Teknisk Dokumentation

Team 12

February 2021

1 Hur man använder programmet

Programmet kräver tre externa filer för att fungera i form av txt-filer som är namnfiler för deltagare med startnummer, starttider för varje deltagare samt måltider för deltagarna. Med dessa filer kan resultatprogrammet användas som ska ha ett grafiskt användargränsnitt där användaren ska kunna lägga in filer som motsvarar dessa tre som programmet behöver. När användaren känner sig färdig skall knappen "Generera resultat" klickas på och då kommer en fil skapas som innehåller all resultat från loppet.

Det går även att manuellt lägga in deltagare i registerprogrammet där tiden på datorns interna klockan används och ett valfritt startnummer får användas.

2 Bygga programmet

Projektet använder sig av Gradle. Scriptet för gradle finns färdigt, dock vid vidare implementation av projektet och konfogurationsfiler kommer det behövas att redigera scriptet så att allt kommer med som får programmet att vara funktionerligt.

3 Programmets beståndsdelar

Programmet består av sex olika beståndsdelar.

- Resultat
- Register
- Startfiler
- Namnfiler
- Målfiler
- Resultatfil

3.1 Vilka huvuduppgifter har de olika programmen och filerna?

Namnfilen har som uppgift att mappa ett startnummer till en förares namn. Startfilen används för att veta när alla förare startade. Målfilerna används för att veta när en förare gick i mål och beroende på lopptyp, vilken tid varven blev avklarade. Resultatfilen har i uppgift att visa resultatet på ett tydligt sätt.

Resulttat- och registerprogrammet är huvudpunkterna i programmet. Resultatprogrammet ska genom att ta in namnfiler och tidsfiler generera ett resultat som ska hamna i en resultatfil där resultat visas utifrån åldersgrupp, startnummer och tider som är sorterade för enklare visning av tiden på loppet. Registerprogrammet ska fungera som en manuel registrering av deltagare, där ditt startnummer skrivs in och som sedan då använder klockan på datorn (system time) för att bestämma din starttid. Om användaren skriver in en ogiltig starttid som inte innehåller något eller innehåller bokstäver kommer programmet att spara klocktiden utifrån första gången användaren trycker på "Registrera", alltså om användaren skriver in fel kommer ändå tiden från knapptrycket att sparas.

3.1.1 Hur samverkar de?

Huvudaktiviteten kommer att ske i resultatprogrammet. Resultatprogrammet itererar genom minst tre txt-filer (kan bli fler om fler måltidsfiler tillhandahålls) som då är startfiler, namnfiler och målfiler. Programmet går igenom filerna för att generera resultat baserat på vilken slags lopptyp det är. Layouten för resultatet med hänsyn till lopptyp kommer att se ut som följande:

- Maratonlopp:
  StartNR; Namn; Totaltid; Start; Mål.
- Varvlopp (varning baseras N-1 där N är hur många varv som utförs):
  StartNr; Namn; Varv; Totaltid; Varv1; Varv2; Varv3; Start; Varvning1; Varvning2; Mål
- Etapplopp:
  StartNr; Namn; Klubb; MC; Klass; Totaltid; Etapper; Etapp1; Etapp2; Etapp3; Etapp4; Etapp5; Start1; Mål1; Start2; Mål2; Start3; Mål3; Start4; Mål4; Start5; Mål5

3.1.2 Hur distribueras de på flera datorer under en tävling?

Under tävlingen har flera datorer registerprogrammet och varje dator gör sin egen målfiler. Dessa målfiler sätts ihop av resultatprogrammet som sedan genererar en resultatfil med hjälp av start och namnfilen. Resultatet kan genereras på flera datorer men behöver då alla filer.

4 Arkitektur

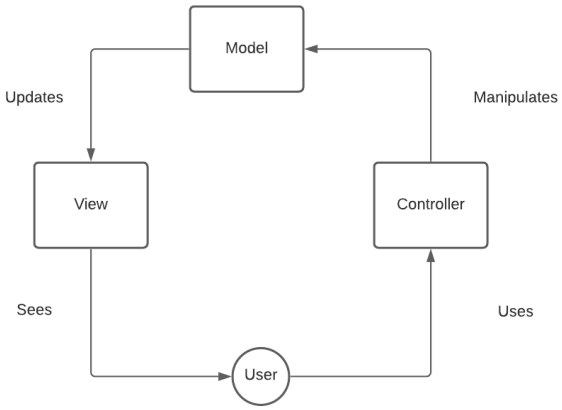

Figure 1: Bild som illustrerar hur MVC mönstret fungerar

Vi har i dagsläget två program som fungerar liknande varandra, ett för tidsregistrering och ett för resultatgenerering. Båda programmen använder sig av Model-View-Controller mönstret. I båda programmen görs det med hjälp av Observer-Obeservable klasserna i Java. Mönstret finns illustrerat i 1.

I resultatprogrammet använder vi oss dessutom av Decorator mönstret för att registerara felakttig input i textfilerna. Det görs genom att varje typ av error är en klass som läggs till under respektive typ av lopp. Det finns specifika errorklasser för varje typ av lopp. Template method används också på flera ställen i koden.

5 Kärnan för programmet

Den mest väsentliga klassen för programmet heter ResultProgram och det är där mainmetoden är skriven och hela programmet initieras ifrån. Klassen väljer vilken typ av lopp som ska köras och skapar sedan instanser av de nödvändiga klasserna för de olika lopptyperna.

En annan viktig fil är den abstrakta klassen matcher som kombinerar ihop den data vi har om förarna för att sedan skapa ett resultat. Det finns konkreta subklasserna för de olika lopptyperna och matcher klassen som en typ av insamlare av inforamtion från de andra beståndsdelarna av programmet. I konstruktorn av matcher hittas också olika typer av fel som finns i programmets textfiler, för att sedan registreras med hjälp av en errorfil.

6 Filformat

De filer vi andvänder är av typerna JSON och txt. Som konfigurationsfil använder vi oss av en JSON-fil. En JSON-fil är en typ av fil som enbart har funktionalitet för att visa enkel data och passar därför bra att använda som typ för konfiguration. För att programmet ska fungera som tänkt skriver man in vilken typ av lopp som ska köras och diverse parametrar i JSON-filen. En JSON-fil är lättare att hantera än en vanlig fil då det finns inbyggd funktionalitet i Java för detta.

Txt-filerna som används är vanliga textfiler som läses in med hjälp av Javas BufferedReader och skrivs till med hjälp av en FileWriter. Programmet som skriver resultatfilen kollar först om det finns en fil med samma namn och om det inte finns så görs en ny fil.

Vår dokumentation finns i en separat mapp och finns tilgänglig i pdf format. Det finns en manual för intruktioner om hur man använder programmet och den tekniska dokumentationen ligger också här.