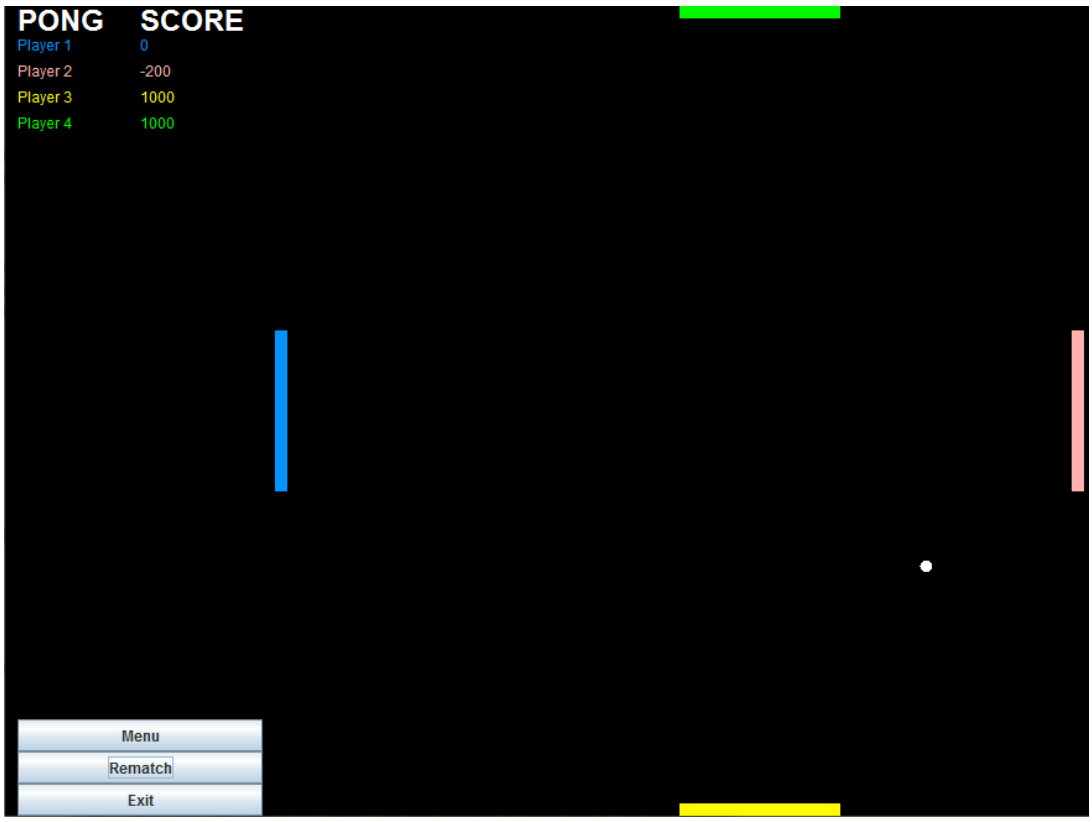
Pong – ett spel

Kurs: Objektorienterade applikationer, DAT055
Datum: 2012-02-29



Grupp 4:
Anders Emanuelsson, andemanu@student.chalmers.se
Daniel Kvist, dkvist@student.chalmers.se
Dina Zuko, zuko@student.chalmers.se
Gustaf Werlinder, gustaf_werlinder@hotmail.com
Markus Schützer, marschu@student.chalmers.se
Patrik Thituson, thituson@student.chalmers.se
Simon Fransson, frsimon@student.chalmers.se

