Grupp 38
Ramverk, del 1

Jonas Enlund
Anton Mårtensson

1 Vad är skillnaden på GoldModel och GameModel, och hur är de relaterade till varandra?

GoldModel är en GameModel (implementationsarv). GoldModel definierar spelet Gold, medan GameModel definierar saker som är gemensamma för samtliga spel och fungerar mer som ett ramverk. GameModel är en abstrakt klass som som i och med det delvis definierar dess subklasser, såsom GoldModel.

2 Vilken klass utför själva ritandet av 'Gold coins' i spelet 'Gold game'?

`public void draw(final Graphics g, final int x, final int y, final Dimension d)`
i RoundTile och
`public void draw(final Graphics g, final int x, final int y, final Dimension d)`
i RectangularTile definierar modellerna som ska ritas, medan själva ritandet utförs av Javas inbyggda `draw`-funktioner som anropas i

```
public void setModel(final GameModel model) {
            this.model = model;
            repaint();
}
```

i GameView. `GameModel model` här innehåller alla de delar som ska ritas.

3 Vilken klass är det som anropar för detta ritande?

Som ovan ritas allt med `repaint();` i GameView. Detta anrop initierar de inbyggda `draw`-funktionerna i Java.

4 Vad är syftet med GameFactory och hur kan den utvidgas för att få med Snake game? (Endast ett kort svar behövs.)

Den ger namn på de spel som finns tillgängliga och ett system för att välja det spel man vill spela. För att få med Snake game kan följande läggas till:

```
return new String[] { "Gold", "Snake"};
```

och

```
@Override
public GameModel createGame(final String gameName) {
            if (gameName.equals("Gold")) {
                        return new GoldModel();
            } else if (gameName.equals("Snake")) {
                        return new SnakeModel();
            }
            throw new IllegalArgumentException("No such game: " + gameName);
}
```

osv.

5 Var återfinns beräkningen och kontrollen av 'gold eaters' rörelser?

I GoldModel återfinns alla tillåtna riktningar och definitioner för hur olika riktningar ska hanteras. Själva lyssnaren, `keyListener` återfinns i GameController. Den nya positionen ges beroende på nuvande position och riktning i `private Position getNextCollectorPos()` i GoldModel.

6 Hur lagras ett spelbräde och i vilken klass?

Storleken anges i klassen Constants och används sen i `private final Dimension modelSize;` i GameView och som `private final Dimension gameboardSize` i GameModel. Själva brädet skapas och anväds med `private final GameTile[][] gameboardState;` i GameModel.

7 Beskriv arbetsflödet av programmet.

Main konstruerar en GUIView baserad på GameFactory.

GUIView konstruerar en GameController och GameView.

I GameFacory skapas en lista med de tillgängliga spelen, och när ett spel valts returneras en GameModel som t.e.x. GoldModel.

GameController handhar keyListener och en tråd gemeThread som uppdaterar det valda spelet och stoppar det när spelet avslutas.

GameView tar in argument för att rita spelbrädet och en GameModel som ska ritas. GameController anropar GameView för att rita om allt.

I GoldModel finns metoden gameUpdate som hanterar spelmekanismen.

8 Ge en plan för hur du tänker fortsätta med deluppgift 2. Vilka klasser kommer du att skriva och vad kan du återanvända.

Jag kommer skapa klassen SnakeModel baserad på GoldModel. Tillagda metoder kommer vara metoder för att lägga till svanselement i en `ArrayList` när ett mynt tas samt att få denna svans att följa `Collector`. GameUpdate ska komma att överskugga den abstrakta metoden i GameModel och inkludera kod för att lägga till svanselement om `(this.coins.remove(this.collectorPos))`.