Université Toulouse III – Paul Sabatier
L2 Informatique

**Système 1**
*6 janvier 2013*
Semestre 3

# Projet — Gestion de photos à partir de tags

Antoine de ROQUEMAUREL (Groupe 1.1)

## 1 Scripts Simples

### 1.1 `Date_prise_de_vue.sh [-o] fichier_photo [fichier_photo ...]`

Ce script assez simple à permis de prendre en main la coommande `exiv2`, c'est également avec ce script que des scripts utiles et simples ont été crées, ceux-ci sont utilisés dans plusieurs scripts du projet. Cf section 2.

Pour obtenir la date de prise de vue d'une photo, il faut utiliser le tag `Exif.Photo.DateTimeOriginal`, la commande `exiv2` possède un argument qui permet d'afficher les informations concernant un tag, ainsi à l'aide de la commande `exiv2 photo -g Exif.Photo.DateTimeOriginal` nous possédons toutes les informations de prises de vue, ensuite à l'aide d'un cut nous pouvons en retirer les informations nécessaires.

Étant donné que nous pouvons choisir de donner plusieurs photos en paramètre, la commande ci-dessus est placé dans un `for`.

```
#[19:27]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ ./Date_prise_de_vue.sh photos/*.JPG

photos/Moulin_des_pres.JPG: 2012:10:28 09:26:32
photos/Transilien.JPG: 2012:10:29 18:02:20
photos/Fort_Foch.JPG: 2012:11:02 11:46:08
photos/Maison_jaune.JPG: 2012:11:02 12:16:25
photos/Historique_du chateau.JPG: 2012:11:03 14:50:42
photos/Panneaux_Randonnees.JPG: 2012:11:03 14:55:41
photos/Pont_du_chateau.JPG: 2012:11:03 14:59:36
photos/Plan_3D.JPG: 2012:11:03 15:16:20
photos/Arbres.JPG: 2012:11:03 15:17:17
photos/Vue_du_donjon.JPG: 2012:11:03 15:19:48
photos/Fosse.JPG: 2012:11:03 15:26:05
photos/Souvenirs.JPG: 2012:11:03 16:02:27
photos/Les Mam'zelles.JPG: 2012:11:03 16:05:14
```

Listing 1 – Exemple d'utilisation du script `Date_prise_de_vue.sh`

### 1.2 `Met_date_dans_nom.sh [-a] fichier_photo`

Ce script nécessite le script précédent, en effet afin de mettre la date dans le nom, je récupère la date à l'aide de `Date_prise_de_vue.sh`.

Ce script peut prendre en paramètre plusieurs photos bien que ça ne soit pas spécifié dans le synopsis par la présence de ..., en effet, il était très simple à mettre en oeuvre la présence de plusieurs paramètre (un simple `for`), et cela peut sembler très pratique.

```
#[21:06]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ ./Met_date_dans_nom.sh -a photos/Arbres.JPG photos/Maison_jaune.JPG
photos/Arbres.JPG -> photos/Arbres.2012-11-03_15.17.17.JPG
photos/Maison_jaune.JPG -> photos/Maison_jaune.2012-11-02_12.16.25.JPG

#[21:06]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ ./Met_date_dans_nom.sh photos/Arbres.JPG photos/Maison_jaune.JPG

#[21:06]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ ls photos/
Arbres.2012-11-03_15.17.17.JPG  Fort_Foch.JPG  Historique_du chateau.JPG
Maison_jaune.2012-11-02_12.16.25.JPG  Panneaux_Randonnees.JPG  Pont_du_chateau.JPG
Transilien.JPG
Date_prise_de_vue.sh            Fosse.JPG      Les Mam'zelles.JPG         Moulin_des_pres
    .JPG
Plan_3D.JPG               Souvenirs.JPG        Vue_du_donjon.JPG
```

Listing 2 – Exemple d'utilisation du script `Met_date_dans_nom.sh`

## 1.3 `Change_description.sh fichier_photo description`

Contrairement aux autres scripts, celui-ci ne prend qu'un seul paramètre : La description et la photo pour laquelle il faut changer la description.

Afin de pouvoir changer la description d'une image, j'ai recherché dans le man d'exiv2 et ai trouvé la commande suivante qui faisait ce dont j'avais besoin :

`exiv2 -M"set Exif.Image.ImageDescription la nouvelle description" image.jpg`

```
#[14:25]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ exiv2 photos/Arbres.JPG -g Exif.Image.ImageDescription

#[14:25]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ ./Change_description.sh photos/Arbres.JPG "une super description"

#[14:25]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ exiv2 photos/Arbres.JPG -g Exif.Image.ImageDescription
Exif.Image.ImageDescription                  Ascii      22  une super description
```

Listing 3 – Exemple d'utilisation du script `Change_description.sh`

## 1.4 `Change_date_modif.sh fichier_photo`

Pour ce script, je me suis permis d'avoir la possibilité de passer en paramètre une liste de photos et non une unique photo, en effet, cela était très simple à mettre en œuvre [1] et cela peut être extrêmement pratique de faire passer `photo/*` par exemple.

Afin de modifier la date de modification d'un fichier, des recherches Google m'ont permis de découvrir que la commande `touch` possédait un argument qui permettait de modifier celle-ci : `touch -t yyyyMMJJhhmm[.ss]` modifie la date de modification au `jj/MM/yyyy` à `hh:mm:ss`

Ce script nécessite le script `Date_prise_de_vue.sh`, en effet celui-ci est utilisé pour obtenir la date de prise de vue.

