le premier point d'injection et la valeur de la variable `age` remplacera le deuxième point d'injection.

### 1.3 Résultat de la fonction printf()

`printf()` est en fait une fonction qui retourne un entier correspondant à la longueur du message qui a été affiché. En cas d'erreur c'est la constante `EOF` qui sera retournée.

## 2. Typage des points d'injection

### 2.1 Format des types principaux

Les formats pour les points d'injection sont typés, le format à utiliser dépend du type de la valeur à afficher. Vous connaissez déjà les formats pour les types les plus courants :

entier	%d
réel	%f
caractère	%c
chaîne de caractères	%s

### 2.2 Format des autres types

Il existe des formats pour d'autres types de donnée :

entier long	long	%ld
réel long	double	%lf
entier non signé	unsigned int	%u
entier long non signé	unsigned long	%lu
adresse (pointeur)		%p
format hexadécimal (en minuscules)	int	%x
format hexadécimal (en majuscules)	int	%X

Quelques exemples :

```
int entier = 1966;  
printf("Valeur en hexa : %x", entier);
```

```
Valeur en hexa : 7ae
```

```
int entier = 1966;  
printf("Valeur en hexa : %X", entier);
```

```
Valeur en hexa : 7AE
```

```
int entier = 1966;  
printf("Valeur d'une adresse : %p", &entier);
```

```
Valeur d'une adresse : 0060FEFC
```

### 3. Formatage

Le formatage permet de préciser la longueur et l'alignement d'une valeur à l'affichage.

Toute valeur possède une longueur d'affichage par défaut. Par exemple l'entier 32 a une longueur d'affichage de 2 (à l'écran son affichage tiendra sur 2 caractères), la chaîne de caractères "BONHEUR" a une longueur de 7, la valeur 3.14 a une longueur de 4, etc.

Quand on affiche une valeur avec `printf`, il est possible d'exiger une autre longueur que la longueur d'affichage par défaut, c'est-à-dire d'indiquer le nombre exact de positions réservées sur l'écran pour l'affichage de la valeur. Toutefois si cette longueur est insuffisante, elle ne sera pas respectée.

Enfin, la plupart du temps il est possible d'indiquer si la valeur doit être alignée à gauche ou à droite. L'alignement par défaut est l'alignement à droite.

#### 3.1 Les chaînes de caractères

##### 3.1.1 Chaîne de caractères alignée à droite

La longueur souhaitée pour l'affichage est à indiquer entre le `'%'` et le `'s'`. Pour afficher une chaîne de caractères sur 10 caractères, on utilisera donc le format `%10s`. Par défaut la chaîne est alignée à **droite**.

```
printf("Chaines=%10s%10s%10s", "AMOUR", "GLOIRE", "BEAUTE");
```



Lors de l'affichage, la chaîne est comblée par des espaces.

##### 3.1.2 Chaîne de caractères alignée à gauche

Pour avoir un alignement à **gauche**, il faut ajouter le signe `-` dans le format.

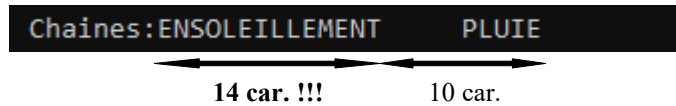
```
printf("Chaines=%-10s%-10s%-10s", "AMOUR", "GLOIRE",  
"BEAUTE");
```



### 3.1.3 Chaîne de caractères plus longue que la longueur indiquée

ATTENTION, si la chaîne à afficher est trop longue, la longueur indiquée dans le format n'est pas respectée.

```
printf("Chaines=%-10s%-10s", "ENSOLEILLEMENT", "PLUIE");
```



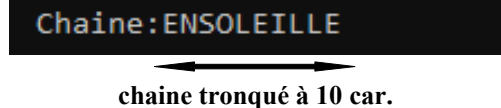
Chaines:ENSOLEILLEMENT PLUIE

14 car. !!! 10 car.

