

# Tp 5

## Mise à jour de Google calendar

Florent DEVIN

### 1 Présentation

L'objectif de ce TP est de mettre à jour votre agenda *Google*, attaché à votre compte `eisti.eu`. Même si il existe une autre façon d'importer des événements dans *Google Calendar*, nous allons ici utiliser une approche *originale*. Dans un premier temps vous aurez besoin d'installer *Google Command Line*, téléchargeable à l'adresse suivante : <http://code.google.com/p/googlecl/>

Deux choix possibles pour installer ce logiciel :

- Le paquet Debian (`.deb`). Dans ces conditions l'installation se fera via la ligne de commande `dpkg -i paquet`. Cette ligne de commande est à entrer en tant que *root*.
- L'archive en `tar.gz`. Dans ces conditions, il vous faudra décompresser l'archive (`tar xvzf archive`), puis installer le programme (lisez le fichier `INSTALL`). Normalement la ligne de commande pour installer le programme, à taper en tant que *root*, est : `python setup.py install`.

Dans tous les cas, cela suppose que python soit installé sur votre système. Pour nous permettre de réaliser correctement le TP, il faudra aussi se documenter sur la commande `sed`, et aussi configurer correctement votre agenda.

Lorsque vous vous connectez à votre mail EISTI, vous avez dans la ligne du haut de la page web, un lien vers *Agenda*. Cliquez sur ce lien, puis cliquez sur le lien *Paramètres*, et choisissez l'option *Paramètres de Agenda*. Dans cette page, choisissez comme langue : **Français**, comme format de date : **31/12/2010**, et comme format d'heure : **13:00**. Voilà notre environnement correctement configuré...

Enfin, connectez vous sur AREL, et téléchargez votre planning en CSV (petite icône CSV sur la page planning). Normalement ce fichier ressemble à celui du listing 1, ou du listing 2.

Listing 1: *Planning CSV*

```
Subject, Start Date, Start Time, End Time, End Date
Ing1 : B - Projet infodif (204),13/12/2010,14:00,16:30,13/12/2010
Ing1 : AB - Systeme exploitation (amphi),15/12/2010,08:00,09:00,15/12/2010
Ing3 : A - Evolut syst nomades (106),13/12/2010,09:30,12:00,13/12/2010
Ing1 : B - Systeme exploitation (202),15/12/2010,10:55,12:25,15/12/2010
Ing1 : A - Systeme exploitation (308),15/12/2010,15:30,17:00,15/12/2010
Ing1 : B - Projet infodif (204),15/12/2010,17:15,18:45,15/12/2010
Ing1 : B - Projet infodif (202),16/12/2010,08:00,09:30,16/12/2010
```

Listing 2: *Planning CSV 2*

```
Subject, Start Date, Start Time, End Time, End Date
0 : AB - Lv 1 (103),06/12/2010,16:15,18:15,06/12/2010
0 : AB - Lv 1 (103),06/12/2010,14:00,16:00,06/12/2010
0 : A - Technologies emergentes (106),07/12/2010,14:00,16:30,07/12/2010
0 : A - Evolut syst nomades (106),06/12/2010,09:30,12:00,06/12/2010
0 : A - Isin-evolut syst nomades (106),10/12/2010,09:30,12:00,10/12/2010
0 : A - Isin-web service (106),10/12/2010,14:00,16:30,10/12/2010
0 : A - Securite (106),07/12/2010,09:30,12:00,07/12/2010
0 : A - Isin-developpement agile (106),08/12/2010,14:00,17:00,08/12/2010
0 : A - Isin-developpement agile (),08/12/2010,09:30,12:30,08/12/2010
0 : A - ??? (106),09/12/2010,14:00,17:00,09/12/2010
0 : AB - ??? (amphi),09/12/2010,10:00,12:00,09/12/2010
```

## 2 Préliminaire

Écrire un script shell qui transforme le fichier `csv`, en un fichier exploitable par `google command line`. Chaque ligne devra contenir les informations nécessaires à la création d'un créneau dans votre agenda. Une ligne correcte à la forme suivante :

Listing 3: *Forme générale d'un créneau*

```
Objet heureDebut - heureFin Date
```

Si on applique cela sur le listing 1, on obtient le listing 4.

### 2.1 Algorithme

Dans un premier temps, transformez tous les caractères de la classe de caractères `[[:blank:]]` par un espace. Puis transformez toutes les suites d'espaces par un espace.

Avant de commencer, il vous faudra supprimer la première ligne du fichier `csv`. Une fois ceci fait, il faut supprimer la première date de chaque ligne, et transformer toutes les `,` par des espaces, sauf la virgule qui sépare les deux heures. Cette dernière il faut la remplacer par un `-`. Pour ce faire la commande `sed` peut être d'une grande aide.

Enfin, pour chaque ligne, lancez la commande `google calendar add "evt"`.

#### Listing 4: *Fichier pour l'importation de créneau*

```
Ingl : B - Projet infodif (204) 14:00 - 16:30 13/12/2010
Ingl : AB - Systeme exploitation (amphi) 08:00 - 09:00 15/12/2010
Ing3 : A - Evolut syst nomades (106) 09:30 - 12:00 13/12/2010
Ingl : B - Systeme exploitation (202) 10:55 - 12:25 15/12/2010
Ingl : A - Systeme exploitation (308) 15:30 - 17:00 15/12/2010
Ingl : B - Projet infodif (204) 17:15 - 18:45 15/12/2010
Ingl : B - Projet infodif (202) 08:00 - 09:30 16/12/2010
```

La première fois que vous lancerez cette commande, vous devrez entrer votre `login@eisti.eu` qui vous permet d'accéder aux mails de l'EISTI. Une fois entré, un navigateur web se lance, avec une page web dans laquelle, vous devrez permettre l'accès. Puis si le navigateur web vous renvoie un token, il faudra copier/coller le token dans le shell pour autoriser le script à insérer des événements dans votre calendrier.

