Commandes & Processus

1 - Commandes

Qu'est-ce qu'une commande ?

Une commande peut être :

- un binaire : résultat d'une compilation.
- un script : une succession d'autres commandes.

Une commande est un programme.

C'est un programme assez rudimentaire : ne fait généralement qu'une seule action assez simple.

2 - Comment lancer une commande ?

La console

Historiquement, dans une console c'est/c'était :

- Un clavier
- Un écran, voire même une simple imprimante.
- Reliée directement à un port de l'ordinateur central.

Une console est un terminal dédié à l'administrateur, utilisable sans aucun réseau.

La console existe toujours sous Linux :

- CTRL+ALT+F1 à F6 : accès à 6 consoles.
- CTRL+ALT+F7 : accès à l'interface graphique si elle est lancée.

Le terminal

Historiquement, un terminal c'est/c'était :

- Un équipement passif (sans os) avec un clavier et un écran branchés sur l'ordinateur par des fils.
- Assez proche d'une console, mais dédié aux utilisateurs.

Aujourd'hui : console et terminal sont généralement des logiciels et ils nécessitent un accès réseau.

Le shell

Le shell tourne dans une console ou un terminal. C'est un logiciel dont le rôle est :

- Lire des ordres (commandes) au clavier.
- Demander à l'OS d'exécuter les ordres.
- Afficher éventuellement ce que les commandes produisent comme résultat.
- Attendre la fin de l'exécution de la commande.
- Recommencer cette séquence.

Le shell apporte une aide à la saisie des ordres :

- Complétion des noms d'objets : <TAB>, <TAB><TAB>.
- Historique :
  - commande history.
  - parcours de l'historique avec flèches haut et bas.
  - CTRL+R : recherche d'une ancienne commande.
  - !<ancienne_commande>
- Développement des jokers et des variables (vu plus tard).

Déroulé d'une commande

Une commande est un fichier sur disque (binaire pour un programme compilé ou texte pour un script).

Les scripts seront vus plus tard.

Une commande lancée devient un processus, quelque chose qui vit dans la mémoire et le CPU de l'ordinateur.

3 - Processus

Une commande, un programme, une application : s'exécutent en tant que processus (process).

Une commande, un programme : c'est juste une recette pour faire quelque chose, c'est statique.

Une recette décrit des étapes dans un livre.
Une commande décrit des actions dans un fichier sur disque.

Processus = l'exécution, la mise en œuvre de la "recette".

Le chef réalise la recette, ça remue dans la cuisine.

Un processus fait des calculs, ça affiche parfois des choses, ça crée des fichiers, ça peut faire du bruit (jouer des sons par exemple), ça interagit parfois avec l'utilisateur, on a l'impression d'être au contact d'une certaine forme d'intelligence, on a bien un sentiment de "vie", même artificielle. Ca bouge dans la RAM et le CPU !

Commande peut être :

- Compilée (issue du C généralement).
- Interprétée (succession d'autres commandes).

Où sont les commandes ?

Les commandes sont donc des fichiers sur disque.
Mais, où sont-elles ? :

- /bin (binaires, mais on peut y trouver des scripts)
- /usr/bin
- /usr/local/bin
- /sbin (réservé à root, statiques)
- /usr/sbin (réservé à root, statiques)
- ~/bin (personnelles dans le home)

Quelques commandes ne sont pas dans des fichiers sur disque, mais sont des commandes internes au shell :

- cd
- exit
- logout
- jobs

D'autres peuvent être à la fois des commandes du shell et des commandes écrites dans des fichiers :

- echo et /bin/echo

Pour savoir si on a affaire à une commande interne ou une commande fichier, on peut utiliser la commande which. Exemple :

```
which ls
```

Ca affiche la localisation du fichier de commande ou rien s'il s'agit d'une commande interne. Exemple :

```
/bin/ls
```

ps - Lister les processus

Origine du nom : process ou processes
Syntaxe : ps [options]

Quelques exemples :

- ps : Processus de la session de l'utilisateur connecté.
- ps -edf (affichage détaillé de tous les processus).
- ps -axuwf (affichage hiérarchique détaillé).

Les colonnes d'un ps -edf :

- UID : propriétaire du processus (généralement l'utilisateur qui a lancé la commande).
- PID : process ID (unique).
- PPID : process ID du processus parent.
- STIME : date & heure de lancement.
- TIME : temps d'exécution (CPU).
- CMD : la commande (+ options et paramètres) ayant servi au lancement.

Comme pour les objets (fichiers, dossiers) qui ont tous un objet parent, les processus ont aussi un processus parent. C'est aussi une structure arborescente.

