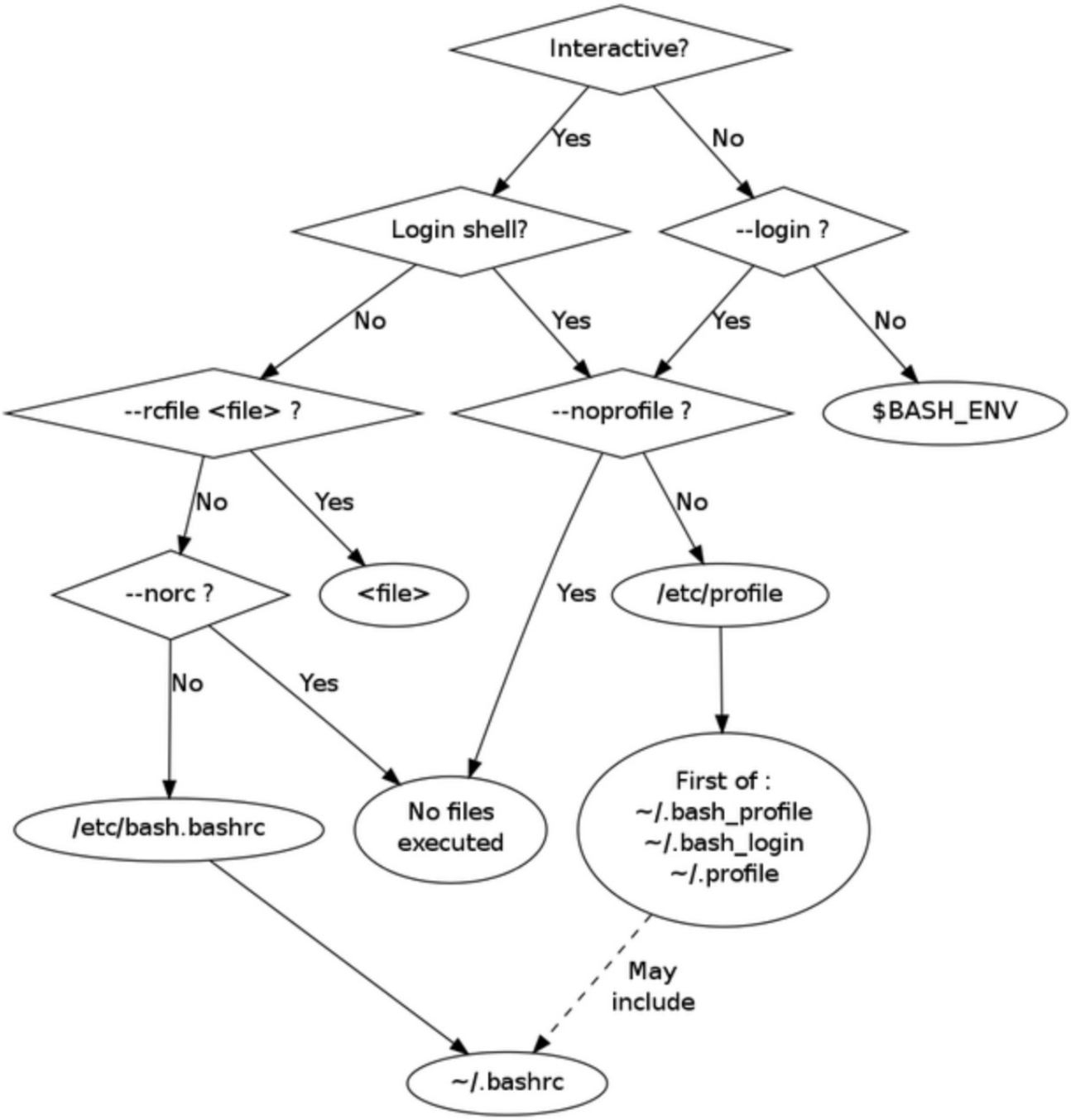
【Linux】載入順序



https://blog.miniasp.com/post/2021/07/26/Bash-and-Zsh-Initialization-Files

- 這張圖的第一層 `Interactive?` 的意思
  Yes: 當你透過 Console 或 SSH 登入 Linux 主機，預設會進入**互動模式** (Interactive shell)，也就是這裡說的 **Interactive** 的意思。
  No: 任何透過 `bash -c '<command>'` 去執行的腳本，就屬於 **非互動模式** (Non-Interactive shell) 的執行。
- 這張圖第二層的 `Login shell?` 的意思
  基本上，透過 Console 或 SSH 登入 Linux 主機時，這個 Shell 就跑在所謂的**Login shell** 模式下！
  不過，當你透過 SSH 遠端執行一個命令，此時就不會啟動 **Login shell** 模式。而直接呼叫一個使用 Bash 執行的腳本，也不是 **Login shell** 模式。例如以下命令：

  ```
  ssh user@host <COMMAND>
  ```

- 這張圖第二層的 `--login ?` 的意思
  就算你的 Bash 不是執行在 Login shell 模式，你一樣可以在呼叫 `/bin/bash` 的時候特別加上 `--login` 參數，這樣也可以被視為是 **Login shell** 模式。
  例如以下這段範例，就會被視為使用 Login shell 模式啟動：

  ```
  #!/bin/bash --login
  ```

- 這張圖第三層的 `$BASH_ENV` 是什麼環境變數？
  當你將 Bash 啟動在**非互動模式**，也沒有特別加上 `--login` 參數的情況，同時這也是大多數執行的預設值，Bash 會優先尋找目前的環境變數中有沒有一個名為 `$BASH_ENV` 的變數，這個變數其實是指向一個檔案路徑。你從 `man bash` 可以發現，Bash 在**非互動模式**啟動的時候，預設會執行以下命令：

  ```
  if [ -n "$BASH_ENV" ]; then . "$BASH_ENV"; fi
  ```

  這也意味者，他會去 sourcing `"$BASH_ENV"` 這個檔案！
- 這張圖第三層的 `--noprofile` 是什麼參數？
