POWERSHELL –
ARGUMENTS

2016 Automatiser des procédures ICT - 122

Arguments d'un script

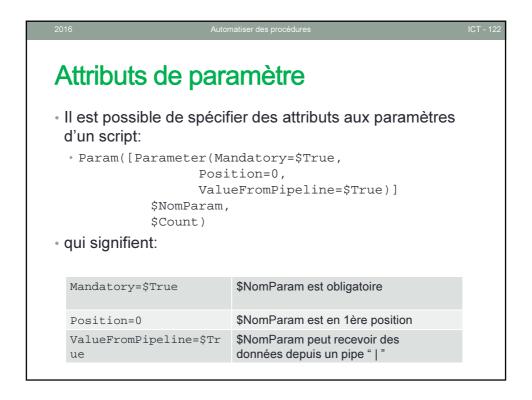
- \$Args est un tableau de chaînes de caractères
- Par exemple:

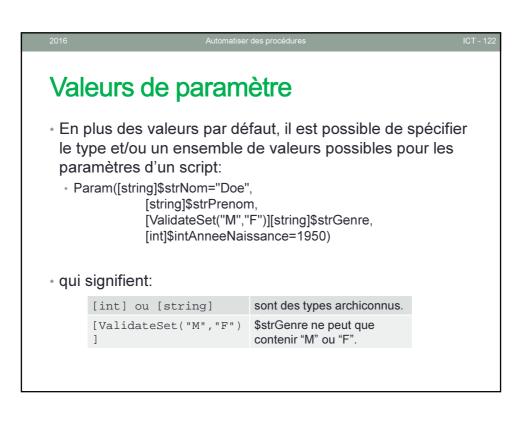
```
PS > args.ps1 -argument1 2 3
```

 Dans le script args.ps1 on accède aux arguments de la manière suivante:

```
$Args.Length => 4
$Args[0] => -argument1
$Args[1] => 2
$Args[1] => 2
```

- Le caractère [espace] sépare les arguments entre eux lors de l'appel du script.
- Tout script qui reçoit des arguments en dispose librement dans \$Args.





2016 Automatiser des procédures ICT - 122

Création et appel de fonction

- Le PowerShell est à l'image des autres langages de programmation en ce qui concerne le mécanisme des fonctions.
- Voici la syntaxe de base d'une fonction:

```
Function <nom de fonction>
{
     Param(<liste des paramètres>)
     #bloc d'instuction
}
```

2016 Automatiser des procédures ICT - 122

Paramètres et retour de fonction

- Ce sont probablement les deux notions les plus simples à inculquer:
- 1) Paramètres de fonction

Tout ce qui s'applique aux paramètres de script (\$Args, \$Param) s'applique à l'identique aux paramètres de fonction.

· 2) Retour de fonction

Une fonction retourne tout objet émis. Dès lors, il suffit d'insérer le code d'émission en fin de fonction pour que celle-ci retourne l'objet à l'appelant. Exemple:

```
Function Display-Message()
{
    Param([string]$strMsg="Aide sans intérêt")
    $strMsg # Retourne le contenu de
    # $strMsg à l'appelant
}
```