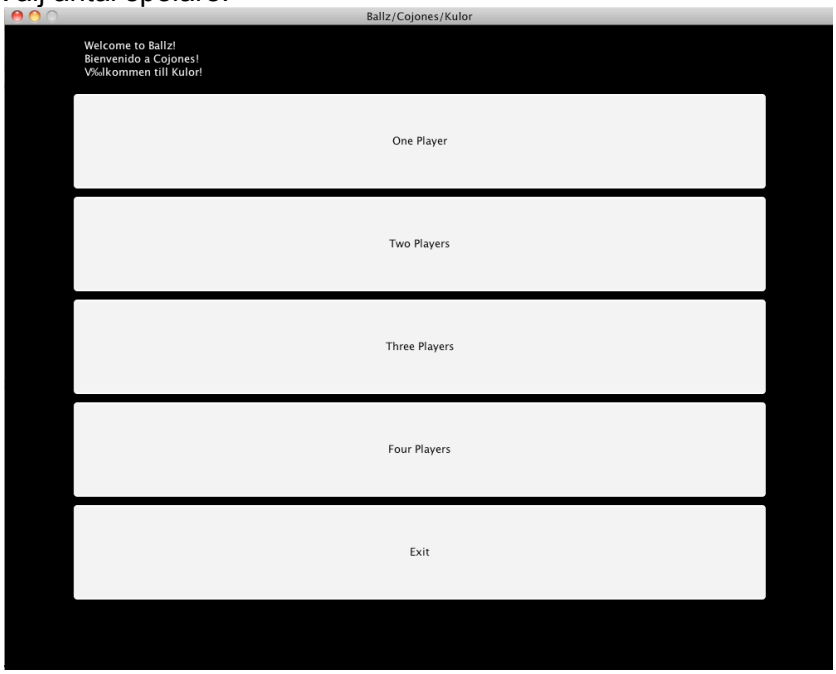
Innehållsförteckning

1. Sammanfattning

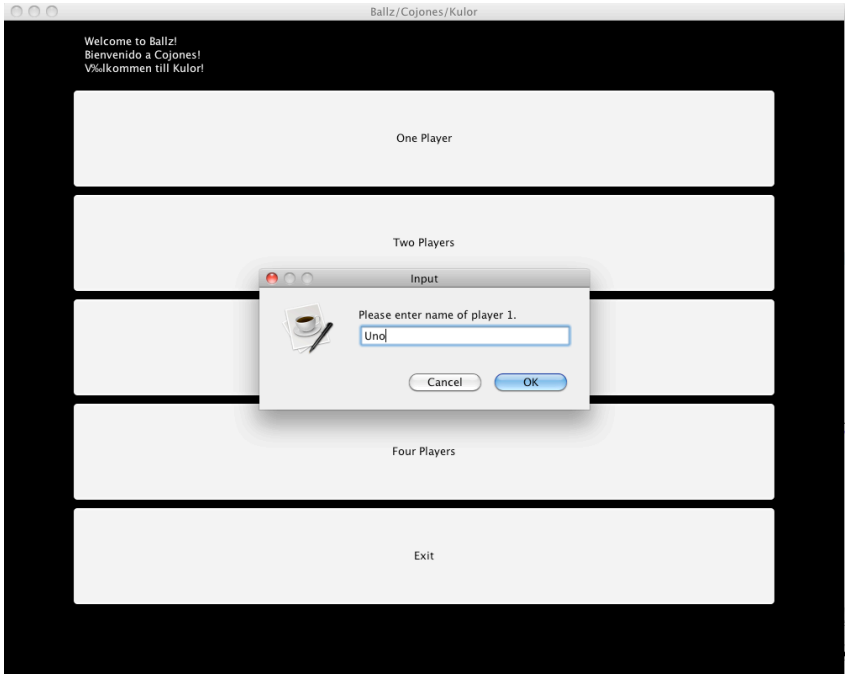
Rapporten behandlar Java-applikationen Pong som är ett spel för upp till fyra användare. En användarmanual berättar i enkla steg hur spelet fungerar. Applikationens struktur behandlas övergripande i en beskrivning av hur den arkitektur, Model View Controller (MVC), och det designmönster, Observer, som använts har implementerats. En mer detaljerad beskrivning av applikationens design ges i diagram över dess paketstruktur och klasstruktur. Därtill presenteras klasspaketvis en enkel beskrivning av klassernas respektive uppgifter i applikationen.

