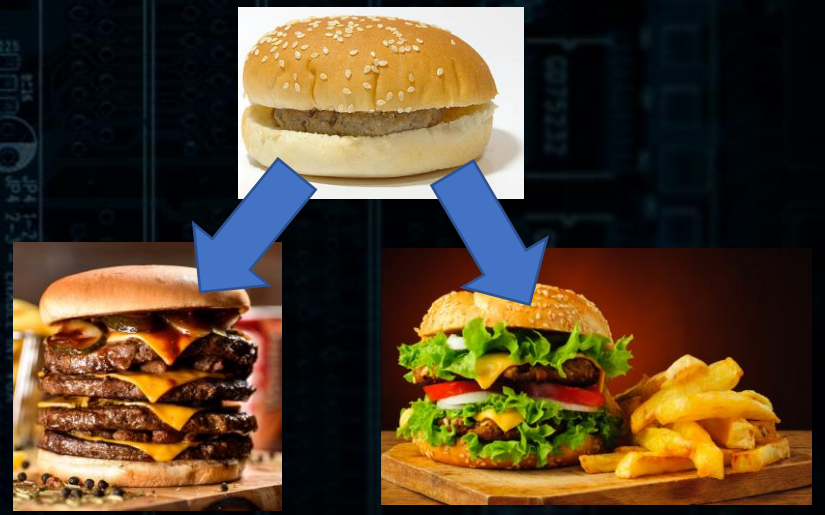
Objectgeoriënteerd programmeren

Klas EHI1V.Tg, docent Jan Willem Boer

We beginnen om 12:30

Vorige week

- Overerving
- Packages



Deze week

- Polymorfisme
- Klassenontwerp / modelleren

Vandaag:

- Kennisquizje
- Polymorfisme, met opdracht

Quiztime

https://bit.ly/poll-2020
Room Name: JBO37

Polymorfisme

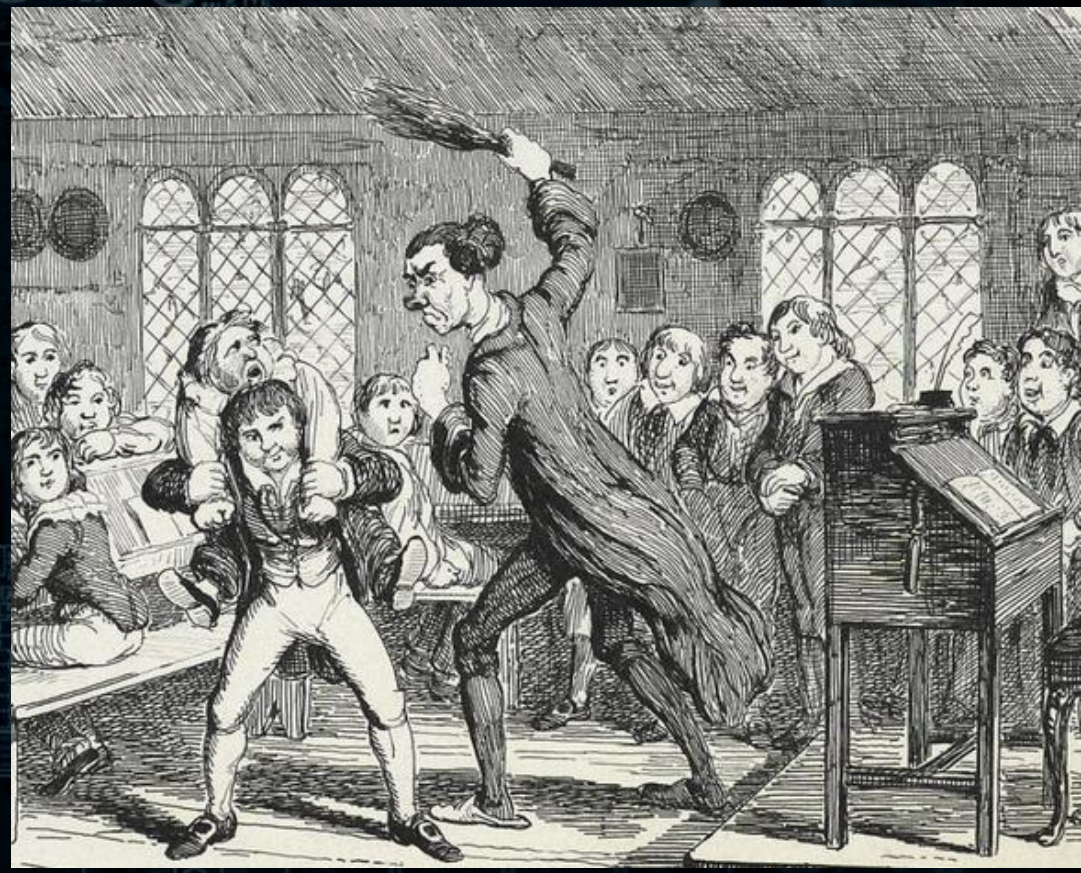
- Verschillende klassen via dezelfde interface benaderen
- Vandaag: verschillende subklassen
  via dezelfde superklasse benaderen.