Lancer une commande crée un processus. Quel en est le parent ?

C'est simplement le processus qui lance cette commande, le shell ! Il est lui-même un processus.

Il s'agit d'une structure arborescente de processus qui ont chacun un processus parent qui, à son tour, a un processus parent, etc.

Le FS a sa racine (/).

L'arbre des processus a aussi sa racine : le processus init, issu de la commande /sbin/init, et son PID est toujours 1. C'est le 1er processus lancé par l'OS. Son rôle est capital, il hérite des processus qui perdent leur parent.

Autre moyen d'afficher tous les processus sous forme hiérarchique :

pstree -psh

Linux est un OS multitâche. Il y a de nombreux processus exécutés en même temps.

Question légitime : le kernel est-il un processus ?

La réponse est clairement : Non.

C'est du code qui s'exécute mais ce n'est pas un processus. Il n'a pas de PID par exemple. Il n'apparaît donc pas dans ps.

Manipuler les processus

Un utilisateur peut manipuler uniquement ses propres processus.

root peut manipuler tous les processus.

Pour manipuler un processus, il faut connaître son PID (avec ps -edf par exemple).

kill

La commande kill envoie un signal à un processus.
On verra les signaux en 2ème année.

Il existe un signal qui s'appelle KILL et qui porte le n° 9.

Le signal KILL ordonne à l'OS d'arrêter immédiatement un processus.

Syntaxe :

kill -KILL pid [pid...]
kill -9 pid [pid...]

Attention à ne pas se tromper de PID. Ca peut être catastrophique.
Exemple : kill -KILL -1 tue tous les processus !!!
Avec root : ça tue tout ce qui tourne. Il faut courir vite ensuite...

killall

La commande killall permet d'envoyer un signal, comme kill, mais à un ensemble de processus qui portent le même nom. Exemple :

killall bash

Tue tous les processus qui s'appellent bash. Et sans doute aussi celui dans lequel on est actuellement ! :-)

4 - Asynchronisme

L’asynchronisme est le caractère de ce qui ne se déroule pas à la même vitesse. C’est à dire, 2 choses qui vivent leurs vies en parallèle, à leur rythme.

C’est assez habituel sur les OS modernes.

Exemple : télécharger un fichier depuis un site Web ne bloque pas un film qu'on regarde en même temps, ou la réception des mails. C’est l’asynchronisme des processus.

Concernant le lancement d'une commande par le shell :

- En mode synchrone, l'accès au shell est bloqué jusqu'à la fin de la commande. L'utilisateur attend. On dit que l'exécution se fait "en avant plan".
  Arrêt avec CTRL+C.
- En mode asynchrone, l'accès au shell n'est pas bloqué. Le shell reprend la main immédiatement. On dit que l'exécution se fait "en arrière plan". La commande vit sa vie de son côté, et le shell redevient alors le processus d'avant plan. Mais, pas de CTRL+C possible !

Problème :

- Il n'y a qu'un seul clavier.
- L'accès au clavier est donné au processus en avant plan, c'est à dire : le shell.

Avec l'écran, il n'y a pas de problème, il est partagé.

Exemple du tableau blanc : tout le monde peut écrire dessus en même temps. Evidemment ça peut s'entremêler un peu si tous les processus s'y mettent !

Lancer en arrière plan

Pour lancer une commande en arrière plan, il suffit d'ajouter & en fin de ligne.

Le shell lance la commande et reprend la main immédiatement après avoir affiché le PID. Exemples :

- firefox &
- geany &

Lancer en arrière plan est utile uniquement pour les longs processus. Exemple : ps & ne sert pas à grand chose.

Certains programmes interagissent fortement avec l'utilisateur par le clavier, comme vi (l'éditeur de texte).

Quand un processus en arrière plan tente de lire au clavier, il est mis en pause pas l'OS. Il sortira de son "sommeil forcé" quand il repassera en avant plan.

Les programmes avec une Interface Graphique (UI) telle que Gnome ou KDE, tournent sans soucis en arrière plan.

Tout l'environnement graphique, toutes les fenêtres des applications graphiques, sont autant de processus qui fonctionnent de façon asynchrone.

Tous ces processus travaillent en arrière plan. Cliquer sur une fenêtre ramène simplement le processus de la fenêtre en avant plan, lui donnant alors accès au clavier.

Jobs

Quand un programme est lancé en arrière plan (donc avec &), il vient se mettre dans une liste qu'on appelle les Jobs.

La commande jobs est une commande du shell qui affiche cette liste de processus qu'il a lancés lui-même.