  執行 `/bin/bash` 的時候，可以額外加上 `--noprofile` 參數，加上之後就不會載入任何啟動檔。
  如果你沒有加上 `--noprofile` 參數，也是一般大多數命令的執行方式，預設是會先載入系統全域的 `/etc/profile` 檔案。

  > 注意：系統全域的 `/etc/profile` 檔案，還會額外載入 `/etc/profile.d/*.sh` 檔案。

  然後再從 `$HOME` 目錄下**依序**找到這三個檔案執行，但重點是，**他只會選擇一個檔案來執行**，先找到的先執行，後面的就不會執行！
  1. `~/.bash_profile`
  2. `~/.bash_login`
  3. `~/.profile`

  由於上述三個檔案，最終只有一個會被執行，所以這絕對是一個潛在的**地雷**！
  我在多年前，就曾經因為我的 `$HOME` 目錄下同時出現 `~/.bash_profile` 與 `~/.profile` 而發生系統異常，原來只要目錄中有出現 `~/.bash_profile` 檔案，就再也不會載入 `~/.profile` 檔案啊！！！

  > 在 Ubuntu 的作業系統中，預設是看不到 `~/.bash_profile` 檔案的，建議都以 `~/.profile` 為主要的登入啟動檔！

  然而，大多數的 `~/.profile` 登入啟動檔，都會在檔案中額外載入 `~/.bashrc` 檔案，因此有些 Bash 相關的環境設定，如 `shopt` 之類的，都會放在 `~/.bashrc` 檔案中。
- 這張圖第三層的 `--rcfile <file>` 是什麼參數？
  當你不是以 Login shell 的方式啟動 Bash，啟動時也沒加上 `--login` 參數，他就會去找你有沒有特別加上 `--rcfile <file>` 參數，明確載入你所指定的檔案路徑。
  如果你額外加上的是 `--norc` 參數的話，那就代表你完全不想載入系統全域的 `/etc/bash.bashrc` 與 `$HOME` 目錄下的 `~/.bashrc` 檔案。
  如果你用 `--rcfile <file>` 參數找不到指定的檔案，或完全沒用 `--rcfile <file>` 或 `--norc` 參數，也是大多數命令預設的方式，那麼 Bash 就會依序載入 `/etc/bash.bashrc` 與 `~/.bashrc` 檔案，兩個檔案都會載入。這種情境下，是不會載入 `~/.bash_profile`, `~/.bash_login` 或 `~/.profile` 檔案的！

