ÉCOLE POLYTECH'NICE SOPHIA
SI4
2021-2022

Julien DeAntoni

Programmation orientée objet : CPP

TD numéro 5

le polymorphisme (et éventuellement utiliser un éditeur UML, comprendre le code généré et la documentation)

Objectif

L'objectif de cet exercice est de maîtriser les notions orientées objects en C++. De plus, habituellement nous utilisons une specification UML pour décrire ce qui est attendu du code (en partie). Lors de ce TD, si vous finissez en avance, vous devrez regarder les possibilités de génération de code du logiciel VisualParadigm (que vous utilisez par ailleurs) ou de l'outil boUML (dont la version libre est nommée doUML). Vous pourrez alors vous questionner sur les modifications à réaliser, les choix par défaut utilisés lors de la génération de code et les problèmes que cela engendre. De plus, vous utiliserez ce même outil pour faire de la rétro-ingénierie (ou mieux, round trip engineering) sur votre code afin de pouvoir vous questionner sur la validité de votre solution.

Menu en cascade

On souhaite définir un ensemble de classes afin de réaliser une gestion simple (et même simpliste) de menu. Pour cela on se donne la spécification de la figure 1.
Aucune autre explication de ces classes n'est donnée dans ce document. Essayez de comprendre par vous même et dites-vous que je suis pas très professionnel de ne pas vous avoir donné des exigences claires. Enfin, vous générerez la documentation de votre projet (doxygen est votre ami)

faire la fonction `main`

Le programme principal exécuté ici est une boucle infinie d'appels à la fonction `activate` du `Menu`. On en sort seulement par sélection de "quitter". Noter que l'entrée "quitter" est une entrée par défaut. Dans cet exemple, les caractères entrés par l'utilisateur sont entre guillemets.

```
jdeanton@linux-hani:~> ./test_Menu
  LE MENU
   0- emacs
   1- htop
   2- COMMUNICATIONS ->
   3- quitter
  Votre choix? "0"
     ************** Execution de EMACS
  LE MENU
```

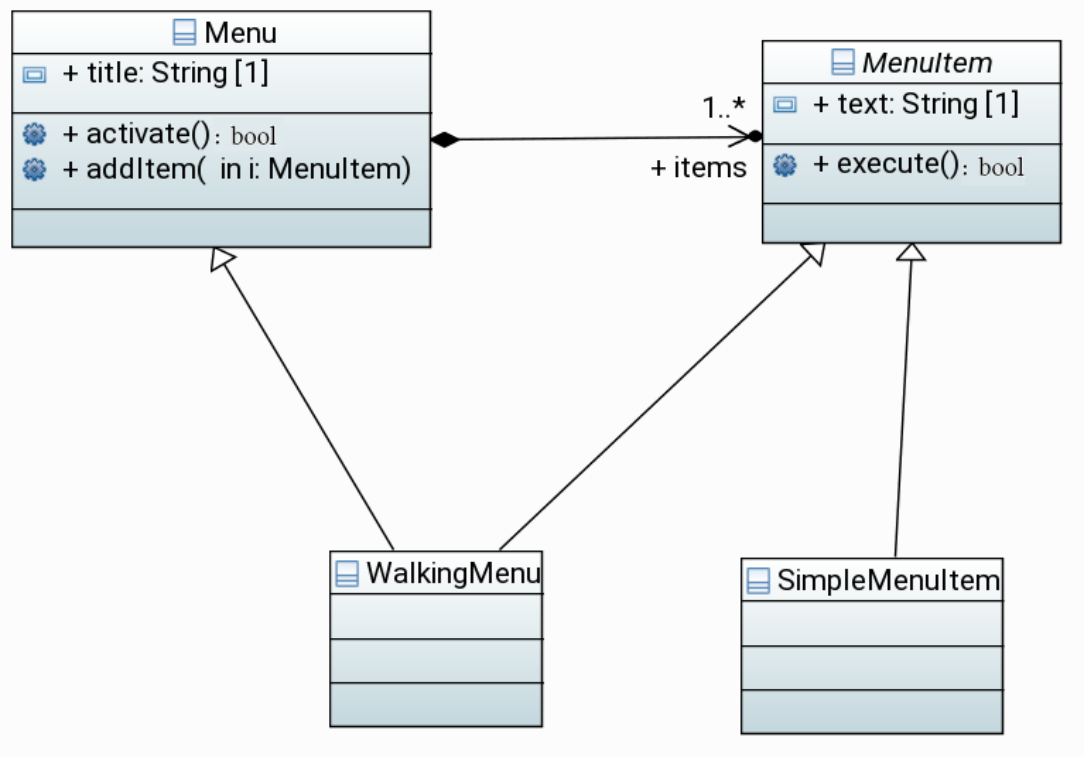

Figure 1: Spécification à réaliser

```
  0- emacs
  1- htop
  2- COMMUNICATIONS ->
  3- quitter
 Votre choix? "2"
 COMMUNICATIONS
  0- pidgin
  1- thunderbird
  2- rsync my home
  3- traceroute
  4- quitter
 Votre choix? "2"
    ************** Execution de rsync
 LE MENU
  0- emacs
  1- htop
  2- COMMUNICATIONS ->
  3- quitter
 Votre choix? "3"
    ************** Execution de QUITTER
jdeanton@linux-hani:~>
```

Reverse Engineering

Une fois que votre code fonctionne correctement, vous aller faire de la rétro-ingénierie afin de voir le diagramme UML issu de votre code. Pour cela regardez ici: `https://circle.visual-paradigm.com/docs/code-engineering/instant-reverse/how-to-generate-uml-from-cpp/`. Vous regarderez les différentes options et une fois le diagramme réalisé, vous le comparerez à celui de la specification Figure 1. Que pouvez vous en conclure ?