Master 1 TIIL Algorithmique et Programmation Objet - TP 1

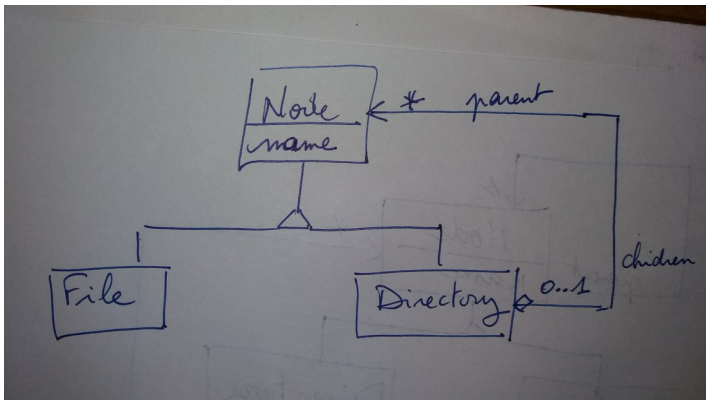
Alain Plantec

Vous utiliserez Eclipse pour développer.

Ne programmez pas de fonctions *main*, Il vous est demandé de produire des tests en utilisant *Junit*. Voir par exemple http://gfx.developpez.com/tutoriel/java/junit/

Modélisation d'une arborescence de fichiers

Voici une modélisation d'un arbre représentant une arborescence de fichiers :



Node est la superclasse abstraite. Tout *Node* a un nom relatif et un ou 0 *Node* parent. *File* et *Directory* sont deux sous-classes de *Node*. Un *Directory* se compose de 0 ou plusieurs *Node*.

Ecrire le code Java pour représenter des arborescences en utilisant ce schéma et produire des tests pour votre mise en œuvre. Vous utiliserez la classe standard *ArrayList* pour stocker le contenu d'un répertoire (attribut *children*)

Recherche dans une arborescence

Comme nous l'avons vu en cours, une recherche dans une collection peut être implantée en utilisant l'abstraction *Critère*. Ce mécanisme permet tout type de recherche en mettant en œuvre le critère de recherche par un critère concret. Ce pattern peut s'appliquer à notre modèle d'arborescence.

Utiliser ce mécanisme pour implanter deux recherches : une première pour retrouver tous les nœuds ayant le même nom local et une seconde pour récupérer tous les fichiers d'une arborescence.

Validez votre mise en œuvre à l'aide de tests.

Construction d'un résultat complexe

Comme nous l'avons vu en cours, pour construire une transformation, il est possible d'utiliser une fonction de parcours prenant en argument un objet représentant une action à effectuer sur chaque nœud.

Utiliser ce pattern pour afficher une arborescence sur la sortie standard. L'affichage utilisera des tabulations pour représenter la profondeur. L'affichage pourra être paramétré pour trois options à l'aide d'une énumération :

1. Afficher les répertoires d'abord
2. Afficher le contenu par ordre alphabétique
3. Afficher uniquement les répertoires

Par exemple, avec l'option 1, voici l'affichage attendu pour un répertoire racine avec deux nœuds *f* et *D*. *f* est un fichier et *D* un répertoire contenant deux fichiers *a* et *x* :

```
/
	D
		a
		x
	f
```