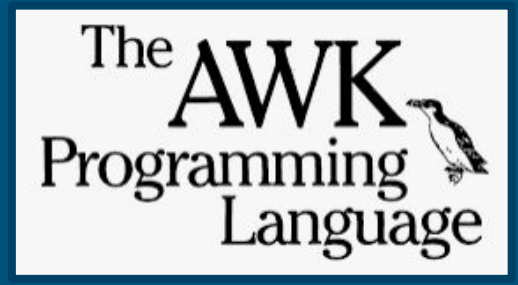
ASUR4 : Administration et sécurité des services

LABEGARIA Arnaud / HOVE Simon

GESTION DES SÉPARATEURS EN AWK



Sommaire :

Définition des séparateurs en AWK

Définition des séparateurs en AWK

Qu'est-ce qu'AWK ?

- AWK est un langage de traitement de ligne
- créé par Alfred Aho, Peter Weinberger et Brian Kernighan (première ver. en 1979)
- le plus souvent utilisé pour des fichiers textes
- langage assez puissant permettant des opérations complexes (recherche, remplacement, ...)

Définition des séparateurs en AWK

Qu'est-ce qu'AWK ?

- suite d'actions de la forme : motif { action }, le motif permettant de déterminer sur quels enregistrements est appliquée l'action
- deux "masques" spéciaux sont régulièrement utilisés :
  - BEGIN : définit un programme avant de commencer l'analyse du fichier
  - END : définit un programme après l'analyse du fichier

Un enregistrement : *une chaîne de caractères séparée par un retour chariot, en général une ligne.*
Un champ : *une chaîne de caractères séparée par un espace (ou par le caractère spécifié par l'option -F), en général un mot.*

Définition des séparateurs en AWK

Qu'est-ce qu'un séparateur ?

- séquence de un ou plusieurs caractères délimitant la "frontière" entre différentes "régions de texte"
- en AWK, les fichiers sont divisés en ***lignes***, puis en ***champs***
- les lignes sont séparées par défaut par "\n"
- les champs sont séparés par défaut par un espace ou une tabulation

Définition des séparateurs en AWK

Quels sont les différents séparateurs en AWK ?

- On en liste 5 : FS, RS, OFS, ORS, SUBSEP

Les différents séparateurs et leurs fonctions

Les différents séparateurs et leurs fonctions

FS : Field Separator

Permet de définir un séparateur entre chaque champ, différent de celui par défaut (qui est un simple espace), par un caractère ou une chaîne de caractère
Deux manières différentes d'utilisation
- *-F*
- comme une variable normale : *FS="x"*, *x* étant le séparateur choisi

Exemples :
awk -F":" 'commandes' nomdefichier
awk 'BEGIN{FS=":";}'

- permet de choisir : comme séparateur sur le texte donné en entrée

Les différents séparateurs et leurs fonctions

OFS : Output Field Separator

Séparateur qui permet d'afficher un ou plusieurs caractère(s) entre les champs en sortie
La valeur par défaut du séparateur OFS est " ", un simple espace

Exemple :

awk 'BEGIN{OFS=":";}' - permet d'afficher le séparateur : entre deux champs en sortie

Les différents séparateurs et leurs fonctions

RS : Record Separator

Permet de définir un séparateur qui déterminera le passage d'une ligne à l'autre, différent de celui par défaut (qui est "\n"), dans le texte donné *en entrée*

Exemple :

awk 'BEGIN{RS=":";}' - permet de choisir : comme retour chariot sur le texte donné en entrée

Les différents séparateurs et leurs fonctions

ORS : Output Record Separator

Séparateur qui permet d'afficher un ou plusieurs caractère(s) entre les lignes en sortie
La valeur par défaut du séparateur ORS est " \n"

Exemple :

awk 'BEGIN{ORS=":";}' - permet de choisir : comme séparateur de ligne en sortie

Les différents séparateurs et leurs fonctions

SUBSEP

Séparateur d'indiçage dans les tableaux multidimensionnels

Transforme les tableaux multidimensionnels en tableaux à simple dimension

S'utilise dans un script, avec des fonctions (split, boucles for ...)