2. Användarmanual

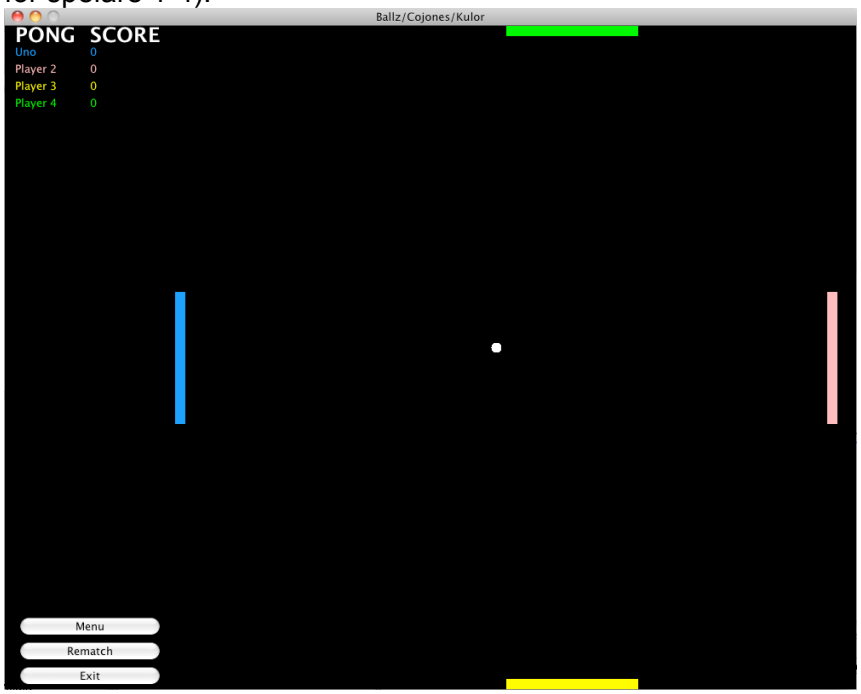
1. Välj antal spelare.



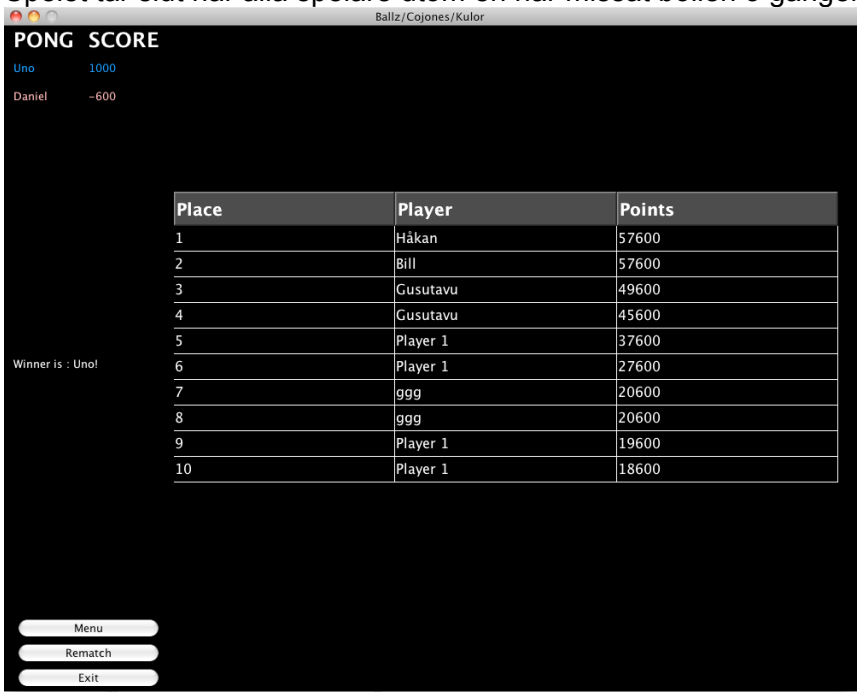
2. Skriv in spelarnamn.
   För att låta datorn kontrollera en spelare lämnas namnfältet tomt. Klicka “Ok”.



3. Spelet startar efter två sekunder. Varje spelare styr varsin paddel (se rubriken Kontroller för spelare 1-4).

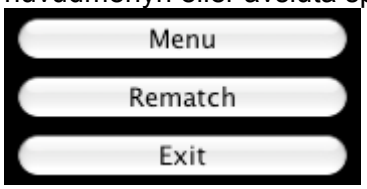


4. Spelet tar slut när alla spelare utom en har missat bollen 3 gånger.



Place	Player	Points
1	Håkan	57600
2	Bill	57600
3	Gusutavu	49600
4	Gusutavu	45600
5	Player 1	37600
6	Player 1	27600
7	ggg	20600
8	ggg	20600
9	Player 1	19600
10	Player 1	18600

5. Klicka på menyn i nedre vänstra hörnet för att spela returmatch, gå tillbaka till huvudmenyn eller avsluta spelet.



6. **Kontroller för spelare 1-4**
   Spelet styrs med hjälp av knappar på tangentbordet. Man trycker ned den knapp som motsvarar den riktning dit man vill styra sin paddel. Kontrollerna är som följer:

