Gruppetime 5

Interface og beholdere

En kjapp oppsummering

Om interface



```
interface Grensesnitt {
    void tullemetode();
}

class Klasse implements Grensesnitt {
      @Override
    public void tullemetode(){
        System.out.println("Dette er tull!");
    }
}
```

- En samling egenskaper man sier en klasse skal ha
- Bruker nøkkelordet `implements`
- Inneholder metodene en klasse skal inneholde
- Man kan bruke nøkkelordet `default` for å lage en standard implementasjon

Interface kan også brukes som pekere

Hva skjer når man kjører dette?

```
interface Grensesnitt {
    void tullemetode();
}

class Klasse implements Grensesnitt {
    @Override
    public void tullemetode(){
        System.out.println("Dette er tull!");
    }
    public void enAnnenMetode(){
        System.out.println("Funker dette?")
    }
}

class Main {
    public static void main(String[] args) {
        Grensesnitt klasse = new Klasse();
        klasse.enAnnenMetode();
    }
}
```

Ukens pensum

Beholdere

- Hva slags beholdere kan dere komme på?

```
AbstractCollection, AbstractList, AbstractQueue, AbstractSequentialList, AbstractSet,
ArrayBlockingQueue, ArrayDeque, ArrayList, AttributeList, BeanContextServicesSupport,
BeanContextSupport, ConcurrentHashMap.KeySetView, ConcurrentLinkedDeque, ConcurrentLinkedQueue,
ConcurrentSkipListSet, CopyOnWriteArrayList, CopyOnWriteArraySet, DelayQueue, EnumSet, HashSet,
JobStateReasons, LinkedBlockingDeque, LinkedBlockingQueue, LinkedHashSet, LinkedList,
LinkedTransferQueue, PriorityBlockingQueue, PriorityQueue, RoleList, RoleUnresolvedList, Stack,
SynchronousQueue, TreeSet, Vector
```

La oss lage en enkel beholder for tall

- Lag en klasse med en tom int array på lengde 4 og en privat ‘int size`
- Implementer grensesnittet i klassen
- Prøv å fyll det opp med tall i en main-metode
- Deretter print innholdet i listen

```
public interface IntListe {
    public void add(int e);
    public int get(int index);
    public int size();
}
```

Interface kan også arve

La oss gjøre listen itererbar

- Dere må lage en ny klasse TallIterator
- Denne klassen må implementere interfacet `Iterator<Integer>` og ta inn listen i konstruktøren.

```
public interface IntListe extends Iterable<Integer> {
    public void add(int e);
    public int get(int index);
    public int size();

    @Override
    public default Iterator iterator() {
        return new TallIterator(this);
    }
}
```

```
public class TallIterator implements Iterator<Integer> {
    private IntListe liste;
    private int index = 0;
    public TallIterator(IntListe liste) {
        this.liste = liste;
    }
  ...
}
```

Kan vi lage en annen implementasjon av IntListe?

Lag en lenket liste

- Dere vil trenge en klasse Node som peker til neste, og som holder en verdi
- Den LenketListe-klassen må holde på start-noden
- Og må implementere de samme metodene i interfacet

```
public interface IntListe {
    public void add(int e);
    public int get(int index);
    public int size();
}
```

Generiske beholdere

Hvis vi har tid

Implementing av Exceptions

- For å gi oss litt kulere feilmeldinger
- Exceptions funker litt som objekter
- Denne klassen kan kalles på med: `throw new ListEmptyException`

```
public class ListEmptyException extends RuntimeException {
    public ListEmptyException(String message) {
        super(message);
    }
}
```