

Programmation efficace et avancée en C++

ArtWorks

V L T M O L K 2

Advanced C++

Workshop



Microsoft Certified

Professional

Solution Developer

Trainer

 **CompTIA**
Certified Technical Trainer+

www.artworks.fr

Michel André

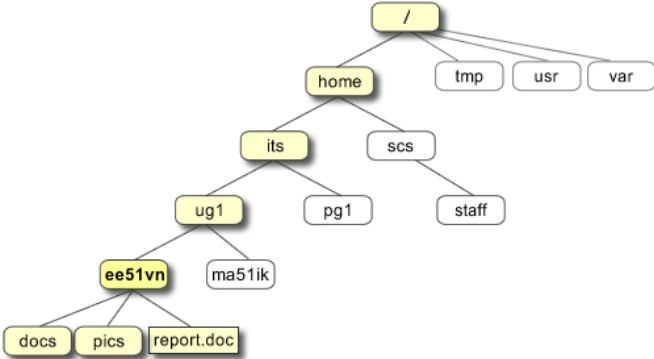
michel@artworks.fr

ArtWorks

V L T M O L K 2

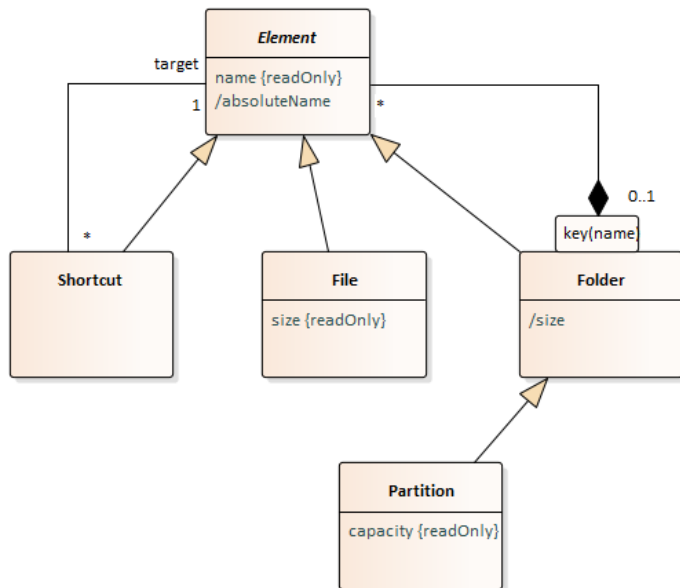
Advanced C++

File System



2

Le domaine métier final



© Artworks Tous droits d'utilisation et de reproduction réservés

Test Driven Requirement

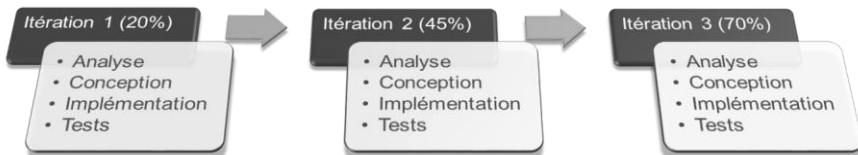
«How do you create a good interface to an abstraction ?

➔Write user code.

When you build the implementation before the interface, the interface inevitably smells like the implementation.»