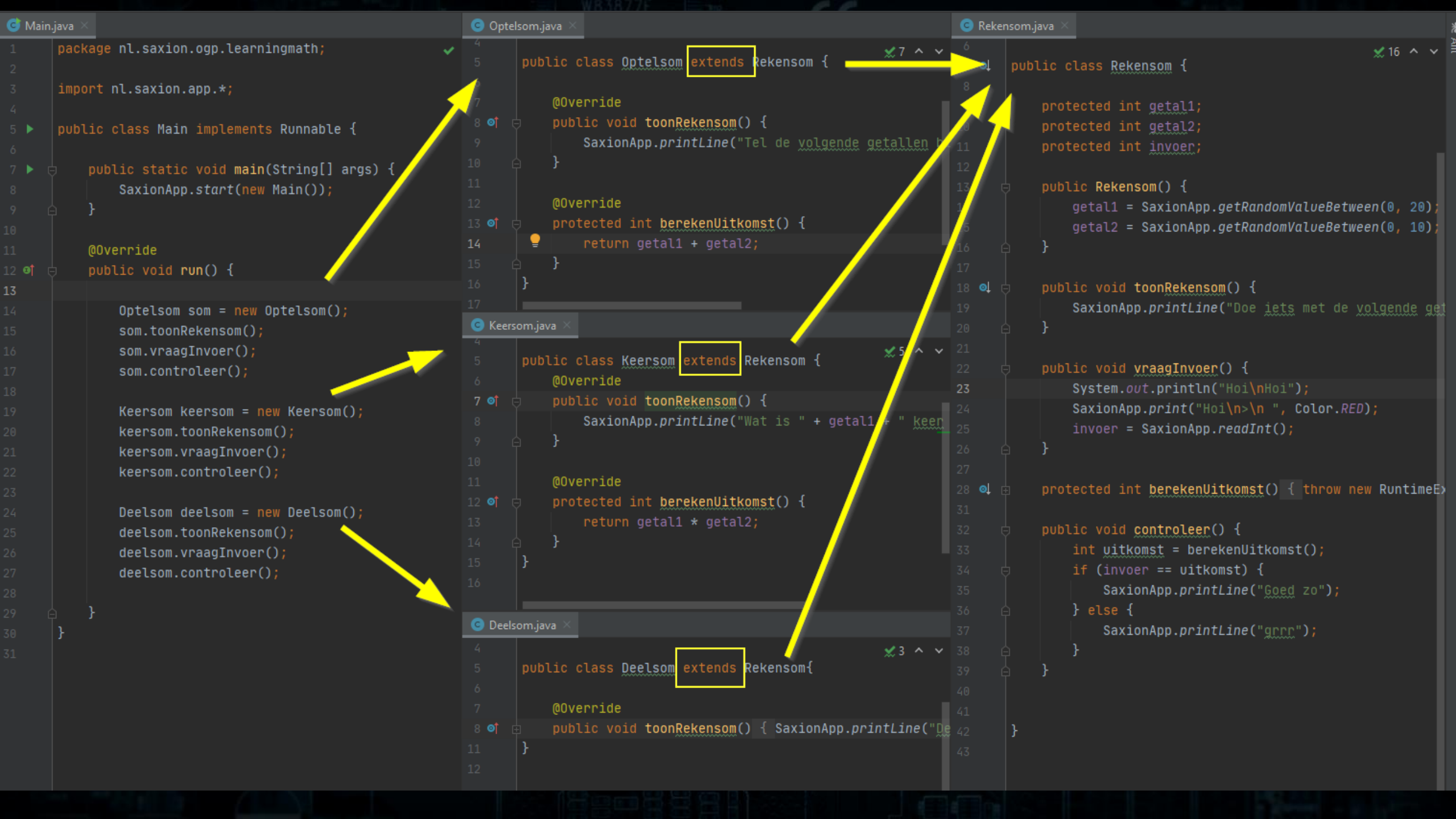
```
Optelsom som = new Optelsom();
som.toonRekensom();
som.vraagInvoer();
som.controleer();

Keersom keersom = new Keersom();
keersom.toonRekensom();
keersom.vraagInvoer();
keersom.controleer();

Deelsom deelsom = new Deelsom();
deelsom.toonRekensom();
deelsom.vraagInvoer();
deelsom.controleer();
```

