Le patron *Composite*

Exercice 1 – *Codage d'un composite.*

Un document peut être vu comme un ensemble d'éléments imbriqués (représenté par un arbre, cf XML) sur lesquels on peut appliquer un certain nombre d'opérations comme sur tout ou partie du document. On propose d'écrire une classe pour représenter chaque type d'éléments du document :

- *Paragraphe* : représente le texte d'un paragraphe.
- *Figure* : représente une figure (on pourra prendre un nom de fichier externe fictif).
- *Liste* : représente une liste pucée.

sur lesquels pourront être appliquées les opérations suivantes :

- Appliquer un style (ce style pourra provenir d'un type énuméré : { Gras, Italique, Souligne}). Note, tous les éléments ne sont pas sensibles à cette action.
- Afficher : donne une représentation texte de l'élément (on pourra préfixer par le style par exemple).
- Écrire de la même manière une fonction qui sérialise dans un fichier le document sous forme de flux HTML.

Bien sur, il n'y aurait pas de patron composite si on ne prévoyait pas un élément *Section* capable de regrouper tout type d'éléments, y compris des sections. Nous retrouvons la structure d'arbre mentionnée plus haut : les éléments de base sont les feuilles, les sections sont les nœuds internes. Chaque section disposera d'un titre et d'une liste d'éléments associés et on pourra :

- Afficher : affiche le titre de la section suivi de tous les éléments la composant. On souhaite que chaque section soit numérotée en fonction de sa profondeur et de sa position dans la section courante, ex : 1 - Introduction, 2.1.3 - Conclusion, ...).
- Ajouter un élément à la fin de la section.
- Retourner et supprimer le dernier élément de la section.

Question 1.1 : Rappelez le diagramme UML de principe du patron.

Question 1.2 : Donnez le diagramme UML de votre solution (en identifiant les acteurs).

Pour simplifier, on utilisera une approche récursive où les composites délégueront à leurs composants les traitements à réaliser.

Question 1.3 : Rédigez précisément le contenu de l'interface commune aux feuilles et aux nœuds. Pensez bien au mécanisme que vous emploierez pour la numérotation !

Question 1.4 : Écrivez les classes d'éléments de bases. Les classes sur lesquelles s'applique la mise en forme pourrait bénéficier d'attributs et de code communs. Cela ne doit cependant pas être écrit dans l'interface ! Testez ces classes de façon unitaire.

Question 1.5 : Écrivez la classe *Section* puis testez la en l'imbriquant avec des éléments de base en faisant un arbre de taille raisonnable. Affichez le résultat en vérifiant que la numérotation fonctionne. Tester en permutant deux sous-sections par exemple ...

Question 1.6 : Critiquez l'approche récursive que nous avons choisie. Est-il possible d'écrire un petit programme de test mettant en défaut cette approche ?