Week 1: Januar-Projekt

02121, E 2022

Paul Fischer

Institut for Matematik og Computer Science
Danmarks Tekniske Universitet

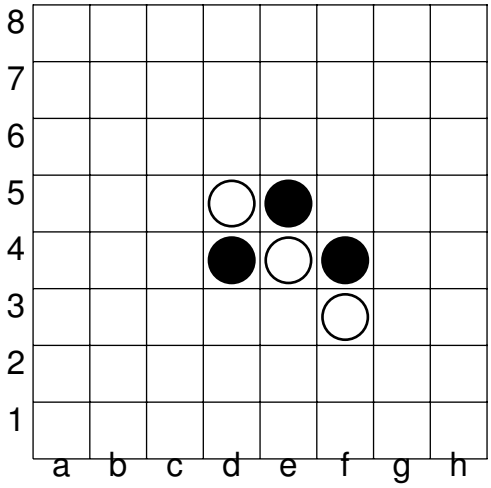
Efterår 2022

Projekt Januar 2023

GODT NYTÅR 2023!
Især sundhed.

Projektopgaven

Implementer spillet *Reversi*.



Detaljer findes i opgavebeskrivelsen.

Krav og Frister

- Opgaven skal laves i grupper af **4 personer**. I må selv vælge samarbejdespartner.
- Gruppedannelsen bliver færdiggjort 2.1.2023. Små grupper bliver fyldt op.
- Spillet skal implementeres i **Java** ved brug af **JavaFX**
- Rapporten skal være i LaTeX (Noter om rapport er på Learn.)

- På onsdag, 4.1., **kl. 12:00**, afleveres en projektplan på max 2 sider, der indeholder følgende:
  - Gruppenummer, navne, studienumre, dato.
  - En problemanalyse, som indeholder udfordringer og hvordan I har tænkt jer at løse dem.
  - En skitse af repræsentationen af problemet, især bordet og brikker.
  - En grov beskrivelse af delopgaver/arbejdspakker og en foreløbig tidsplan.

  Feedback gives senest mandag 9.1.
- **Obligatorisk** demonstration af produktet den 19. og 20.1.
- Opgaven skal afleveres senest fredag d. 20.01.2023, kl. 23:59 på DTULearn. Karakteren forventes circa 2 uger senere.

Bemærninger

- Afklar ambitionsniveauet i gruppen.
- Find en platform for kommunikation.
- Brug et par dage, for at analysere problemet og strukturere løsningen, inden i rører ved computeren.
- Væsentlige ændringer senere i projektet koster dyrt.
- Lav en skøn af tidsbehov af delopgaver og fordele opgaver.
- Definere grænseflader mellem delopgaverne.
- Glem ikke rapporten; det er den væsentlige del for bedømmelsen.
- Hvis der er alvorlige problemer/konflikter, så informer underviseren med det samme.

Plan for Januar

- **2.1.2023** Gruppedannelse afsluttes på DTULearn. Grupper **skal** består af **fire** medlemmer.
- **2.1.–20.1.2023** 3-ugers-perioden. Arbejde med projektet.
- **4.1.2023, kl. 12:00**, afleveres en projektplan
- **19.1.2023–20.1.2023 Obligatorisk** demonstration af produktet.
- **20.1.2023** Aflevering af projektet, detaljer på opgavebeskrivelsen.

Der er ingen forsvar af rapporten.

I 3-ugers perioden er der spørgetime hver mandag til torsdag kl 10-11, frem til 18.1., i bygning 322, rum 006. Der er ingen virtuel spørgetime, heller ikke på mail.

Der er to hjælpelærer, Nynne og Benjamin. En tidsplan kommer snart.

Følgende rum er tildelt i hele 3-ugers-perioden kl. 8-17: **324-060** og **324-070**.

