

# GNU/Linux Shell Bash

- [commande shell bash](#)
- [Programmation Bash-fr.pdf](#)
- [Introduction à la programmation en Bash FR](#)

## Gestion des Dates et heures sous Linux avec Bash

1. `#!/bin/bash`
2. `man date` # affiche le manuel et permet d'obtenir les commandes suivantes
3. `date +%H%M` # affiche l'heure sous forme 0804 ==> 08h 04mn
4. `date +%Y%m%d` # affiche la date sous forme 20180609 année mois jour
5. # pour avoir les variables
6. `heure=$(date +%H%M)` # insere dans la variable "heure" l'heure sous forme 0804
7. `jour=$(date +%Y%m%d)` # date est une commande donc je change de nom

## Utilisation de la commande date

Pense-bête pour l'utilisation de la commande date en bash / shell.

### Base

Retourne la date du jour avec les règles de localisation de la machine, par exemple pour une bécane Franco-française :

```
date retourne mardi 17 novembre 2009, 06:55:32 (UTC+0100)
```

### l'option -d

Ensuite y'a la paramètre -d assez "marrant" qui permet ce genre de chose

```
date -d 'now' # retourne mardi 17 novembre 2009, 06:57:53 (UTC+0100)
```

```
date -d 'yesterday' # retourne lundi 16 novembre 2009, 06:58:32 (UTC+0100)
```

```
date -d "tomorrow" date -d "year" # retourne mercredi 18 novembre 2009, 06:58:55 (UTC+0100)
```

```
date -d "days" # retourne mardi 18 novembre 2009, 06:59:15 (UTC+0100)
```

```
date -d "week" # retourne mardi 24 novembre 2009, 06:59:30 (UTC+0100)
```

```
date -d "month" # retourne jeudi 17 décembre 2009, 06:59:59 (UTC+0100)
```

```
date -d "year" # retourn mercredi 17 novembre 2010, 07:03:08 (UTC+0100)
```

## Ensuite on peut utiliser des précisions pour le nombre de jour/semaine/mois/année

`date -d "3 days" # retourne date 'now' + 3 jours`

`date -d "2 week" # Retourne la date dans 2 semaines`

Bon etc, ça marche pour jour, semaine, mois année, ensuite on peut ajouter le mot ago pour afficher la date passé.

`date -d "1 month ago" # retourne la date il y a un mois`

Pareil, ça marche pour les jours, semaines, mois et années Mettre en forme la date

Enfin (je vais terminer la dessus), on peut mettre en forme la date un peu à la manière de `date()` en php

`date "+%Y-%m-%d" # return ANNEE-MOIS-JOUR`

Notez que la chaine de caractère (pattern de format) doit être rédigé de façon assez précise, elle commence par un '+' et les caractères de substitutions sont toujours précédés d'un '%'.  
Pour obtenir par exemple en timestamp au format mysql ça donne

`date "+%Y-%m-%d %H:%M:%S"`

`date "+%Y-%m-%d %H:%M:%S"`

Voici un petit tour rapide des patterns supportés (les principaux) Année

```
%Y : Année sur 4 chiffres  
%C : Le siècle (en gros les 2 premiers chiffres de l'année, si elle a 4 chiffres...)
```

Mois

```
%b : Nom du mois sur 3 lettres  
%B : Nom du mois  
%m : Numéro du mois sur 2 chiffres
```

Jours

```
%a : Nom du jour de la semaine sur 3 lettres  
%A : Nom du jour de la semaine  
%d : Numéro du jour dans le mois sur 2 chiffres  
%j : Numéro du jour dans l' année
```

Heures

```
%H : Heures sur 24 heures  
%I : Heures sur 12 heures
```

Heures

```
%M : Minutes sur 2 chiffres
```

Secondes

```
%S : Secondes sur 2 chiffres
```

Raccourcis

```
%F : YYYY-MM-DD
```

```
%T : HH-MM-SS
```

Voilà pour les principaux patterns, un **man date** vous les détaillera tous

### Pour finir

Sachez enfin qu'un cumul de -d "durée" et de "+%PATTERN" est possible et cela s'avère parfois assez pratique :

```
date -d "2 week" "+%F %T" # Retourne un timestamp MySQL du jour qu'il sera dans 2 semaines
```

# Créer des boîtes de dialogues en Bash

Boîtes de dialogue avec **Whiptail**

[Boite de dialogue en Bash](#)

Utilisation de la commande **dialog** sous bash

[Commande Dialog](#)

[man dialog en Français](#)

[doc dialog en francais](#)

Utilisation de **zenity** sous bash

[doc zenity](#)

From:

<https://magenealogie.chanterie37.fr/www/fablab37110/> - Castel'Lab le Fablab MJC de Château-Renault

Permanent link:

<https://magenealogie.chanterie37.fr/www/fablab37110/doku.php?id=start:rasberry:bash&rev=1673448203>

Last update: **2023/01/27 16:08**