### 3.1.4 Chaîne de caractères tronquée

Pour éviter le problème précédent, il est possible de tronquer une chaîne de caractères.

```
printf("Chaines=%10.10s", "ENSOLEILLEMENT");
```



Chaîne:ENSOLEILLE

chaîne tronqué à 10 car.

## 3.2 Les entiers

### 3.2.1 Valeur entière formatée et alignée à gauche

Le principe utilisé précédemment pour les chaînes de caractères vaut aussi pour l'affichages de valeurs entières.

```
int entier1 = 57;
int entier2 = 24;
printf("Entier:%-4d%-4d", entier1,entier2);
```

```
Entier:57 24
```

### 3.2.2 Valeur entière formatée et alignée à droite

Affichage d'entiers sur 4 positions, alignées à droite :

```
int entier1 = 57;
int entier2 = 24;
printf("Entier:%4d%4d", entier1,entier2);
```

```
Entier: 57 24
```

### 3.2.3 Affichage des zéros non significatifs

Dans le cas d'un alignement à droite, il est possible de remplacer les espaces qui précèdent la valeur par des 0.

```
int entier1 = 57;
int entier2 = 24;
printf("Entier:%04d%04d", entier1,entier2);
```

```
Entier:00570024
```

### 3.2.4 Forcer l'affichage du signe

Un entier négatif s'affiche toujours avec son signe. Mais il est possible de forcer l'affichage du signe + pour les entiers positifs (ça peut être utile dans une application financière) : il faut ajouter le caractère + dans le format.

```
int entier1 = -57;
int entier2 = 24;
printf("Entier:%+4d\n", entier1);
printf("Entier:%+4d", entier2);
```

```
Entier: -57
```

```
Entier: +24
```

### 3.3 Les nombres réels

Un nombre réel comporte un nombre de chiffres avant le séparateur décimal, le séparateur décimal et un nombre de chiffres après le séparateur décimal. Tout cela rentre en ligne de compte pour sa longueur d’affichage. Par exemple le nombre 123.456 a une longueur d’affichage de 7 caractères.

Sans précision contraire dans le format d’affichage, une valeur réelle sera affichée avec 6 décimales (au besoin complété par des 0)

#### 3.3.1 Longueur imposée avec nombre de décimales

Dans le format d’affichage d’un nombre réel, il est possible d’indiquer la longueur totale souhaitée ainsi que le nombre de décimales à afficher.

```
float reel = 123.456;  
printf("Nombre:%10.2f", reel);
```



Nombre: 123.46

La valeur occupe 10 caractères à l’écran, dont 3 pour la partie entière, 1 pour le séparateur décimal et 2 pour les décimales ; elle est par défaut alignée à droite.

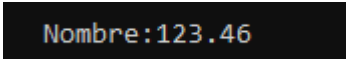
Notez que l’affichage de la valeur est complété par des espaces.

Notez enfin que la valeur a été arrondie car on lui demande d’afficher moins de décimales qu’elle n’en contient réellement.

#### 3.3.2 Longueur quelconque avec nombre de décimales

Il est possible d’omettre la longueur totale et de ne préciser que le nombre de décimales souhaité.

```
float reel = 123.456;  
printf("Nombre:%.2f", reel);
```

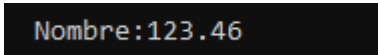


Nombre:123.46

#### 3.3.3 Alignement de la valeur à gauche

Comme pour les entiers on retrouve la possibilité d’aligner la valeur à gauche.

```
float reel = 123.456;  
printf("Nombre:%-10.2f", reel);
```



Nombre:123.46

### 3.3.4 Affichage des zéros non significatifs

Si la valeur ne possède pas autant de décimales que le nombre indiqué, des 0 non significatifs sont affichés.

```
float reel = 123.4;  
printf("Nombre:%-10.3f", reel);
```

Nombre:123.400

Et comme pour les entiers, on peut afficher des 0 non significatifs devant la valeur.

```
float reel = 123.456;  
printf("Nombre:%010.2f", reel);
```

Nombre:0000123.46

### 3.3.5 Partie entière plus longue que la longueur souhaitée

Si la longueur spécifiée est insuffisante, l'affichage n'en tiendra pas compte.

```
float reel = 123.456;  
printf("Nombre:%2.1f", reel);
```

Nombre:123.5

Il est impossible ici d'afficher le nombre 123.456 sur une longueur de 2. C'est la longueur par défaut de la partie entière qui est donc utilisée.