comp.lang.c++.faq 70

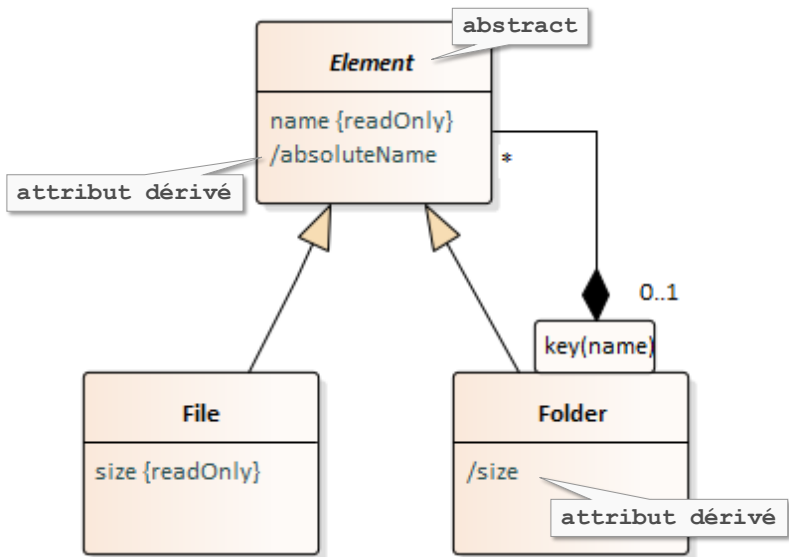
Cycle de vie itératif



© Artworks Tous droits d'utilisation et de reproduction réservés

4

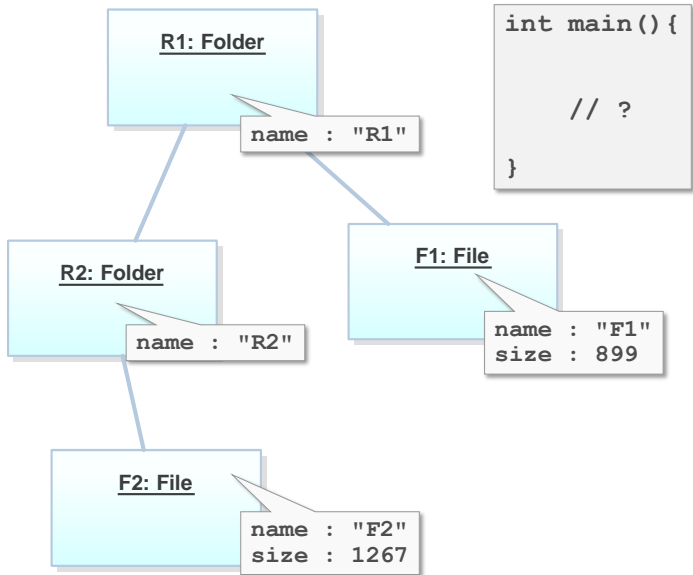
Step 1 – Setup (1 / 2)



@ Artworks Tous droits d'utilisation et de reproduction réservés

5

Step 1 – Setup (2 / 2)



@ Artworks Tous droits d'utilisation et de reproduction réservés

6

Step 1 – Règles de gestion

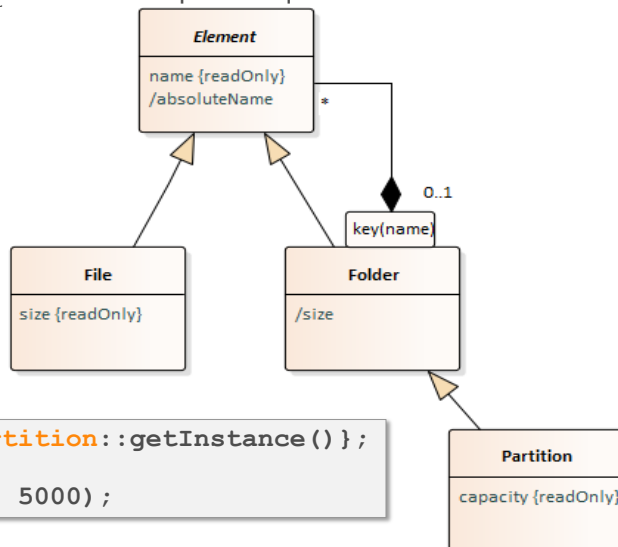
- Règles de gestion sur les noms:
 - un nom ne doit pas être vide
 - les noms des éléments enfant doivent être uniques au sein d'un répertoire.

Step 2 – Lazy Computation Idiom

- La taille d'un répertoire est dérivée (algorithmique)
 - elle peut être mise en cache lors de la première demande
 - elle doit être invalidée en cas de modification
 - ajout ou retrait d'un fichier par exemple
 - de façon *récursive* sur les *ascendants*

Step 3 - Partition

- on ne peut ajouter un fichier à un répertoire que si la taille restante de la partition est suffisante



© Artworks Tous droits d'utilisation et de reproduction réservés

9

Step 4 – RAI

- Utilisation d'un smart pointer pour gérer la destruction des éléments

© Artworks Tous droits d'utilisation et de reproduction réservés

10

Step 5 – Affichage Récursif

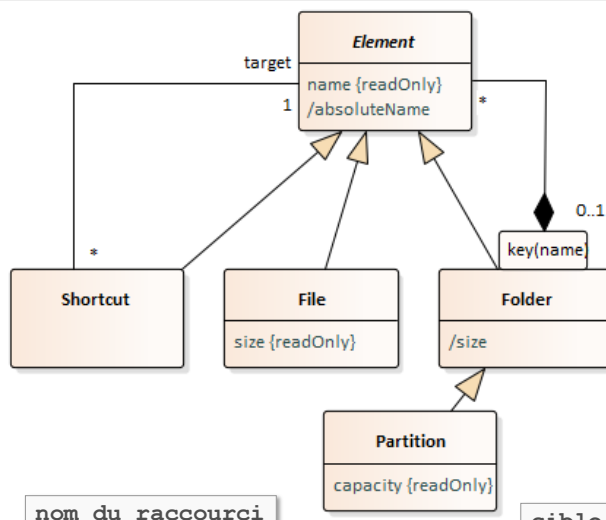
```
Folder& root {Partition::getInstance ()};
// .....
cout << root;
```

```
Partition: C:
File: autoexec.bat
Folder: temp
File: readme.txt
File: tmp1.dat
Folder: winnt
File: system.ini
Folder: system32
File: win.ini
```

© Artworks Tous droits d'utilisation et de reproduction réservés

11

Step 6 : Raccourcis (1 / 2)



```
r2.createShortcut("raccourci vers tmp1.dat", f);
```

