PARIS DESCARTES IUT

DUT Informatique - 1ère année
SDA - Structures de Données et Algorithmes

TRAVAUX PRATIQUES – Semaine n°4

Thèmes

- Structuration de code et compilation séparée
- Composants et donnée concrète

Le but des exercices est de vous familiariser à la structuration du code source de vos applications pour une compilation séparée. D'une part, vous devez structurer une application développée en semaine 3 (exercice 2 du TP) et la spécialiser à un nouveau type de données. D'autre part, vous devez modifier le composant de pile présenté au cours n°4 (pile à capacité fixée) de manière à rendre la pile extensible. Les sources de TP sont sur COMMUN.

Exercice 1. Structuration du code source d'une application

Suivant les principes donnés en cours, vous devez structurer le code de l'application développée à l'exercice 2 du TP de la semaine 3 pour une compilation séparée. L'application consistait à tester un conteneur de dates (de type `ConteneurTDE`, alloué en mémoire dynamique et extensible). L'architecture logicielle retenue pour structurer l'application est la suivante :
(1) le composant de date (`Date.h`, `Date.cpp`), (2) le composant conteneur d'items (`ConteneurTDE.h`, `ConteneurTDE.cpp`), (3) le fichier de spécialisation du type `Item en Date (Item.h`), (4) le programme principal de test du conteneur de dates (`testConteneurTDEDates.cpp`).

Créez un nouveau Projet/Solution de type « `Projet C++` » (« `Sem4-TP` »)

1.1. Composant de dates – test du composant

Créez un nouveau projet « Exo1-1 ». Importez les fichiers du répertoire `Exo1-1` : il s'agit du composant de dates (`Date.h`, `Date.cpp`) et du programme de test du composant (`testDate.cpp`). Le composant de dates a déjà été étudié au cours n°4. Complétez les inclusions dans les fichiers `Date.cpp` et `testDate.cpp`, puis testez le composant.

1.2. Composant conteneur d'items - Structuration et compilation

Créez un nouveau projet « Exo1-2 ». Importez tous les fichiers de `Exo1-2` : (1) le fichier `ConteneurTDE_Sem3.cpp` qui correspond au source de l'exercice 2 du TP semaine 3, (2) le fichier `Item.h` de spécialisation du type `Item` et (3) le fichier `main.cpp` (programme principal vide) utilisé pour générer le projet. A partir du fichier `ConteneurTDE_Sem3.cpp`, structurez le composant de conteneur d'items (`ConteneurTDE.h` et `ConteneurTDE.cpp`). Spécialisez le type `Item` en `int`. Générez le projet.

1.3. Test du conteneur de dates - Structuration et exécution

Créez un nouveau projet « Exo1-3 ». Importez le fichier `ConteneurTDE_Sem3.cpp` de `Exo1-3` et créez un fichier source `testConteneurTDE_Dates.cpp` pour tester le conteneur de dates. A partir de `ConteneurTDE_Sem3.cpp`, recopiez/modifiez, dans le fichier source créé, le code correspondant au test puis supprimez `ConteneurTDE_Sem3.cpp`. Pour éviter le « partage de sources », recopiez au niveau des répertoires tous les fichiers nécessaires des précédents projets. Vous aurez à modifier le fichier `Item.h` pour spécialiser le type `Item` en `Date`. Compilez, exécutez et testez le programme.

Exercice 2. Adaptation à un composant généralisé

L'architecture choisie pour le composant de conteneur d'items permet une adaptation rapide à un changement du type des items. Le but de cet exercice est de tester un conteneur de positions sur une grille. Dans un premier temps, on considère pour le type *Position* (donné ci-dessous) les deux seules fonctions de saisie et d'affichage de position.

```
struct Position {
      unsigned int abscisse; //abscisse de la position
      unsigned int ordonnee; //ordonnée de la position
};
```

2.1. Créez un nouveau projet « Exo2 ». Importez dans le projet tous les fichiers de `Exo2` : les fichiers du composant `Position` (`Position.h` et `Position.cpp`) et le fichier de test `testConteneurTDEPositions.cpp`. Recopiez au niveau des répertoires à partir des projets précédemment testés tous les fichiers nécessaires à l'application.

Adaptez le fichier `Item.h` à l'application. Complétez le code du fichier `testConteneurTDEPositions.cpp` et testez l'application.

Exercice 3. Modification de donnée concrète d'un composant

Le but de cet exercice est de développer un composant de pile d'items extensible contrairement au composant de pile à capacité fixe implémenté au cours n° 4. Par conséquent, vous devez implémenter une structure concrète de la pile utilisant un conteneur de type `ConteneurTDE`. Testez une pile de positions.

3.1. Créez un nouveau projet « Exo3-1 ». Importez de `Exo3` les fichiers du composant de pile étudié au cours n°4 et le fichier `testPilePositions.cpp`. Recopiez au niveau des répertoires, à partir des projets déjà créés, tous les fichiers nécessaires au développement.

3.2. La donnée concrète de `Pile` (structure de stockage de la pile) doit être modifiée dans le fichier d'entête `Pile.h`. Modifier en conséquence le code du fichier source `Pile.cpp`.

3.3. Testez le composant de pile en spécialisant le type `Item` au type `Position` après avoir complété le code du fichier `testPilePositions.cpp`.