
Design Exercises

Solutions

Wolfgang Pree

C. Doppler Laboratory for Software Engineering

University of Linz

Mailing system

Methods of class HotelRoom:

IsAvailable(from, to): bool

DoReservation(from, to)

GetCategory(): String

CheckOut()

CalcRate(from, to): float

Instance variables of class HotelRoom:

list of reserved time periods

category/rate

cumulated extra costs (mini-bar, phone)

The abstract class **RentalItem** can factor out and implement the reservation mechanism (list of reserved time periods; **IsAvailable**, **DoReservation**). Otherwise only the protocol can be defined.

Fehler! Es wurden keine Einträge für das Inhaltsverzeichnis gefunden.**Flattened view of NestedFolder:**

Mailing system

NestedFolder	
Folder	folderName: String List <TextDocument *> *docs
NestedFolder	List <Folder *> *folders
Folder	GetNoOfItems(): Integer SetFolderName(name: String) GetFolderName(): String AppendDoc(td: ^TextDocument) RemoveDoc(td: ^TextDocument) GetDoc(name: String): ^TextDocument
NestedFolder	NestedFolder(name: String) SetFolderName(name: String) GetNoOfItems(): Integer AppendFolder(fp: ^Folder) RemoveFolder(fp: ^Folder) GetFolder(name: String): ^Folder

Mailing system

Abstract class DesktopItem:

<i>DesktopItem</i>
itemName: String
DesktopItem(name: String) SetItemName(name: String) GetItemName(): String <i>GetSizeInBytes(): Integer</i>

Remarks:

- It is much more important to introduce abstract classes than to define an appropriate interface right from the beginning
- The interface of abstract classes evolves. Thus design and implementation phases overlap.

Mailing system

Note the elegant implementation of **Folder** methods due to polymorphism and dynamic binding:

```
int Folder::GetSizeInBytes() {  
    int size= 0;    // folder sizes are not taken  
                   // into account  
    for each item in itemList  
        size= size + item->GetSizeInBytes();  
    return size;  
}
```