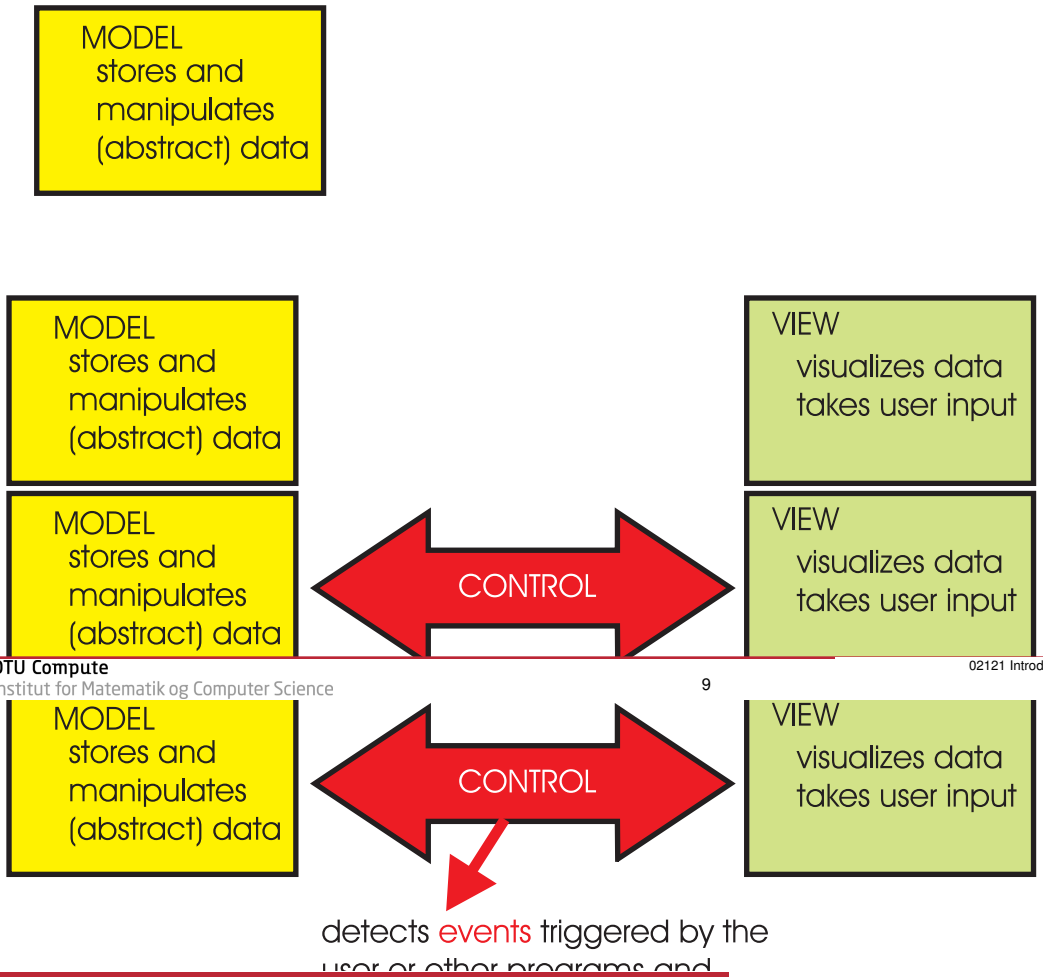
Spørgsmål?

MVC

Model-View-Controller Koncept

på Engelsk

Model-View-Controller



Model-View-Controller

The programmer has to do the following:

(Model) select appropriate data structures to store and the data and implement the methods to manipulate it.

(Controller) specify the events to which the program should react.

(View) select the graphical components to visualize the data and those actions that the user can use to trigger events.

(Controller) specify the reactions to the events.

Sometimes a strict distinction can be artificiel. For example the JavFX-view-components contain elements of controlers.

Life-cycle of a MVC-program

Start-up Memory is allocated, data structures are initialized, data is loaded, start-up computaions are performed. The GUI is constructed.

Show The GUI is made visible. From now on the application is completely event-driven.

Working Events are detected and processed. The data is modified and the changes are displayed.

Shut-down Data is saved, memory is cleaned, GUI is made invisible, program is stopped.

