PGdP

Datenstrukturen

Enums

Streams

Method References

P-Aufgaben

Folien: go.tum.de/904005

Tobias Schamel

Datenstrukturen

Struktur	Zugriff	Suche	Einfügen	Löschen	Sonstiges
Array					Limitierte Größe
Stack (array)					Limitierte Größe
Stack (list)					
LinkedList					
ArrayList					Limitierte Größe
HashTable					Kollisionen möglich

Hashing: beliebige Elemente werden über einfache Funktionen auf feste Zeichenketten abgebildet.

Datenstrukturen

Struktur	Zugriff	Suche	Einfügen	Löschen	Son
Array					Lim
Stack (array)					Lim
Stack (list)					
LinkedList					
ArrayList					Limitierte Größe
HashTable					Kollisionen möglich

Für interessierte:
BigOCheatSheet

Hashing: beliebige Elemente werden über einfache Funktionen auf feste Zeichenketten abgebildet.

Enums

Aufzählungstyp für verschiedene Zustände/Konstanten

```
public enum Jahreszeit {
   Frühling("spring"), Sommer("summer"),
    Herbst("autumn"), Winter("winter");
   private String engName;
   private Jahreszeit(String engName) {
      this.engName = engName;
   }
   public String toString() { return this.engName; }
}
```

Enums

Aufzählungstyp für verschiedene Zustände/Konstanten

```
public enum Jahreszeit {
   Frühling {
     public String toString() { return "spring"; }
   },
   Sommer {
     public String toString() { return "summer"; }
   },
   Herbst {
     public String toString() { return "autumn"; }
   },
   // ... für Winter
}
```

Enums

Aufzählungstyp für verschiedene Zustände/Konstanten

```
switch (jz) {
   case Frühling: write("Frühling"); break;
   case Sommer: write("Sommer"); break;
   case Herbst: write("Herbst"); break;
   case Winter: write(
    Jahreszeit.Winter.toString()); break;
   //default: break;
}
```

Streams

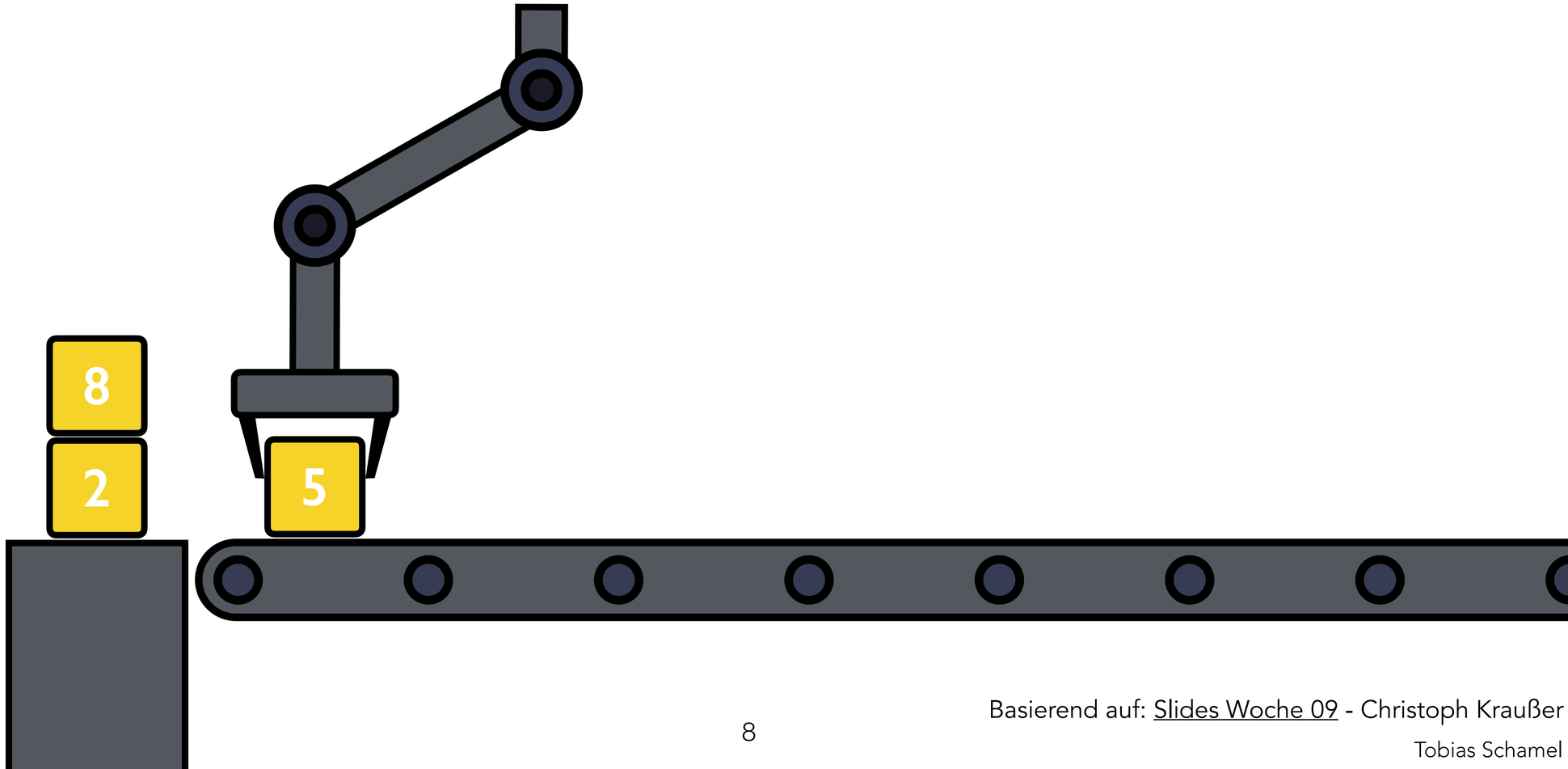
Fließband, auf dem für jedes Teil Operationen ausgeführt werden.