Les jobs sont numérotés de 1 à N.

Ces numéros n'ont rien à voir avec les PID des processus.

On peut aussi arrêter un processus par son numéro de job, avec un kill, comme ceci :

kill -TERM %<num_job>

Envoyer en arrière plan

Un processus qui est en avant plan peut être envoyé en arrière plan en appuyant sur CTRL+Z. Il passe alors en arrière plan en mode pause.

La commande bg (background, arrière plan) permet de laisser en arrière plan le dernier job utilisé, en le sortant de sa pause éventuelle.

La commande fg (foreground, avant plan) permet de faire revenir en avant plan le dernier job utilisé.

Il devient alors le processus qui est attaché au clavier.

Attaché au clavier signifie qu'il peut lire des choses au clavier mais il peut aussi être arrêté par CTRL+C ou renvoyé en arrière plan avec un CTRL+Z.

Pour utiliser fg et bg sur des jobs autres que le dernier utilisé, on peut simplement ajouter le numéro du job (pas le PID) :

- bg 2
- fg 3

Certaines commandes, certains programmes, refusent le CTRL+Z ou le CTRL+C, ils en ont la possibilité.

Par exemple, vi ne réagit à aucune de ces 2 actions.

5 - Jouons un peu

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

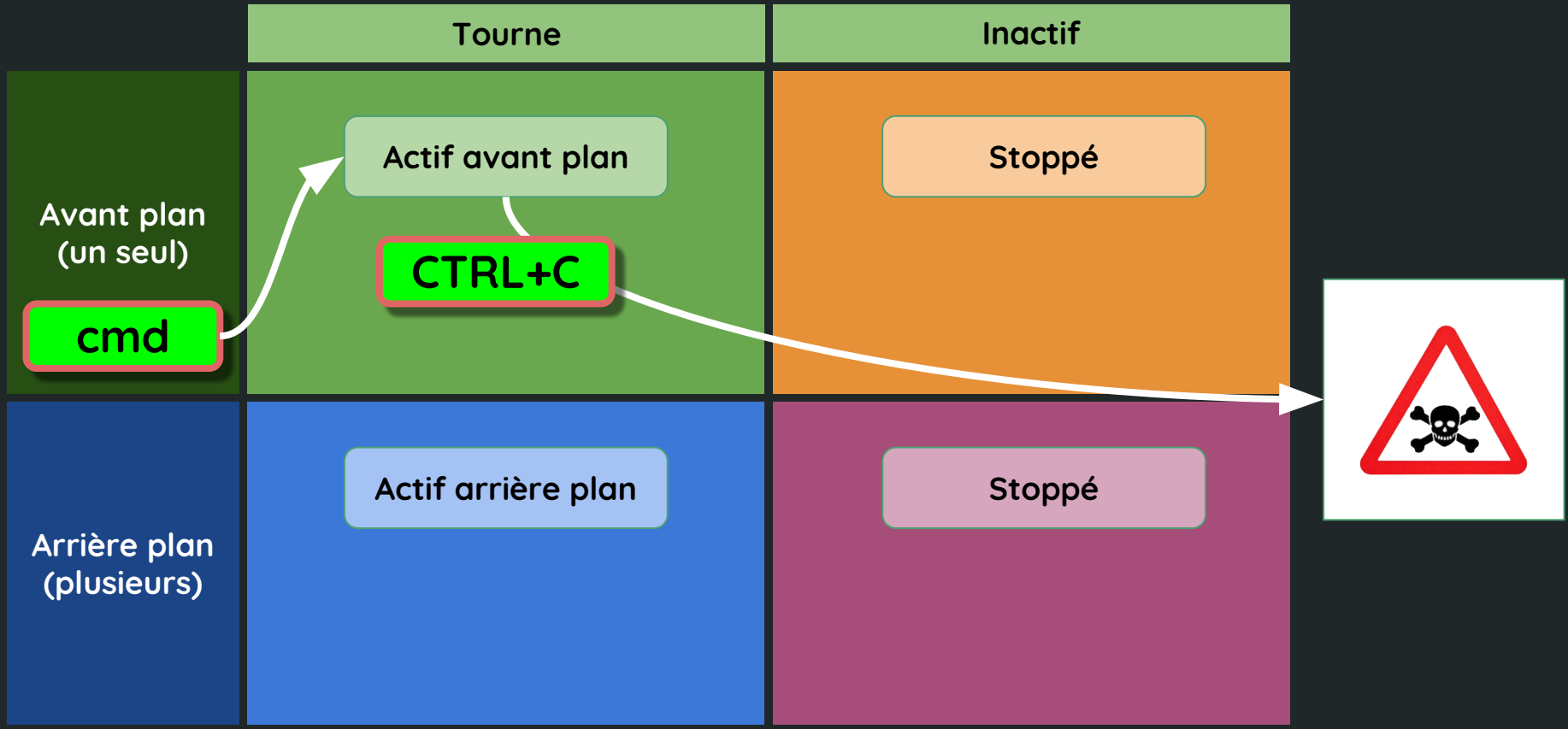
	Tourne	Inactif
Avant plan (un seul) cmd	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan CTRL+C	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd



	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd

CTRL+Z

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd

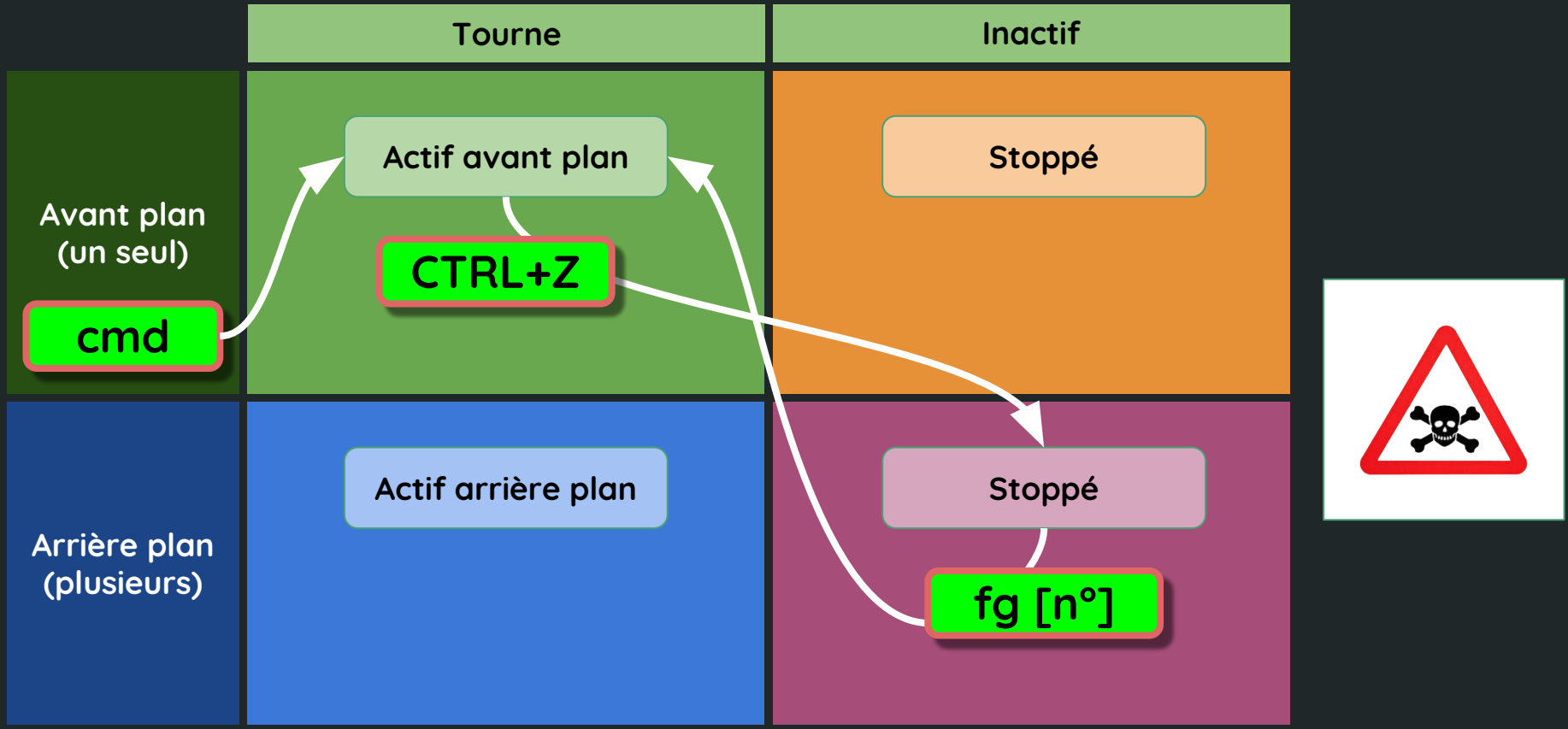
CTRL+Z

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd

CTRL+Z

fg [n°]



	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd

CTRL+Z

bg [n°]

