**REGEX**

|  |  |  |
| --- | --- | --- |
| **SYNTAXE** | | |
| /REGEX/ | Délimiteur de REGEX dans SED et AWK | |
| "REGEX" | Délimiteur de REGEX dans GREP - EGREP | |
| Espace dans REGEX = espace compté dans la chaine (l’espace n’est pas un délimiteur) | | |
| \. | Echappement. Doit être échappé :  . \* \ ^ $ | |
| (abc|xyz) \1 | Back référence : de 1 à 9  \1 se réfère au match du premier groupe : si abc a matché, \1 sera abc  (et non abc|xyz)  Uniquement supporté par GREP (et non EGREP) | |
| **GENERALE** | | |
| a | Toute occurrence avec a matchera | |
| .a | Wildcard : remplace n’importe quel caractère  **Ma**ry  **aHa**d  The  …  Toute lettre suivi d’un a  **Attention, le wildcard \* du Bash seul ne fonctionne pas : s’y prendre avec les quantificateurs.** | |
| ^<chaine> | La ligne commence par | |
| <chaine>$ | La ligne se finie par | |
| \(Mary\) | Version GREP : on échappe les parenthèses | Groupement de mots  If **Mary had** gone… |
| (Mary)( )(had) | Version EGREP  3 groupements : Mary, espace et had |
| [iou] | I ou o ou u | |
| [a-z]a | Une minuscule puis a | |
| [^0-9] | Ne contenant pas de chiffres  (= lettres ou chaine vide…) | |
| <pattern>|<pattern> | OU | |
| **QUANTIFICATEURS** | | |
| Aucun | Le pattern apparait exactement une fois. | |
| <pattern>\* | 0 ou plusieurs fois  .\* en REGEX fonctionne pareil que \* sous BASH | |
| <pattern>? | Apparait 0 ou 1 fois | |
| <pattern>+ | Apparait au moins une fois | |
| <pattern>{3} | Apparait 3 fois | |
| <pattern>{3,5} | Apparait de 3 à 5 fois | |
| <pattern>{3,} | Apparait au moins 3 fois | |
| **QUANTIFICATEURS SED ET AWK** | | |
| <pattern>\*?b | Autant de fois le pattern jusqu’à b  Empêche de prendre trop de chose dans le match | |

**Généralités :**

* REGularEXpression
* BASH ne reconnait pas les REGEX. BASH s’occupe du globbing :
  + Wildcard \* pour un ou plusieurs caractères
  + Wildcard ? pour n’importe quel caractère
  + Liste à une position [ab] ou [a-c] ou [^ab]
* Mnémotechnique : les Wildcard du Bash sont les Wildcard des quantificateurs REGEX
* Les REGEX sont greedy : ils match le plus qu’ils peuvent : on match des fois beaucoup trop
  + Classtype.\* ; ne s’arrêtera pas au premier point-virgule mais au dernier
  + Classtype[^ ;]\*. Changement de paradigme : on ne se demande pas ce qu’on match mais ce qu’on ne matchera pas plutôt