### 3.3.6 Affichage du signe

Comme pour les entiers, on peut forcer l'affichage du signe.

```
float reel = 123.456;  
printf("Nombre:%+.2f", reel);
```

Nombre:+123.46

## 3.4 Écriture scientifique d'un nombre réel

L'écriture scientifique est une autre manière d'écrire un nombre réel, basée sur les puissances de 10. Par exemple, le nombre 123.456 peut s'écrire  $1.23456 \times 10^2$ .

Pour forcer l'écriture scientifique d'un nombre réel en C, il faut utiliser le format %e.

```
float reel = 123.456;  
printf("Nombre:%e", reel);
```

Nombre:1.234560e+002

ATTENTION, l'écriture scientifique n'est prévue que pour les nombres réels. il n'est pas possible d'afficher l'entier 100 000 000 en écriture scientifique.



### 3.5 Les caractères

#### 3.5.1 Format de base pour les caractères

Le format pour afficher un caractère est "%c". Tout caractère affichable occupera une seule position à l'écran.

#### 3.5.2 Caractères spéciaux

Il existe quelques caractères spéciaux non affichables qui sont interprétés de manière particulière :

<code>\n</code>	provoque une saut de ligne
<code>\t</code>	insère une tabulation
<code>\r</code>	provoque un retour en début de ligne

#### 3.5.3 Afficher les caractères réservés

Quand on utilise l'instruction `printf`, on l'a vu, on utilise le caractère " pour encadrer le message et on peut utiliser le caractère % dans un format et le caractère \ si on ajoute un caractère spécial. Mais pour afficher ces caractères tels quels à l'écran, suivez ces exemples :

```
printf("tva=20%%");  
printf("Caractère backslash : \\") ;  
printf("Le caractère \" permet d'encadrer le message");
```

#### 3.5.4 Afficher le code d'un caractère

Tout caractère possède un code. Pour afficher le code d'un caractère, il suffit d'afficher ce caractère en tant que valeur entière.

```
printf("Code du caractere A : %d", 'A');
```

Code du caractere A : 65

#### 3.5.5 Afficher un caractère par son code

À l'inverse, on peut afficher un caractère grâce à son code.

```
printf("Caractere : %c", 65);
```

Caractere : A

### 3.6 Les booléens

Il n'existe pas de format défini pour le pseudo type booléen. Il est possible d'afficher une valeur booléenne soit sous forme d'entier (1 pour `true` et 0 pour `false`) soit sous forme de chaîne de caractères ("`true`" / "`false`" ou bien "`vrai`" / "`faux`", etc.).

#### 3.6.1 Afficher un booléen comme un entier

Il faut dans ce cas utiliser le format "`%d`".

Exemple :

Le code...

```
bool monBooleen = true;
printf("Valeur du booleen :%d", monBooleen);
```

...produira : **Valeur du booleen : 1**

#### 3.6.2 Afficher un booléen comme une chaîne de caractères

Il faut dans ce cas créer une fonction qui par exemple remplace 1 par "`true`" et 0 par "`false`". Un moyen très simple consiste à définir une macro avec `#define`.

Le code...

```
#define valBool(b) (b==true)?"true":"false"1
...
bool monBooleen = true;
printf("Valeur du booleen :%d", valBool(monBooleen));
```

...produira **Valeur du booleen : true**

### 3.7 Les autres types de données

Dans le cas d'une valeur de type tableau ou structure, il faudra écrire une procédure ou fonction particulière pour son affichage.

1 L'expression contenant `?` et `:` est un opérateur ternaire ; c'est un raccourci pour `if...else`.