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd

CTRL+Z

bg [n°]

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs) cmd &	Actif arrière plan	Stoppé

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

scanf

cmd &

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

scanf

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

fg [n°]

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

fg [n°]

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

kill -9 %n°

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

kill -9 %n°

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

CTRL+C

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

Pfffttt

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

CTRL+Z

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

cmd &

Pfffttt

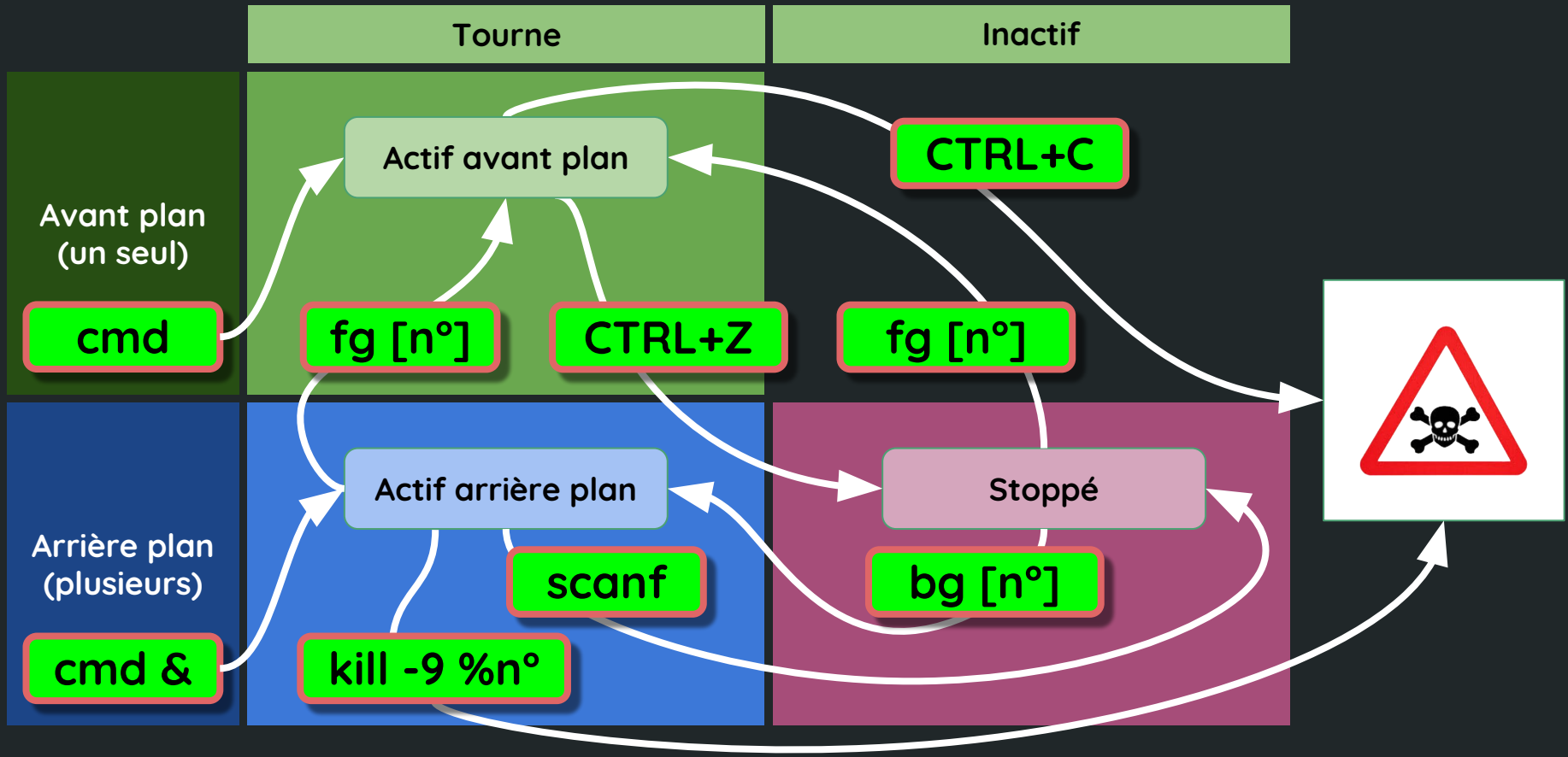
	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé

?

?

?

	Tourne	Inactif
Avant plan (un seul)	Actif avant plan	Stoppé
Arrière plan (plusieurs)	Actif arrière plan	Stoppé



https://commons.wikimedia.org/wiki/File:Thats_all_folks.svg