LEFT TO DO

Inlämningsuppgift Objektorienterade principer och testning

DEN 28 NOVEMBER 2020

Wivianne Grapenholt
grwi20ql@student.ju.se

Innehåll

Beskrivning

Funktion

När programmet starta skapas en instans av klassen ToDoList, den skickas med som parameter när metoden för att visa huvudmenyn anropas. Alla menyer hanteras av en switchsats där [0] är tillbaka eller avsluta och en default som fångar fel inmatningar.

Visa ToDo / Archive först kontrolleras att listan innehåller uppgifter att visa. Sedan loopas ToDo igenom beroende på typ av uppgift anropas respektive metod med objektet och index som parametrar. Är det en checklistuppgift så anropas även en metod för att visa dessa, dvs itererar igenom subTask listan.

I undermenyn till ToDo finns det val och instruktioner till kunden. Lägga till uppgift anropar ytterligare en meny-metod vars uppgift är att guida användaren att välja typ av uppgift och lotsa flödet vidare till lämplig klass / metod. En instans av vald typ skapas som används för att anropa en metod som efterfrågar detaljer om uppgiften. Dessa sparas i objektets properties. Instansen läggs till i ToDo.

Markera uppgift. Metod för att hitta vilken uppgift som ska markeras anropas. Här efterfrågas vilken uppgift (index) som skall (av-)markeras, värdet subtraheras med ett för att motsvara rätt index i listan. Iterationen startar, om det är en Checklist instans efterfrågas nytt index nummer och en inre loop startar, denna håller även koll på om alla uppgifter i underlistan är markerade. I så fall markeras även rubriken.

Arkivera utförda uppgifter. ToDo itereras och letar efter instanser som är utförda, lägger till de i Archive, tar bort de från ToDo och efter varje flyttat objekt subtraheras i med ett för att inte missa någon instans i listan.

Struktur

- Menu – abstrakt klass, hanterar instruktions - output till användaren och styr programflödet baserat på användarens inmatade val.
- ToDoList – en instans av denna klass har två listor som properties, ToDo samt Archive. Metoder för att lägga till, visa, flytta (arkivera) samt hitta och (av-)markera objekt i respektive lista.
- **Tasks** – abstrakt superklass. Har följande medlemmar som de tre underklasserna ärver ifrån:
    - fyra properties som används för att urskilja vilket typ av objekt, spara själva texten på uppgiften, att hålla reda på om uppgiften är utförd samt att spara dagar till deadline.
    - ett fält för checklistans underuppgifter.
    - två metoder. En för att (av-)markera uppgifter som slutförda och en abstrakt metod för att ta in specifikationer från användaren när instansen skapas.
- SimpleTask – har utöver de ärvda metoderna
    - en överlagrad metod för att visa uppgift, avsedd för underuppgifter i checklistor som är instanser av klassen.
- Deadline – har utöver de ärvda medlemmarna
    - en try-parse metod för att förhindra krasch om fel värde matas in.
    - en metod för att räkna ut dagar till deadline från datum. Denna metod testas men är inte implementerad i själva programflödet.
- Checklist – har utöver de ärvda metoderna
    - lite extra funktion i metoden som tar in information, underuppgifter av typen SimpleTask (”type” ändras från ”S” till ”sub”)
    - en metod för att lägga till dessa i subTasks listan.

Använda principer

SimpleTask, Checklist och Deadline ärver alla från Tasks.
Detta möjliggör implementering av polymorfism genom att alla instanser kan läggas till i samma lista, använda samma metod för att markeras och arkiveras.
Inkapsulation åstadkom jag i Deadline genom att metoden för uträkningen av dagar är privat och en separat metod som returnerar värdet, i detta fall bara till enhetstestet. Och i Checklist klassen finns metoden för att lägga till subTasks. Om jag förstår begreppet korrekt så anser jag att även metoderna Create och ShowTask eftersom de är objekt specifika funktioner.

Tester för klasser

Deadline – jag valde att göra ett test som kontrollerar att metoden ComputeDaysLeft räknar ut rätt antal dagar utifrån angivna datum och returnerar svaret via GetDaysLeft. Från början använde jag DateTime.Today och ett fast datum men med intentionen att användaren skulle kunna skriva i datum och automatiskt räkna ut dagarna. Men för enkelhetens skull har jag angett fasta datum.
Checklist – här testar jag att det går att lägga till instanser av SimpleTask i subTasks listan via AddSubTask.
Jag har även lagt till de två typerna i övriga tester för att säkerställa att de har samma funktioner som SimpleTask.