1. un simple `for`

# 2 Scripts utiles

Afin de limiter la redondance au maximum, j'ai créer trois scripts qui sont appelés dans chacun de scripts, ceux-ci sont des scripts de base de vérifications.

## 2.1 `haveArgument.sh`

Ce script vérifie que le premier paramètre est présent dans la liste suivante de paramètre, ce script est utile afin de vérifier que les options sont présentes, ou absente d'un appel de script.

Il est toujours utilisé de la façon suivante :

```
var=1
if ‘util/./haveArgument.sh -o $*‘
  var=0
fi
```

Listing 4 – Exemple d'utilisation du script `haveArgument.sh`

La variable `var` valant 0 si l'argument est présent et 1 sinon.

## 2.2 `isAccessibleDirectory.sh`

Ce script comme son nom l'indique sert à vérifier qu'un répertoire est accessible en lecture, dans le cas d'une accessibilité en écriture, je rajoute un test manuellement.

## 2.3 `isAccessibleFile.sh`

Ce script comme son nom l'indique sert à vérifier qu'un fichier est accessible en lecture, dans le cas d'une accessibilité en écriture, je rajoute un test manuellement.

## 2.4 `isImage.sh`

Ce script sert à tester si un fichier est bien une image, pour cela j'utilise la commande `file`, si le fichier est une image un `grep "image"` devrait retourner 0.

Il est ainsi appelé dans tous les scripts afin d'éviter des erreurs de la part d'exiv2.

# 3 Scripts complexes

## 3.1 `Range_selon_date_et_description.sh [-c | -a] fichier_photo [répertoire]`

Cette fonction comporte un certain nombre de sous-programme afin de simplifier sa compréhension.

#### 3.1.1 haveGoodNbArguments

Ce sous-programme sert simple à vérifier que le nombre d'argument du programme est correct, ceci à l'aide du script `util/haveArgument.sh`.

#### 3.1.2 replaceAlphanumericByUnderscore

Cette fonction remplace les caractères alphanumériques par des underscore (_) à l'aide du programme `sed`.

#### 3.1.3 directoryArgumentExistAndIsWrittable

Ce sous-programme vérifie que le répertoire passé en argument existe et est accessible en écriture.

#### 3.1.4 getFileAndDirectory

Ce script permet d'obtenir le répertoire de départ et le nom de la photo à déplacer, ceci en fonction des options présentes ou non, en effet, en fonction de ces options, les numéros d'arguments peuvent changer.

Si le répertoire n'est pas spécifié dans les arguments, celui-ci est `./`.

#### 3.1.5 getGoodPartOfDate

Avec `getGoodPartOfDate` nous pouvons obtenir la partie de la date de prise de vue de la variable `file` [2] que nous voulons en fonction de l'argument : 1 nous retourne l'année, 2 nous retourne le mois et 3 nous retourne le jour.

#### 3.1.6 getDescription

Cette fonction nous permet d'obtenir la description de `file`, la description ayant ses caractères alphanumériques remplacés par des underscore à l'aide du script `replaceAlphanumericByUnderscore`.

#### 3.1.7 getDirectoriesPath

Cette fonction nous retourne le chemin des répertoires à créer, ainsi celui-ci sera affiché si l'option `-a` est spécifié, et sera utilisé avec un `mkdir -p` dans les autres cas.

```
#[15:40]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ ./Range_selon_date_et_description.sh photos/Arbres.JPG ; ls -R 2012/
2012/:
11/

2012/11:
Wangenbourg/
```

2. Celle-ci est obtenur après le lancement du script `getFileAndDirectory`

```
2012/11/Wangenbourg:
Arbres.JPG
```

Listing 5 – Exemple d'utilisation du script `Date_prise_de_vue.sh`

## 3.2 `Range_Mes_Photos.sh [-d répertoire] fichier_photo [fichier_photo ...]`

Ce script utilise simplement le script précédent `Range_selond_date_et_description.sh` dans un `for` qui parcours la liste des arguments.

```
#[18:38]aroquemaurel@Garp:~/cours/L2/systeme3/gestionPhotos
$ ./Range_Mes_Photos.sh -d dest/ photos/* ; ls -R dest/
dest/:
2012/  chateau.JPG/  Mam'zelles.JPG/

dest/2012:
10/  11/

dest/2012/10:
Montalet_le_Bois/  Transilien.JPG*

dest/2012/10/Montalet_le_Bois:
Moulin_des_pres.JPG*

dest/2012/11:
Niederhausbergen/  Wangenbourg/

dest/2012/11/Niederhausbergen:
Fort_Foch.JPG*  Maison_jaune.JPG*

dest/2012/11/Wangenbourg:
Arbres.JPG*  Fosse.JPG*  Panneaux_Randonnees.JPG*  Plan_3D.JPG*  Pont_du_chateau.JPG*
    Souvenirs.JPG*  Vue_du_donjon.JPG*

dest/chateau.JPG:
2012/

dest/chateau.JPG/2012:
Historique_du chateau.JPG*

dest/Mam'zelles.JPG:
2012/

dest/Mam'zelles.JPG/2012:
Wangenbourg/

dest/Mam'zelles.JPG/2012/Wangenbourg:
Les Mam'zelles.JPG*
```

Listing 6 – Exemple d'utilisation du script `Date_prise_de_vue.sh`