Patron composite

a. L'intention du patron composite est de pouvoir traiter un groupe d'objets composés récursivement de manière uniforme. Dans le cadre du logiciel produit pour ce travail pratique, utiliser ce patron de conception nous permet d'effectuer des macros.
b. Voir DiagrammeDeClasse_Composite.pdf
c. La classe qui est responsable de l'arbre des composantes est la classe CompositeTransform avec la méthode addChild. C'est ainsi qu'une structure de données utilisant le patron composite structure ses données, chaque composite a dans ses paramètres un ensemble d'objets héritant d'une classe abstraite commune pouvant être de sous-composites : il est donc logique que ce soit la classe composite qui s'occupe de se lier à ses enfants.

Patron proxy

a. L'intention du patron proxy est de fournir une doublure pour un autre objet afin de contrôler l'accès à ce dernier. Cette doublure sert d'intermédiaire entre le client et la classe doublée. Dans le cadre du logiciel produit pour ce travail pratique, utiliser ce patron nous permet de rendre le logiciel plus efficace en limitant l'accès aux fichiers binaires seulement aux phases de démarrage et de terminaison du programme. Puisque le proxy (MemAudioFile) est implémenté par une variable locale (dans la mémoire vive), l'accès et la modification de celui-ci pendant l'éxécution du programme sont beaucoup plus rapide que si on utilise directment sujetRéel (AudioFile, qui écrit et lit directement sur le disque).
b. Voir DiagrammeDeClasse_Proxy.pdf
c. Étant donné que la classe MemAudioFile a été créé afin de réduire le nombre d'interactions avec le disque dur pour des fins de performances, les modifications faites à un fichier ne sont faites qu'à la fin de l'exécution du programme. Donc, la réécriture dans le fichier dont l'accès est limité par le proxy et modifié seulement lorsque le destructeur de la classe proxy est appelé. Tout au long de l'exécution du programme, le fichier reste intact. On pourrait donc, avant de détruire l'objet proxy, tester si le fichier dont on limite l'accès est le même qu'au début du programme (en s'assurant qu'il n'a pas été modifié d'aucune autre façon) ou on pourrait le comparer à la nouvelle version que le proxy contient avant de détruire le proxy. Nous n'avons pas besoin d'implémenter une fonction qui surveille si le fichier protégé par le proxy est identique à celui contenu par le proxy suite à la destruction du proxy puisque ce test est déjà présent.