   - *Spelare 1 (blå)*
     - Upp: Q
     - Ned: A
   - *Spelare 2 (rosa)*
     - Upp: Pil upp
     - Ned: Pil ned
   - *Spelare 3 (gul)*
     - Vänster: K
     - Höger: L
   - *Spelare 4 (grön)*
     - Vänster: F
     - Höger: G

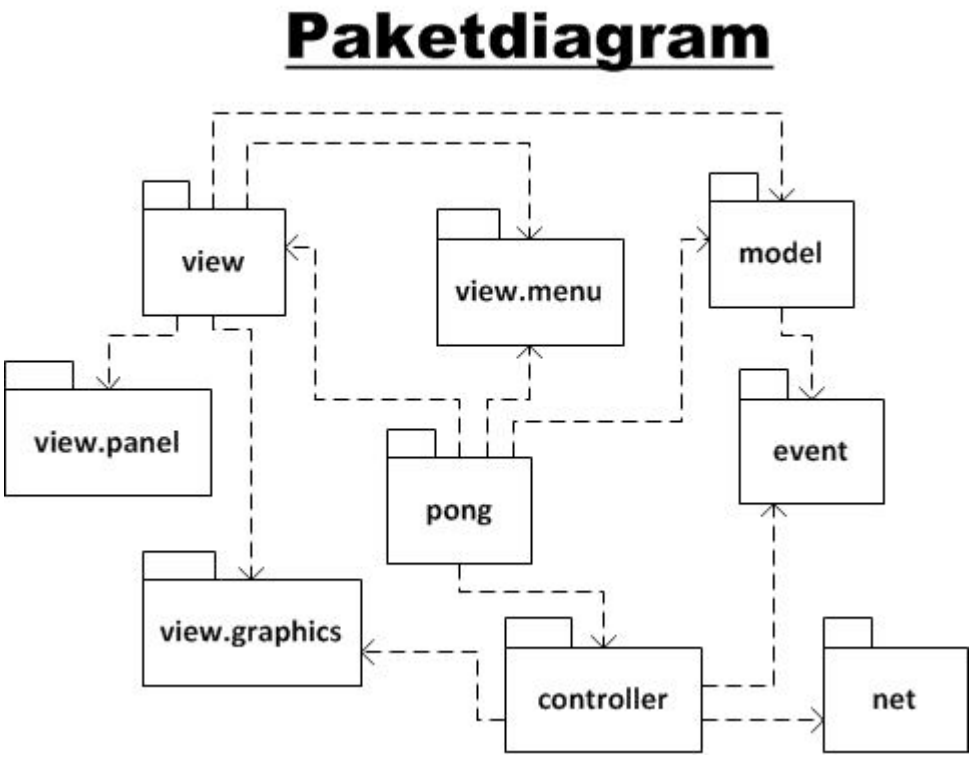
3. Modellbeskrivning

Övergripande modellbeskrivning

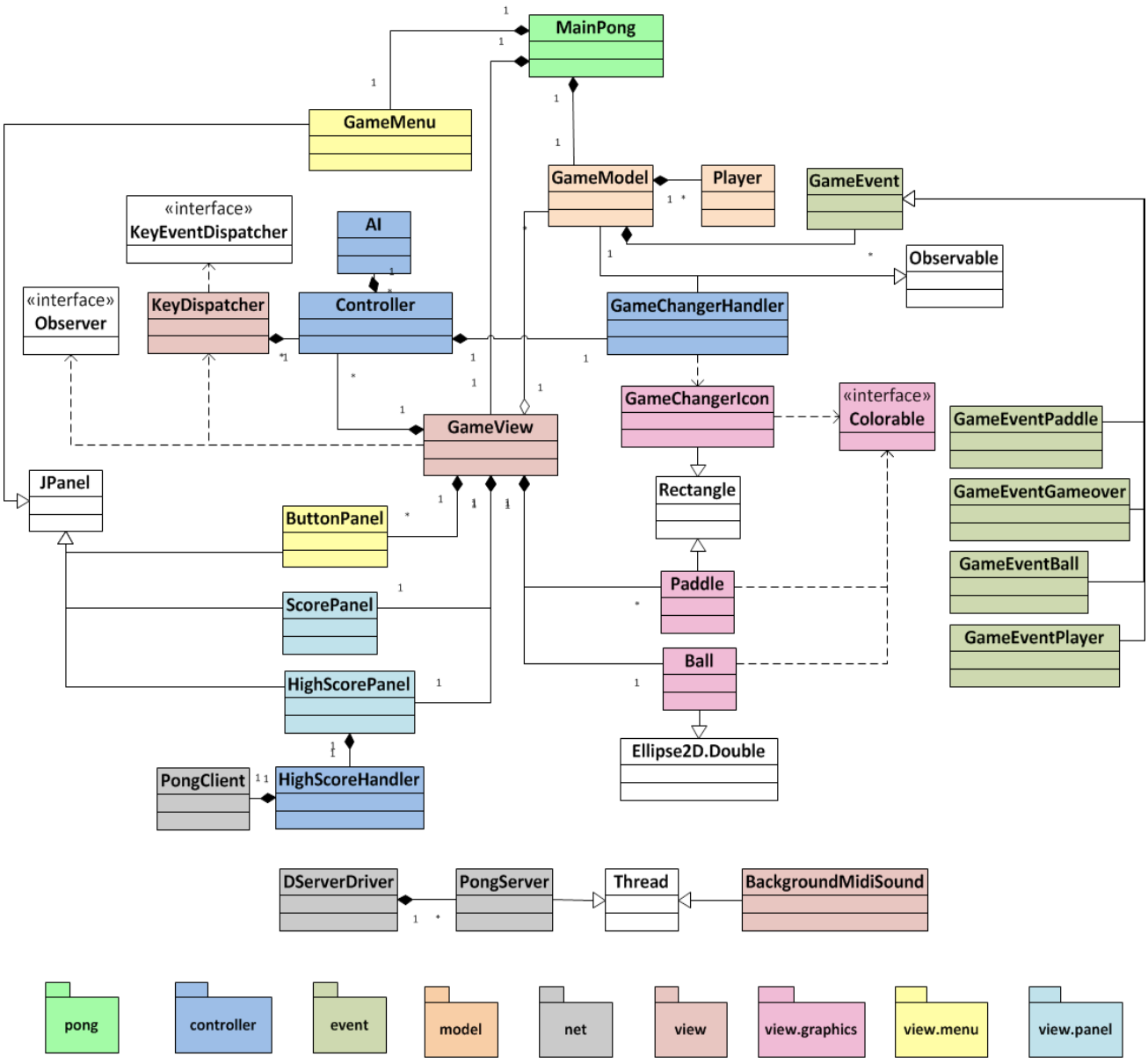
Vi har valt en Model View Controller (MVC) -arkitektur för applikationen. MVC passar mycket bra för den här typen av program eftersom det har tydliga gränser där man kan dela upp programmet. För att implementera MVC har designmönstret Observer använts. Applikationens View representeras i huvudsak av klassen *GameView*, dess Model motsvaras av klassen *GameModel* och dess Controller-klass heter *Controller*.

GameView innehåller de grafiska elementen och tar även hand om knapptryckningarna på tangentbordet från användaren. *GameView* säger sedan åt *Controller* att räkna ut vad som skall hända i spelet härnäst. Det kan vara att en paddel eller boll skall flytta på sig eller ändra riktning. *Controller* sköter om logiken och meddelar sedan *GameModel* som sparar de aktuella värdena och spelets tillstånd.

GameModel som är en *Observable* enligt Observermönstret är observerad av *GameView*. När modellen sparat sin data anropar den *notifyObservers*() för att *GameView* skall kunna uppdateras. I anropet till *notifyObservers*() finns ett GameEvent-objekt med som argument så att *GameView* skall kunna avgöra vad som skall förändras i vyn.



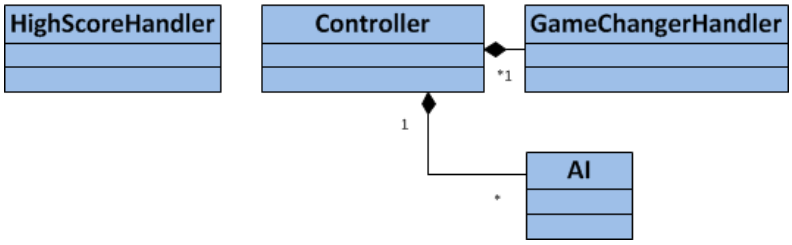
Klassdiagram Pong



Klassbeskrivning

Controller-paketet

Detta paket innehåller spelets logik. Klasserna i detta paket anropas från vyn och räknar ut vad som skall ske härnäst i spelet. Huvudklassen i paketet är *Controller* som tar hjälp av de övriga klasserna för att driva spelet framåt.



