Ecrire un programme à plusieurs

Xavier Dupré
http://www.xavierdupre.fr/

1er février 2013

Résumé

Deux pages et quelques idées pour travailler à plusieurs sur le même projet de programmation.

1 Trois conseils pour écrire un programme

1.1 Des petites fonctions

Pour plusieurs raisons :

1. Il est plus facile de corriger un programme qui est constitué de petites fonctions plutôt que de quelques grandes. Chaque fonction peut être vérifiée séparément.
2. Il est plus facile de réutiliser des petites fonctions.
3. Il est plus facile de répartir le travail sur plusieurs personnes.

Remarque 0.1 : variables globales
Il vaut mieux éviter les variables globales qui sont considérées que comme des paramètres cachés.

1.2 Séparer les calculs, le chargement des données, l'interface graphique

Pour plusieurs raisons :

1. Il est plus facile de vérifier un calcul s'il est dans une fonction indépendante plutôt que caché dans le code d'une interface graphique.
2. C'est facile de faire un calcul une fois lorsqu'un utilisateur appuie sur un bouton, si on veut faire ce calcul cent fois, on ne peut pas lui demander d'appuyer cent fois sur le même bouton.
3. Les calculs ou le chargement des données peuvent être utilisés dans d'autres programmes.

1.3 Utiliser des fonctions de tests

Ces fonctions peuvent être exécutées au début du programme pour vérifier que certaines parties du programme fonctionnent toujours même après les avoir modifiées.

L'exemple suivant considère une fonction qui doit retourner une somme réelle même si les éléments de la liste sont entiers. On écrit la fonction qui vérifie cela.

```
def somme_double (liste) :
    return 1.0 * sum(liste)

def test_somme_double () :
    y = somme_double([ 1 ]) / 2
    if y == 0 : raise Exception ("valeur > 0 attendue")

if __name__ == "__main__" :
    test_somme_double()
```

Si plus tard, quelqu'un modifie la fonction `somme_double` en enlevant la multiplication parce qu'il considère cela inutile. La fonction de test provoquera une erreur. Elle est là pour rappeler que la fonction a été programmée pour retourner un nombre réel et que quiconque l'utilise s'attend à ce qu'elle retourne ce type de résultat.

```
Traceback (most recent call last):
  File "conseil.py", line 10, in <module>
    test_somme_double()
  File "conseil.py", line 7, in test_somme_double
    if y == 0 : raise Exception ("valeur > 0 attendue")
Exception: valeur > 0 attendue
```

2 Trucs et astuces

2.1 Partager du code

Il existe aujourd'hui des solutions qui permettent d'éviter les envois de programme par email. Des outil comme *DropBox*, *SkyDrive*, *GoogleDrive* permettent de partager un répertoire. Un même répertoire peut être partagé sur plusieurs ordinateurs et plusieurs personnes. Une modification (y compris une suppression) sur l'une des répliques sera propagée sur tous les ordinateurs dès qu'ils sont connectés à Internet.

Il est possible de coupler cette solution avec *SVN* ou *TortoiseSVN* qui sont des logiciels de suivis de source. On garde à la fois la dernière version et l'historique des modifications [1].

2.2 Moteurs de recherche

Lorsqu'on ne comprend un message d'erreur, il est souvent utile de recopier le texte dans un moteur de recherche (Google, Bing, ...). Il est très rare de ne pas réussir à trouver d'indices.

1. http://mlevit.wordpress.com/2009/11/18/how-to-use-subversion-dropbox-to-create-an-effective-work-managementbackup-system/