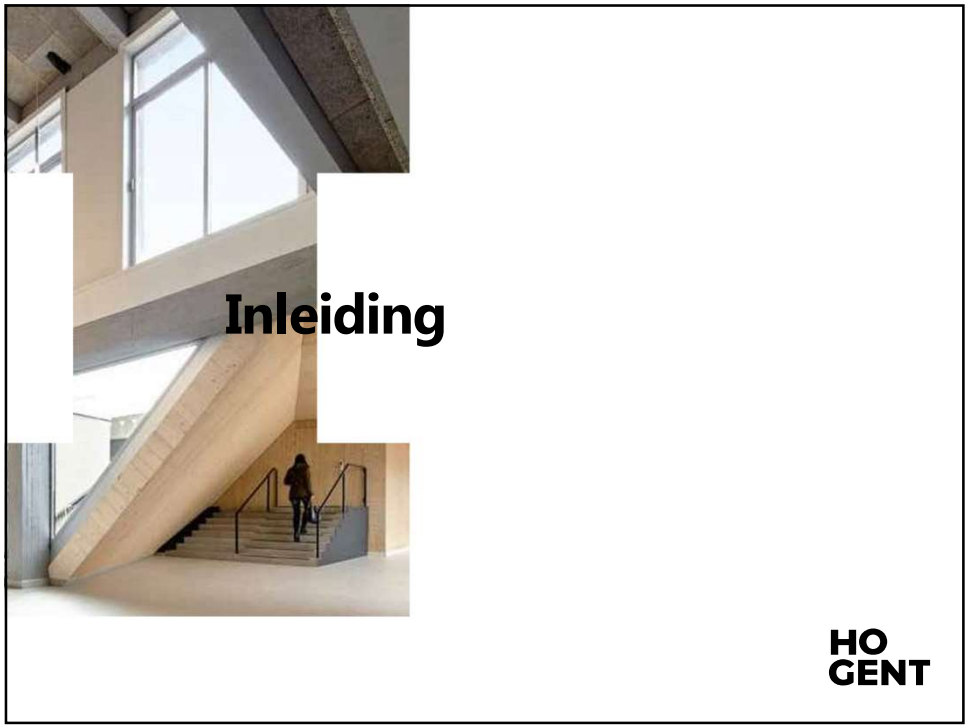

OO Programmeren III
Lamba's - extra
HO
GENT


Inleiding
HO
GENT

Voorbeeld eventafhandeling

```
button.setOnAction(
        new EventHandler<ActionEvent>()
            {
               @Override
               public void handle(ActionEvent event)
               {
                  System.out.println("button clicked");
               }
            });

button.setOnAction(
        event -> System.out.println("button clicked"));
```

HO GENT

Voorbeeld werken met collections

```
String[] friendsArray = {"Brian","Nate","Neal","Sara","Betty"};
List<String> friends = new ArrayList<>(Arrays.asList(friendsArray));

for(String name : friends)
        System.out.println(name);

friends.forEach(new Consumer<String>() {
        @Override
           public void accept(String t){
              System.out.println(t); }
    });
```

HO GENT

Voorbeeld werken met collections

Functionele interfaces kunnen herschreven worden

- via een lambda-expressie:

```
friends.forEach(
    (String t) -> { System.out.println(t);}    );
```

OF

```
friends.forEach( t -> System.out.println(t)    );
```

- via een method reference:

```
friends.forEach( System.out::println);
```

HO GENT

- **Lambda expressie** kan gebruikt worden waar functionele interfaces verwacht worden.
- **Lambda** = verkorte vorm van functionele interface
- **Methodreference** = verkorte vorm van lambda

HO GENT

Functionele interfaces

	Methode	Voorbeeld
Predicate<T>	boolean test(T t)	Voorwaarde controleren
Consumer<T> en ook BiConsumer<T,U>	void accept(T t)	Afdrukken van een waarde
Function<T,R>	R apply(T t)	Geef de naam van een artiest
Supplier<T>	T get()	Ophalen waarde
UnaryOperator<T>	T apply(T t)	Logische not
BinaryOperator<T> En ook BiFunction<T,U,R>	T apply(T t1, T t2)	2 getallen vermenigvuldigen