可以完整釐清 Bash 所有啟動檔的載入條件與順序，心裡的感覺格外踏實，非常棒！

完整的 Zsh 啟動檔載入順序

由於許多 macOS 用戶都採用 Zsh 為主，這邊我也特別研究了一下 Zsh 的啟動檔載入順序，基本上 Zsh 會依據以下順序載入：

1. `~/.zshenv`
   任何啟動情境下，都會載入這個檔案，請將各種環境變數請全部設定在這裡。
2. `/etc/zsh/zprofile` 與 `~/.zprofile`
   如果執行在 Login shell 才會依序執行 `/etc/zsh/zprofile` 與 `~/.zprofile` 檔案。
3. `/etc/zsh/zshrc` 與 `~/.zshrc`
   如果執行在 Interactive 互動模式下，才會依序執行 `/etc/zsh/zshrc` 與 `~/.zshrc` 檔案。
4. `/etc/zsh/zlogin` 與 `~/.zlogin`
   如果執行在 Login shell 下，最後才會依序執行 `/etc/zsh/zlogin` 與 `~/.zlogin` 檔案。
5. `~/.zlogout` 與 `/etc/zsh/zlogout`
   當你用 `exit` 或 `logout` 命令登出時，會自動依序執行 `~/.zlogout` 與 `/etc/zsh/zlogout` 檔案。

我只能說，Zsh 的啟動順序實在比 Bash 好理解太多了，也沒什麼地雷！

總結幾種常見情境

1. 直接呼叫某個 Shell Script

   ```
   #!/bin/bash
   [ -z "$PS1" ] && echo "Non-Interactive" || echo "Interactive"
   shopt -q login_shell && echo "Login shell" || echo "Not login shell"
   ```

   - `Interactive?` `No`
   - `Login shell?` `No`
   - `--noprofile?` `No`

   執行時完全不會載入任何新的啟動檔設定！
2. 使用 SSH 登入遠端主機

   ```
   ssh user@host
   ```

   依序載入 `/etc/profile` --> `/etc/bash.bashrc` --> `~/.profile` [-> `~/.bashrc`]
   - `Interactive?` `Yes`
   - `Login shell?` `Yes`

- `--noprofile?` ➡ `No`

> 注意: Bash 不會主動載入 `~/.bashrc` 執行，而是我們通常在 `~/.profile` 會載入 `~/.bashrc` 執行。

3. 使用 SSH 登入遠端主機後，執行 `bash` 命令

```
ssh user@host
```

剛登入，依序載入 `/etc/profile` --> `/etc/bash.bashrc` --> `~/.profile` [-> `~/.bashrc`]
- `Interactive?` ➡ `Yes`
- `Login shell?` ➡ `Yes`
- `--noprofile?` ➡ `No`

然後我們在遠端主機執行 `bash` 命令，進入下一層 Shell 環境：

```
bash
```

登入後執行 `bash` 預設是**互動模式**，會依序載入 `/etc/bash.bashrc` --> `~/.bashrc`
- `Interactive?` ➡ `Yes`
- `Login shell?` ➡ `No`
- `--rcfile <file> ?` ➡ `No`
- `--norc ?` ➡ `No`

4. 使用 SSH 登入遠端主機後，執行 `bash -c '<command>'` 命令

```
ssh user@host
```

剛登入，依序載入 `/etc/profile` --> `/etc/bash.bashrc` --> `~/.profile` [-> `~/.bashrc`]
- `Interactive?` ➡ `Yes`
- `Login shell?` ➡ `Yes`
- `--noprofile?` ➡ `No`

```
bash -c '[ -z "$PS1" ] && echo "Non-Interactive" || echo "Interactive"'
```