Si vous vous trompez dans les entrées, vous pouvez effacer le fichier `~/config/googlecl`, et recommencez l'ensemble des saisies.

Pour vous permettre d'utiliser *Google Command Line* sans problème vous utiliserez la ligne suivante :

```
echo "'google calendar add \"ligne\"'"
```

Cette commande peut mettre du temps à répondre. Vous pourriez peut être indiquer à l'utilisateur que vous êtes en cours d'insertion d'un créneau. Avant de réaliser le script, faites des essais, en lançant les commandes manuellement.

## 2.2 Explication rapide de sed

La commande `sed`, vue dans le TP précédent, est une commande qui permet de remplacer une chaîne de caractère, par une autre. Il y a possibilité de définir des groupes de chaînes lors de la recherche, pour les rappeler lors du remplacement. Pour expliquer cette commande, considérons le listing 5.

#### Listing 5: *Exemple : fichier*

```
asterix ; 20
troll de troy ; 8
rubrique à brac ; 4
Lanfeust de troy ; 8
autre / diver ; 30
```

La commande `sed -e 's/a/z/g' fichier` change tous les a du fichier en z, et affiche le résultat sur la sortie standard.

Nous pouvons définir des groupes de caractères dans `sed`. Cette définition se fait par l'utilisation de `\(` qui marque le début d'un groupe, et `\)` qui marque la fin d'un groupe. Par exemple, la commande `sed -e 's/\(.*\) ; \(.*\)\/\2 ; \1/g' fichier` définit deux groupes séparés par  `;` . C'est à dire si l'on reprend le fichier original, les titres des

BD, ainsi que leur nombre. L'utilisation de \1 permet de spécifier que l'on veut rappeler le premier groupe défini dans la recherche, c'est à dire les BD. Cette commande a donc pour effet d'inverser le nombre et les titres.

## 3 Utilisation du script

### 3.1 Modification

Modifier le script précédent pour qu'il traite la ligne de commande. Les options que vous aurez à traiter sont :

- **-h**, **--help** : Afficher de l'aide sur le script
- **-f** : Traiter un fichier spécifique
- **-d** : Traiter un répertoire particulier.

Voici l'usage de votre script : **agenda [OPTIONS]**. Ce qui signifie que votre script doit pouvoir être lancé aussi sans option. Dans ce cas, vous considérerez que votre script travaille sur un répertoire par défaut (celui que vous souhaitez).

Lorsque vous traitez un répertoire, vous isolez dans ce répertoire l'ensemble des fichiers portant l'extension **csv**. Une fois les fichiers traités, vous renommez chaque fichier en **csv.old**.

### 3.2 Intégration dans Unix

Le script que vous venez d'écrire est un script qui peut être utile. Cependant, cela peut devenir pénible de lancer se script dès que l'on vient de télécharger un fichier **csv**. Pour éviter d'avoir à lancer un script manuellement, nous avons plusieurs possibilités. Nous allons en étudier deux.

#### 3.2.1 Déclenchement à heure fixe

Pour cette partie, nous allons présupposer que vous restez connecté plus d'une heure consécutive sous Unix. Nous allons donc faire exécuter ce script toute les heures. Unix possède cette fonctionnalité intéressante. Pour déclencher un script à heure précise, ou jour, mois, année, ou répéter un script toutes les  $x$  minutes, heures, ... Vous avez accès à cette fonctionnalité avec la commande **crontab -e**. Il est à noter qu'il existe des *crontabs* par chaque utilisateur. Certains systèmes demande le redémarrage du *cron* lors de la modification d'une **crontab** afin que les modifications soient prises en compte. Voici le format général d'un fichier de type *crontab*

Listing 6: *Forme générale d'un cron*

minute	heure	jour	mois	jourdelasemaine	commande
--------	-------	------	------	-----------------	----------

Pour chacune des valeurs, vous pouvez spécifier :

- `*` : indique que toutes les valeurs sont valides
- `n-m` : de  $n$  à  $m$
- `n,m,l,k` : les valeurs  $n$ ,  $m$ ,  $l$ , et  $k$
- `n-m/k` : toutes les  $k$  valeurs dans l'intervalle  $[n - m]$
- `*/n` : toutes les  $n$  valeurs

Donc pour exécuter le script `/home/toto/bin/save` tous les jours à 23h59, on écrirait :

Listing 7: *Exemple de cron*

```
59 23 * * * /home/toto/bin/save
```

Il y a, au minimum, deux façons de modifier, les *crontabs* soit directement dans le répertoire `/var/spool/cron/crontabs`, soit en utilisant la commande `crontab -e`. Si vous éditez manuellement la *crontab*, attention à respecter les normes utilisées par `cron`. En tout cas, n'intervenez pas directement sur les fichiers du répertoire `/etc`

Mettez en place un *cron* pour qu'il déclenche le script d'exportation dans *Google* toutes les heures.

### 3.2.2 Déclenchement lors d'événement

Il existe un outil qui permet de scanner des fichiers, afin d'être averti d'un changement. Cet outil s'appelle `inotify`. Il vous faudra donc installer le paquetage `inotify-tools`. L'installation de ce paquetage vous donne accès à deux commandes `inotifywait`, et `inotifywatch`. Pour vous permettre de lancer le script, vous allez devoir utiliser la commande `inotifywait`.

Utilisez cette commande pour mettre en place un déclenchement à événement. Pour vous aider, voici une partie de commande qui vous sera utile :

Listing 8: *Début d'aide pour le déclenchement*

```
inotifywait -qm -e create /home/toto/.agenda --format %f
```