[ISS-2020] Tutoriel d'installation de Bash 5.0 sur MacOS

- Depuis MacOS Catalina, Apple a remplacé Bash par Zsh comme Shell par défaut. Il y a donc de fortes chances que le Shell que vous utilisez sur votre Mac soit Zsh, et malgré que ce dernier soit très proche de Bash, quelques fonctionnalités, dont certaines utilisées dans le cadre de l'UE, différents. Je vous recommande donc de privilégier Bash sur votre Mac pour cette UE, et ce tutoriel a pour but de vous aider pas-à-pas à le configurer.
- Pourquoi ne suffit-il pas d'utiliser la version native dans `/bin/bash` ?
  - La version de Bash recommandée pour notre UE est la 4.0 (au minimum, sachant que la 5.0 a été release en 2019). Or, si vous exécutez la commande:

```
bash --version
GNU bash, version 3.2.57(1)-release (x86_64-apple-darwin19)
Copyright (C) 2007 Free Software Foundation, Inc.
```

  - Comme vous pouvez le voir, la version installée nativement est la 3.2, qui date de 2007!! La raison pour laquelle Apple n'a pas mis à jour celle-ci est que la version suivante (la 4.0) est passée sous licence GPLv3 incompatible avec la politique d'Apple qui veut signer son propre code.
  - Bien heureusement, il est possible d'installer une version plus récente, alors passons au concret.

1. Installation de Brew

- Si vous ne l'avez pas déjà installé, Homebrew est le gestionnaire de packets le plus utilisé dans l'univers Mac, il vous sera de toute manière utile dans votre vie de dev... alors pour l'installer il suffit d'executer la commande:

```
$ /bin/bash -c "$(curl -fsSL
<https://raw.githubusercontent.com/Homebrew/install/master/install.sh>)"
```

2. Installation de Bash 5.0

- On va donc installer le packet Bash depuis Brew, en exécutant la commande:

```
$ brew install bash
```

- On peut se servir de la commande `which` avec son option `-a` vue en TD pour vérifier qu'on a maintenant deux versions de Bash installées:

```
$ which -a bash
/usr/local/bin/bash
/bin/bash
```

- On peut également vérifier la version de chacun des deux programmes:

```
$ /usr/local/bin/bash --version
GNU bash, version 5.0.0(1)-release (x86_64-apple-darwin18.2.0)
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

$ /bin/bash --version
GNU bash, version 3.2.57(1)-release (x86_64-apple-darwin18)
Copyright (C) 2007 Free Software Foundation, Inc.
```

- Vu que la nouvelle version dans `/usr/local/bin` vient par défaut avant l'ancienne dans `/bin` dans `$PATH` la version qui sera utilisée lorsque nous tapons simplement `bash` sera la nouvelle:

```
$ bash --version
GNU bash, version 5.0.0(1)-release (x86_64-apple-darwin18.2.0)
Copyright (C) 2019 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software; you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
```

3. Autoriser la nouvelle version à la connexion

- Dans les systèmes UNIX une protection permet de restreindre les Shell pouvant être utilisés comme des Login Shells. Pour autoriser notre nouvelle version de Bash, il faudra rajouter ce dernier dans le fichier listant les Shells autorisés `/etc/shells` , pour ce faire on va d'abord ouvrir ce fichier avec les droits sudo:

```
$ sudo nano /etc/shells
```

- On y rajoute le Shell `/usr/local/bin/bash` :

```
# List of acceptable shells for chpass(1).
# Ftpd will not allow users to connect who are not using
# one of these shells.

/bin/bash
/bin/csh
/bin/dash
/bin/ksh
/bin/sh
/bin/tcsh
/bin/zsh
/usr/local/bin/bash
```

- On n'oublie pas d'enregistrer le fichier et on sera bon pour cette étape.

4. Définir le Shell par défaut

- Il suffit d'exécuter la commande suivante:

```
$ chsh -s /usr/local/bin/bash
```

- Cette commande change le Shell par défaut pour l'utilisateur l'ayant exécutée, il faudra donc l'exécuter en tant que sudo:

```
$ sudo chsh -s /usr/local/bin/bash
```

5. Utiliser le bon Shebang

- Cette solution implique que nos scripts Bash devraient utiliser comme Shebang `#!/usr/local/bin/bash` alors que `#!/bin/bash` est presque un standard.
- Je vous recommande de privilégier la version portable des Shebang:

```
#!/usr/bin/env bash
```

- Cette version inspecte le `PATH` pour utiliser le `bash` par défaut, et fera en sorte que vos scripts soient plus portables.

En option, garder uniquement la version récente

- Le tutoriel présenté conserve inutilement l'ancienne version dans `/bin/bash`, pour rester compatible avec le SIP (System Integrity Protection d'Apple) qui consiste à interdire l'accès en modification à certains dossiers (y compris en root) dont `/bin` concerné ici.
- Bien heureusement on peut désactiver cette protection (en tant qu'informaficiens adultes qui savent ce qu'ils font), mais pour cela il faudra aller un peu plus loin dans nos manips:
  1. Démarrer votre machine ne mode `Recovery OS` en redémarrant votre machine puis en restant appuyé simultanément sur `Command` et `R` au démarrage;
  2. Attendre que l'OS démarre puis ouvrir l'app `Terminal` dans le menu `Utilities` ;
  3. Saisir la commande:

```
csrutil enable
```

  1. Fermer `Terminal` est redémarrer la machine (normalement). Le SIP est maintenant désactivé.
- Maintenant que le SIP est désactivé, on va supprimer l'ancienne version de Bash est la remplacer par un lien symbolique vers la nouvelle:

```
$ sudo rm /bin/bash
$ sudo ln -s /usr/local/bin/bash /bin/bash
```

- Enjoy!

===

Ref: https://apple.stackexchange.com/questions/193411/update-bash-to-version-4-0-on-osx/292760#292760