Valeur par défaut : *\034*

Exemples d’utilisation

Exemples d’utilisation

FS :

Fichier : “a1=b2=c3”

awk -F"=" '{print $1,$3}' Fichier

Résultat : a1 c3

awk 'BEGIN {FS="="} {print $1,$3}' Fichier

Résultat : a1 c3

Exemples d’utilisation

FS :

Fichier : /etc/passwd

root:x:0:0:root:/root:/bin/bash

awk -F":" '{print $1,$3,$5}' Fichier

Résultat : root 0 root (uniquement la première ligne)

awk 'BEGIN {FS=":"} {print $1,$3,$5}' Fichier

Résultat : root 0 root (uniquement la première ligne)

Exemples d’utilisation

OFS :

Fichier : “a1 b2 c3”

cat Fichier | awk 'BEGIN {OFS=”_"} {print $1, $2, $3}'

Résultat : a1_b2_c3 (uniquement la première ligne)

Exemples d’utilisation

OFS :

Fichier :

root x 0 0 root /root /bin/bash

cat Fichier | awk 'BEGIN {OFS=”="} {print $1, $2, $3}'

Résultat : root=x=0 (uniquement la première ligne)

Exemples d’utilisation

RS :

Fichier : “a1 a2,b1 b2,c1 c2”

```
awk 'BEGIN{RS=","} {print $1}' Fichier
```

Résultat :

```
a1
b1
c1
```

Exemples d’utilisation

RS :

Fichier :

16/02/2000 27/08/1680 | 15/05/1953 14/06/2005 06/05/2004 |

14/07/1224 25/12/1819 30/01/1519 |

```
awk 'BEGIN{RS="|"} {print $1,$3}' Fichier
```

Résultat :

```
16/02/2000
15/02/1953 06/05/2004
14/07/1224 30/01/1519
```

Exemples d’utilisation

ORS :

Fichier :

a1 a2 a3
b1 b2 b3
c1 c2 c3

cat Fichier | awk 'BEGIN {ORS=”_"} {print $1, $2}'

Résultat : a1 a2_b1 b2_c1 c2_

Exemples d’utilisation

ORS :

Fichier :

001476	034	0554	0456
2065	0126	05314	056
1476	564	646	41897

```
cat Fichier | awk 'BEGIN {ORS=”=''} {print $3, $4}'
```

Résultat : 0554 0456=05314 056=646 41897=

Exemples d’utilisation

SUBSEP :

Entrée : SUBSEP=":"

a="4", b="10"

tab[a,b]="MontDeMarsan"

print tab[4:10]

Sortie :

MontDeMarsan

Autres variables et conclusion

Autres variables

NR : compteur donnant le nombre total d'entrée dans l'instance AWK en cours

```
cat > Fichier1
a
b
c

cat > Fichier2
d
e
f
```

```
awk '{print NR}' Fichier1 Fichier2
1
2
3
4
5
6
```

Autres variables

FNR : compteur donnant le nombre total d'entrée dans les fichiers en cours de l'instance AWK en cours

```
cat > Fichier1
a
b
c

cat > Fichier2
d
e
f
```

```
awk '{print FNR}' Fichier1 Fichier2
1
2
3
1
2
3
```

Autres variables

NF : compteur donnant le nombre de champ dans l'instance AWK en cours

```
cat > Fichier1
a1 a2
b1 b2 b3 b4
c1
```

```
awk '{print NF}' Fichier1
2
4
1
```

Pour conclure

- Facilite la recherche
- Permet de faire des traitements complexes sur des fichiers
- Permet un meilleur affichage des résultats
- Facile à utiliser
- Facile à comprendre (excepté subsep)