AI

Denna klass beräknar hur en paddel i spelet ska röra sig om den inte styrs av en mänsklig spelare. En AI-spelare kan ses som en sparringpartner att öva med.
beroende av användargränssnittet. Informationen förändras på initiativ från andra delar av spelet.

Controller

Detta är spelets motor. Den innehåller de algoritmer och fasta konstanter (de konstanter som ej utgår från skärmupplösningen) som styr hur spelets rörliga delar fungerar och interagerar.

GameChangerHandler

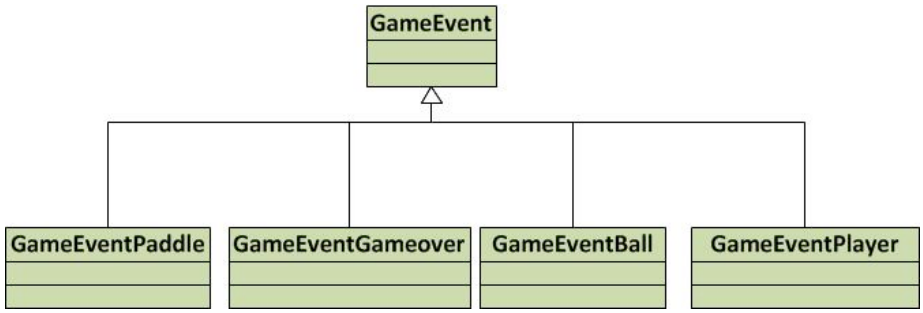
Denna klass hanterar spelets muterare (för närvarande endast en) under spelets gång. Klassen ser till att en ikon (en *GameChangerIcon*) skapas på spelplanen, initierar rätt förändring när bollen krockar med ikonen och tar bort ikonen efter en viss tid om ingen kollision med bollen inträffat.

HighScoreHandler

Denna klassen läser och skriver information om high-score-tabellen från/till fil via client/server. Den hanterar en befintlig lista, ordnar spelarnamn efter högsta poängsats. Om det inte går att läsa från filen skapas istället en “standardlista”.

Event-paketet

Detta paket innehåller klasser som används för att transportera data mellan vår modell och vy. Applikationen innehåller flera olika typer av händelser och varje händelse har en egen Event-klass. Modellen skapar instanser av dessa klasser och skickar med dem till observatören när modellen anropar *notifyObservers*.



GameEvent
Basklass till de specifika Event-klasserna som innehåller gemensam och grundläggande event-information.

GameEventBall
Klassen är en hjälpklass för att olika funktioner i programmet ska fungera vid olika händelser, speciellt kring bollens beteende i spelet. Den innehåller information om faktorer som påverkar bollens fart och riktning.

GameEventGameOver
Detta är en hjälpklass för att olika funktioner i programmet ska fungera vid händelsen att matchen är över. Den innehåller spelarlistan för aktuell match samt vinnaren.

GameEventPaddle
Ser till att en paddels koordinater kan uppdateras.

GameEventPlayer
Detta är en hjälpklass för att olika funktioner i programmet ska fungera vid olika händelser, speciellt kring spelarens tillstånd i spelet. Den innehåller information om huruvida en spelare är utslagen eller kvar i spelet, om dess poäng har ändrats, hur poängställningen ska ändras och hur spelarens antal liv ska påverkas.

Model-paket

Model-paket består av de klasser som utgör modellen i applikationen. Dessa klasser lagrar spelets nuvarande status. Information som lagras är till exempel hur mycket paddlarna skall flytta på sig vid nästa uppdatering av vyn.

GameModel

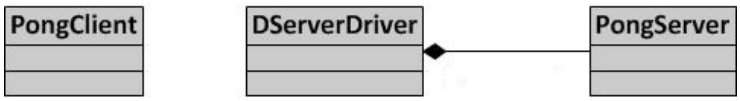
Håller den information (listor och variabler) om spelets beståndsdelar som inte är direkt

Player

Klassen representerar en spelare i spelet. Om namnfältet lämnas tomt då spelarna registreras räknas spelaren som datorstyrd av *AI*. Klassen innehåller information om spelarens namn, om den är en mänsklig spelare eller styrs av *AI*, spelarens poängställning samt kvarvarande antal liv.