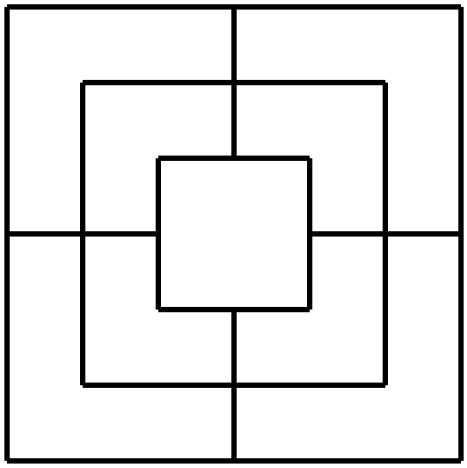
Events

Possible events to which the program has to react:

- Pressing a button on the screen.
- Clicking a mouse button.
- Moving the mouse.
- Pressing a key on the keyboard.
- Selecting a menu.
- Requests coming form other programs or the operational system.

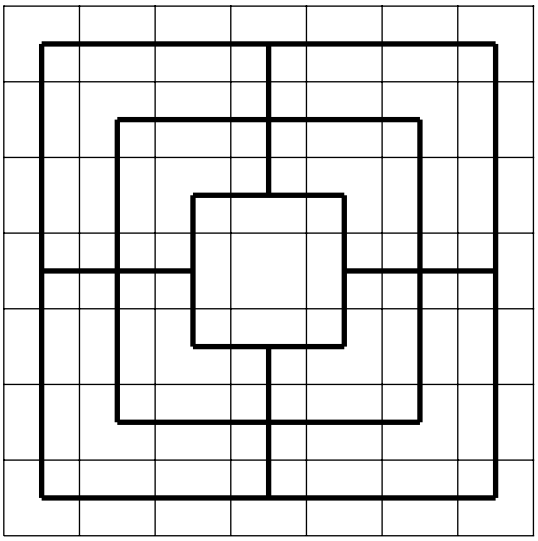
Om Modeller

Eksempel: Spillet *mølle*



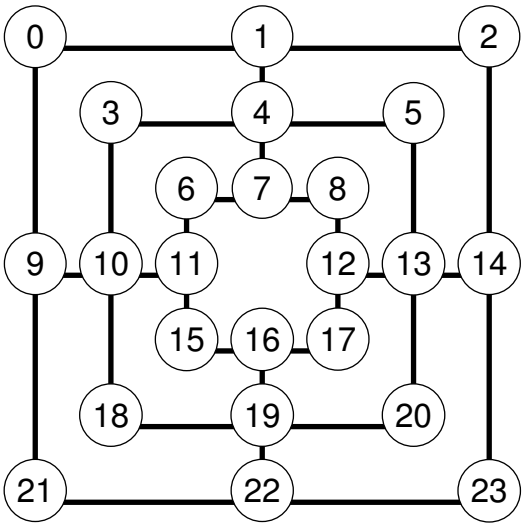
Om Modeller

Model 1: fliser. Der er 49 fliser af 11 forskellige typer (for array lovers).



Om Modeller

Model 2: En graf som datastruktur (mere i 02105/02110).



Naboskab:

```
nabo(10,9) = true
nabo(10,7) = false
```

Beboer:

```
beboer(2) = sort/hvid/ingen
```

Træk: Sort vil trække fra a til b

```
if (nabo(a,b) and
beboer(a) = sort and
beboer(b) = ingen)
  beboer(a) = ingen
  beboer(b) = sort
else
  ERROR
```


Tests

Test jeres modeller efter implementationen.

Lave en liste med inputs og forventede outputs.

Check om programmet/classer/metoder opfylder dem.

I vil høre mere om tests i kursus 02161 “Intro til software engineering”.

Tilstandsmaskiner (Finite Automaton)

Komponenter: Tilstand, overgange udløst af hændelser.

Tilstand: beskriver den nuværende status (data) af maskinen.

Hændelser (events): kommer uden fra (controler) og skal håndteres. Udløser ændringer af maskinens status og (muligvis) overgang til en anden tilstand.

Konceptet **kan** bruges til at bedre strukturere modellen og til at opdele arbejdsopgaver, især hvis opgaven indeholder en dynamisk komponent.

Tilstandsmaskiner (Finite Automaton)

Eksempel: Cola-automat.