登入後執行 `bash -c` 命令屬於非互動模式，而且沒有 `$BASH_ENV` 的情況下，不會載入任何啟動檔！
- `Interactive?` ➡ `No`
- `Login shell?` ➡ `No`
- `$BASH_ENV` ➡ `No`

5. 使用 SSH 執行遠端命令

```
ssh user@host '[ -z "$PS1" ] && echo "Non-Interactive" || echo "Interactive"'
ssh user@host 'shopt -q login_shell && echo "Login shell" || echo "Not login shell"'
```

透過 SSH 執行遠端命令，預設將會跑在 **Non-Interactive** 模式下，但還是會依序載入 `/etc/bash.bashrc` --> `~/.bashrc`
- `Interactive?` ➡ `No`
- `Login shell?` ➡ `No`

如果你想在執行遠端命令時載入額外的啟動檔，也可以這樣寫：

```
ssh user@host "source /etc/profile; source ~/.profile; /your/script.sh"
```

6. 使用 SSH 執行遠端命令，透過 `bash` 呼叫另一個命令

```
ssh user@host -t 'bash -c '"'"'[ -z "$PS1" ] && echo "Non-Interactive" || echo "Interactive"'"'"''
ssh user@host -t 'bash -c '"'"'shopt -q login_shell && echo "Login shell" || echo "Not login shell"'"'"''
```

> 要在單引號(`'`)的字串中間加入一個單引號，必須輸入五個字元 `'"'"'` 才能代表一個單引號！（噁心的語法）

> 在 ssh 執行時加入 `-t` 參數，可以在遠端執行時取得一個 pseudo-terminal allocation (`pts/0`)！

透過 SSH 執行遠端命令，預設將會跑在 **Non-Interactive** 模式下，但還是會依序載入 `/etc/bash.bashrc` --> `~/.bashrc`
- `Interactive?` ➡ `No`
- `Login shell?` ➡ `No`

第二層 `bash` 應該是跑在 **Non-Interactive** 模式，但是卻依序載入 `/etc/bash.bashrc` --> `~/.bashrc` 檔案，**這部分我還沒辦法理解為什麼會這樣**，感覺透過 SSH 執行遠端程式，預設就會載入這兩個檔案！
- `Interactive?` ➡ `No`
- `Login shell?` ➡ `No`

底下這段是讓命令跑在 **Interactive** 模式，但載入啟動檔的順序竟然跟 **Non-Interactive** 竟然一樣！

```
ssh user@host -t 'bash -i -c '"'"'[ -z "$PS1" ] && echo "Non-Interactive" || echo "Interactive"'"'"''
ssh user@host -t 'bash -i -c '"'"'shopt -q login_shell && echo "Login shell" || echo "Not login shell"'"'"''
```

簡單來說，使用 SSH 直接遠端命令時，使用 `-i` (互動模式) 或不使用 `-i` 對載入啟動檔沒有什麼兩樣，不過都會載入兩次。不過還是有一個地方不同，如果你的**啟動檔**有設定 alias 的話，只有使用 `-i` 互動模式才能使用。
例如以下這段命令，就會得到 `bash: ll: command not found` 的錯誤：

```
ssh user@host -t 'll'
```

```
ssh user@host -t bash -c 'll'
```

如果改用 `-i` **來執行，就可以正常執行** `ll` **這個 alias 命令：**

```
ssh user@host -t bash -i -c 'll'
```

7. **使用 SSH 執行遠端命令，並以 Login shell 啟動** Bash

```
ssh user@host bash -l -c 'set'
```

第一層 `bash` **依序載入** `/etc/bash.bashrc` --> `~/.bashrc`
第二層 `bash` **依序載入** `/etc/profile` --> `~/.profile` [-> `~/.bashrc`]

修訂版本 #3
★由 treeman 建立於 2 ﷿T﷿﷿ 2023 15:10:48
由 treeman 更新於 10 ﷿Q﷿@﷿﷿ 2024 15:23:01