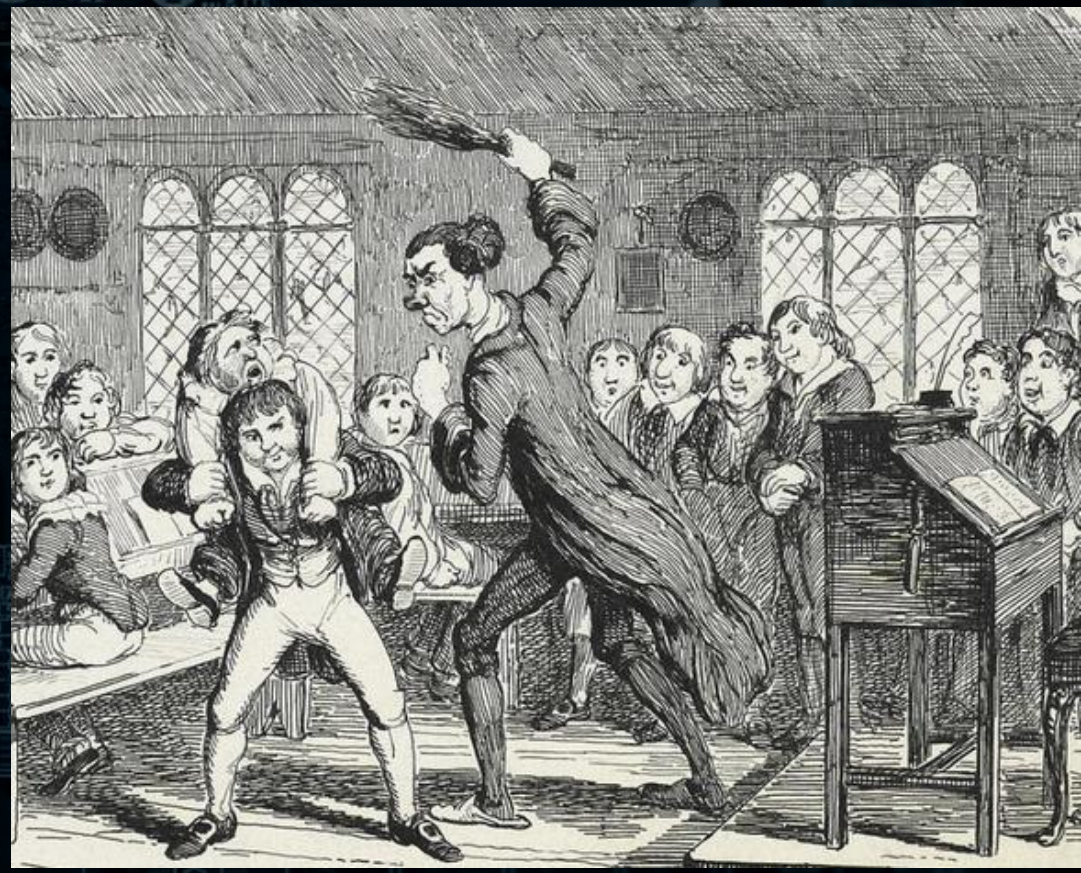
Case: polymorfisme bij “leren rekenen”




Main.java
package nl.saxion.ogp.learningmath;
import nl.saxion.app.*;
public class Main implements Runnable {
public static void main(String[] args) {
SaxionApp.start(new Main());
}
@Override
public void run() {
Optelsom som = new Optelsom();
som.toonRekensom();
som.vraagInvoer();
som.controleer();
Keersom keersom = new Keersom();
keersom.toonRekensom();
keersom.vraagInvoer();
keersom.controleer();
Deelsom deelsom = new Deelsom();
deelsom.toonRekensom();
deelsom.vraagInvoer();
deelsom.controleer();
}
}
Optelsom.java
public class Optelsom extends Rekensom {
@Override
public void toonRekensom() {
SaxionApp.printLine("Tel de volgende getallen
}
@Override
protected int berekenUitkomst() {
return getal1 + getal2;
}
}
Keersom.java
public class Keersom extends Rekensom {
@Override
public void toonRekensom() {
SaxionApp.printLine("Wat is " + getal1 + " keer
}
@Override
protected int berekenUitkomst() {
return getal1 * getal2;
}
}
Deelsom.java
public class Deelsom extends Rekensom{
@Override
public void toonRekensom() { SaxionApp.printLine("De
}
Rekensom.java
public class Rekensom {
protected int getal1;
protected int getal2;
protected int invoer;
public Rekensom() {
getal1 = SaxionApp.getRandomValueBetween(0, 20);
getal2 = SaxionApp.getRandomValueBetween(0, 10);
}
public void toonRekensom() {
SaxionApp.printLine("Doe iets met de volgende get
}
public void vraagInvoer() {
System.out.println("Hoi\nHoi");
SaxionApp.print("Hoi\n>\n ", Color.RED);
invoer = SaxionApp.readInt();
}
protected int berekenUitkomst() { throw new RuntimeE
public void controleer() {
int uitkomst = berekenUitkomst();
if (invoer == uitkomst) {
SaxionApp.printLine("Goed zo");
} else {
SaxionApp.printLine("grrr");
}
}
}

Case: polymorfisme bij “leren rekenen”

- Lees de documentatie van week 4, en bekijk het eerste filmpje.
- Pas class Main aan van het “leren rekenen” appje:
  - Zorg ervoor dat de drie verschillende “rekensommen” in één ArrayList opgeslagen worden
  - Roep in een lusje bij elke som de 3 methoden aan.



Kortom

Vandaag

- Polymorfisme via overerving

Morgen

- Klassenontwerp / modelleren