HO GENT

Voorbeelden

HO GENT

Enkele voorbeelden

```
String[] elementen;

for (String element: elementen)
        System.out.printf("%s ",element);

Arrays.stream(elementen)
          .forEach
                (s -> System.out.printf("%s ", s));
```

HO GENT

Doorlopen van een lijst

```
List<String> list;

for (String element : list)
       System.out.printf("%s ", element);

list.forEach(element ->
            System.out.printf("%s ", element));
```

HO GENT

Sorteren

/*Sorteer de arrayList op titel in STIJGENDE volgorde. Bij gelijke titels sorteren op isbn_nr in DALENDE volgorde.*/

Collections.sort(
boeken, new BoekComparator());

HO GENT

```
private class BoekComparator implements Comparator<Boek>
  {
    @Override
     public int compare( Boek boek1, Boek boek2 )
      {
        int boekCompare = boek1.getTitel().
                                    compareTo( boek2.getTitel() );
        if ( boekCompare != 0 )
              return boekCompare;
        return
          Long.compare(boek2.getIsbn_nr(),boek1.getIsbn_nr());
      }
  }
```

HO GENT

Vanaf java8

```
boeken.sort(
    Comparator.comparing(Boek::getTitel())
                .thenComparing( Comparator.comparing
                                (Boek::getIsbn_nr().reversed())
    );
```

HO GENT

Toon het kleinste element van de array getallen

```
//JAVA 7
System.out.printf("min = %d (oplossing = 3)\n",
        Collections.min(Arrays.asList(getallen)));

//JAVA 8
System.out.printf("min = %d (oplossing = 3)\n",
  Arrays.stream(getallen).
            min(Integer::compare).get());
```

HO GENT

Toon het grootste element van de array getallen

```
//JAVA 7
System.out.printf("max = %d (oplossing = 9)\n",
        Collections.max(Arrays.asList(getallen)));

//vanaf JAVA 8
System.out.printf("max = %d (oplossing = 9)\n",
Arrays.stream(getallen).max(Integer::compare)
                                .get());
```

HO GENT

Doorlopen van een map

```
//JAVA 7
for ( Map.Entry<String,Double> eenEntry : fruit.entrySet() ){
    System.out.printf("Prijs van %s : ", eenEntry.getKey());
    double prijs = in.nextDouble();
    eenEntry.setValue(prijs);
}

//vanaf JAVA 8
fruit.entrySet().forEach(eenEntry -> {
        System.out.printf("Prijs van %s : ", eenEntry.getKey());
        double prijs = in.nextDouble();
        eenEntry.setValue(prijs);
});
```

HO GENT

Doorlopen van een map (vervolg)

```
//JAVA 7
for (Map.Entry<String, Double> eenEntry : fruit.entrySet())
        System.out.printf("%s\t%.2f%n",
                eenEntry.getKey(), eenEntry.getValue());

//vanaf JAVA 8
 fruit.forEach(
    (sleutel,waarde) -> System.out.printf("%s\t%.2f%n",
                                          sleutel ,waarde));
```

HO GENT

Opzetten van een Map

```
Map<String,List<Student>> mapPerWoonplaats =
                        new HashMap<>();
String huidigeKey;

for(Student s : lijstStudenten)
{
    huidigeKey=s.getWoonplaats();
    if(!mapPerWoonplaats.containsKey(huidigeKey))
       mapPerWoonplaats.put(huidigeKey, new ArrayList<>());

    mapPerWoonplaats.get(huidigeKey).add(s);
}
```

HO GENT

Opzetten van een Map – vanaf Java8

```
Map<String, List<Student>> byWoonplaats
   = lijstStudenten.stream()
                   .collect(Collectors.groupingBy(
                                   Student::getWoonplaats));
```

HO GENT

- Extra voorbeelden:
  - Zie chamilo => StreamsLambdasExtraVbn