Net-paketet

Klasserna i detta paket har hand om client-server kommunikationen som krävs för att lagra high-score på en extern server.



DServerDriver

En fristående applikation som innehåller en *main* funktion som startar en instans av *PongServer*.

PongClient

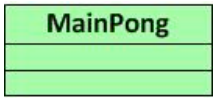
Klassen representerar klienten som kommunicerar med och lagrar high score på servern.

PongServer

Klassen hanterar serverns kommunikation för att hantera high-score. Servern kommunicerar med en klient för att utbyta data i form av high-score-listan.

Pong-paketet

Detta är applikationens default paket som ligger i rooten för alla andra paket. Här ligger endast en klass som innehåller *main* funktionen för applikationen.

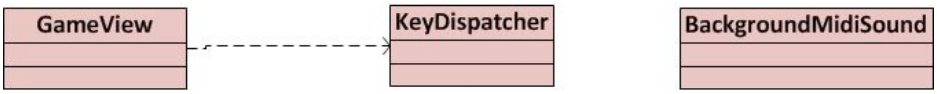


MainPong

Klassen startar, startar om och avslutar spelet. Omstart och avslut initieras av andra delar av programmet.

View-paketet

Klasserna i detta paket har hand om allt det som spelaren ser, hör och gör i spelet. Musiken som spelas hanteras i detta paket tillsammans med tangentbordstryckningar och de grafiska elementen i spelet.



BackgroundMidiSound

Denna klass spelar en melodi till så länge programmet är igång.

GameView

Klassen är spindeln i nätet vad gäller det grafiska gränssnittet. Skapar och ritar upp dels spelplanen (med dess olika beståndsdelar), dels en tabell som visar spelarnas poäng och dels den knappsats som finns i klassen *ButtonPanel*.

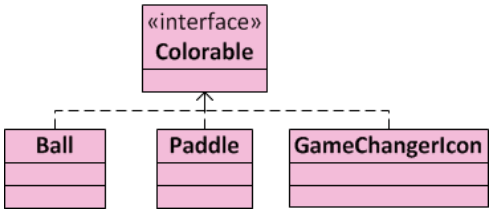
Utöver att initialt beräkna lämplig storlek (utifrån aktuell skärmupplösning) på spelplanen och dess beståndsdelar utför *GameView* själv inga beräkningar utan tar bara emot information om beståndsdelarnas tillstånd från andra klasser. Bilden uppdateras sedan kontinuerligt för att visa hur spelplanens beståndsdelar och poängtabell förändas över tid.

KeyDispatcher

Registrerar de knapptryckningar på tangentbordet som ska styra paddlarna och i viss mån påverka bollens riktning.

View.graphics-paketet

I detta paket ligger alla de grafiska element som ritas upp i spelet.



Ball

Klassen representerar bollen i spelet. Den innehåller information om bollens färg, diameter samt position för bollen.

Colorable

Detta interface bestämmer att de klasser som implementerar *Colorable* kan färgläggs och även berätta om vilken färg den har.

GameChangerIcon

Klassen representerar en ikon på spelplanen. Den innehåller information om storlek, färg, koordinater samt om huruvida figuren är aktiv och därmed kan påverka spelet eller ej.

Paddle

Klassen representerar en paddel. Den innehåller information om huruvida paddeln är styrbar eller om den är inaktiv som en “vägg”, vilken färg den har, dess koordinater och dess storlek.

View.menu-paketet

I detta paket ligger våra menyer som finns i spelet.

ButtonPanel

Knappsats som möjliggör för spelarna att återvända till startmenyn (*GameMenu*), spela en ny match med samma deltagare som i senaste matchen eller avsluta hela spelet.

GameMenu

Knappsats där spelarna att anger hur många spelare som ska vara med i matchen varpå spelarnas namn skrivs in. Om ett namnfält lämnas tomt blir spelaren en *AI*. Alternativt avslutas spelet istället med ett klick på Exit-knappen. Knappsatsen är det första spelaren möter när programmet startas.

View.panel-paketet

I detta paket ligger paneler som vi använder oss av för att visa till exempel high-score och nuvarande poäng.

HighScorePanel

Klassen representerar high score-tabellen som presenteras då matchen är över. Den både uppdaterar och hämtar information om high score-tabellen via *HighScoreHandler*.

ScorePanel

Klassen representerar rutan med spelarnamn, aktuell poängställning och färg för samtliga spelare i matchen.