© Artworks Tous droits d'utilisation et de reproduction réservés

12

Step 6 : Raccourcis (2 / 2)

```
Partition: C:
File: autoexec.bat
Folder: temp
  Shortcut: raccourci vers winnt --> /C:/temp/toto.fic
File: readme.txt
File: tmp1.dat
File: toto.fic
Folder: winnt
  File: system.ini
  Folder: system32
  File: win.ini
Partition: C:
File: autoexec.bat
Folder: temp
  Shortcut: raccourci vers winnt --> inexisting element
File: readme.txt
File: tmp1.dat
Folder: winnt
  File: system.ini
  Folder: system32
  File: win.ini
```

avant destruction

après destruction

© Artworks Tous droits d'utilisation et de reproduction réservés

13

Step 7 – Comptage des instances

```
cout << Folder::getNbInstances() << " folders\n";
cout << File::getNbInstances() << " files\n";
cout << Shortcut::getNbInstances() << " shortcuts\n";
```

© Artworks Tous droits d'utilisation et de reproduction réservés

14

Step 8 – Factory Method / Variadic Templates

- Mutualisation du code des 3 factory methods avec une fonction de ty variadic template

```
Folder& createFolder(const Name& folderName);  
  
File& createFile(const Name& fileName, Size fileSize);  
  
Shortcut& createShortcut(const Name& shortcutName,  
                        const Element& target);
```

© Artworks Tous droits d'utilisation et de reproduction réservés

15

Démarrage

```
Folder& root{ Folder::getRoot() };  
root.createFile("autoexec.bat", 5000_bytes);  
Folder& r1{ root.createFolder("winnt") };  
r1.createFile("win.ini", 300_bytes);  
r1.createFile("system.ini", 7000_bytes);  
r1.createFolder("system32");  
Folder& r2{ root.createFolder("temp") };  
r2.createFile("tmp1.dat", 645_bytes);  
r2.createFile("readme.txt", 200_bytes);  
cout << root.getSize() << " bytes" << endl;  
  
root.deleteElement("autoexec.bat");  
cout << root.getSize() << " bytes" << endl;  
  
cout << r2.getAbsoluteName() << endl;
```

© Artworks Tous droits d'utilisation et de reproduction réservés

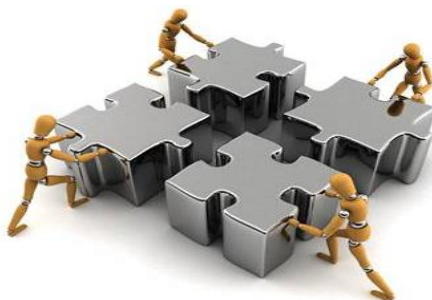
16

ArtWorks

VLMOLK2

Advanced C++

Généricité



17

Taxinomie animale

- Herbivore
 - mange de l'herbe
- Quadrupède
 - se déplace en trottant
- Bipède
 - se déplace en marchant
- Omnivore
 - mange de tout
- Carnivore
 - mange de la viande
- Prédateur
 - carnivore
 - tue sa proie
- Oiseau (Bird)
 - vole
 - omnivore par défaut
- Rapace (Raptor)
 - oiseau prédateur

Taxons

Animaux

Gazelle
Moineau
Singe
Aigle
Guépard

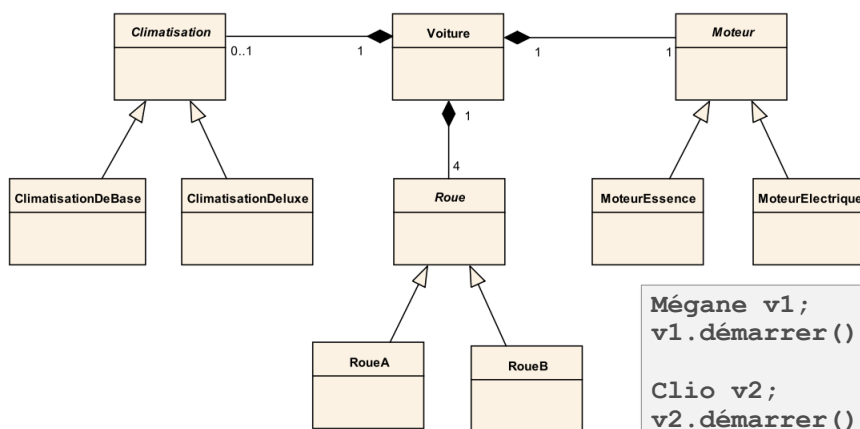


© Artworks Tous droits d'utilisation et de reproduction réservés

18

Séries automobiles

- Une Mégane possède un moteur électrique, des roues A et pas de climatisation
- Une Clio possède un moteur à essence des roues B et une climatisation de luxe
- Une TurboClio possède un moteur 3 fois plus puissant



```

Mégane v1;
v1.démarrer();

Clio v2;
v2.démarrer();
    
```

© Artworks Tous droits d'utilisation et de reproduction réservés

19