Beispiel:

```
Stream.of(5, 8, 2).map(i -> i+1)
   .filter(i -> i%2 != 0);
```

Streams

Fließband, auf dem für jedes Teil Operationen ausgeführt werden.

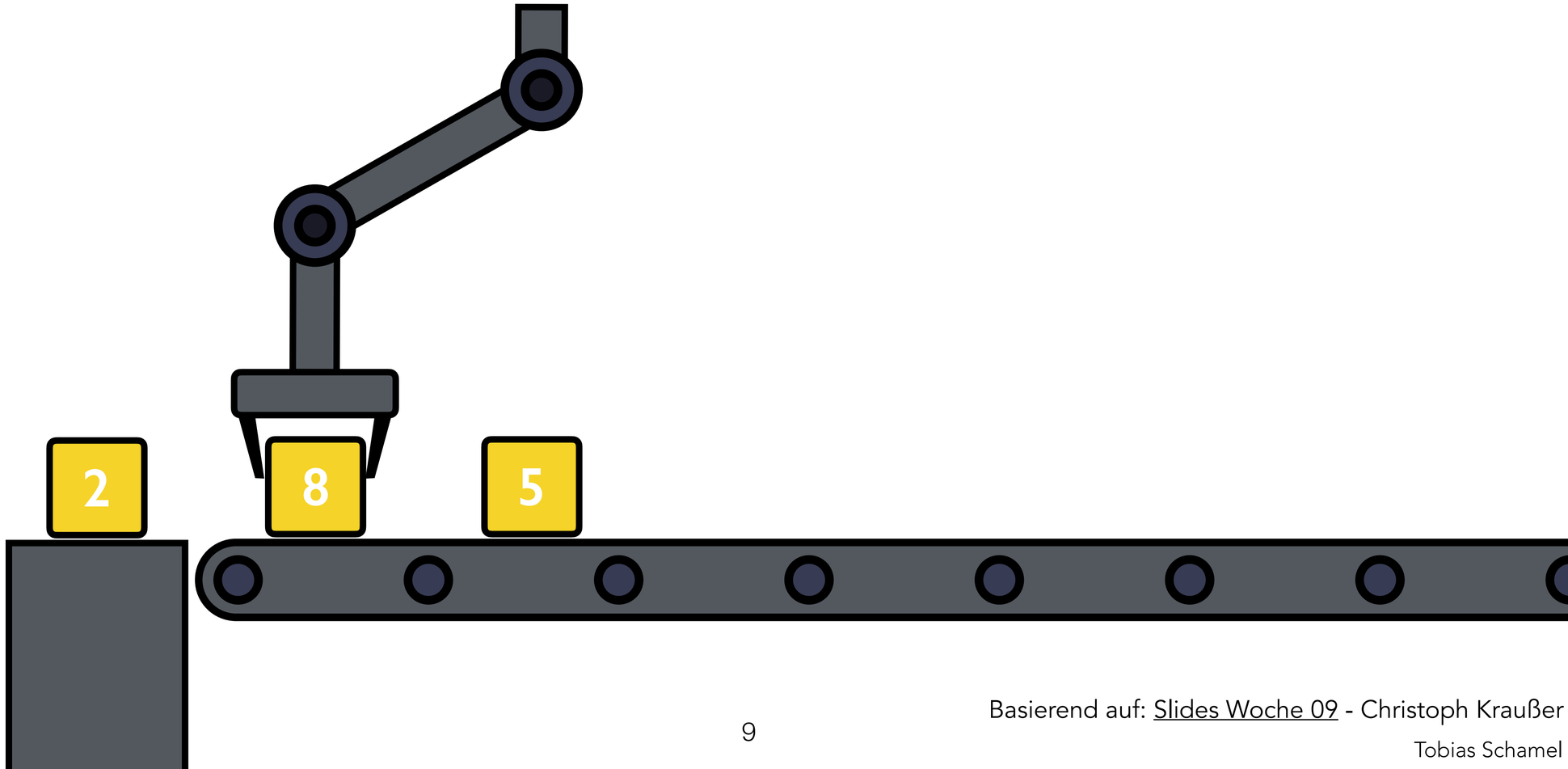


Basierend auf: Slides Woche 09 - Christoph Krauß

Tobias Schamel

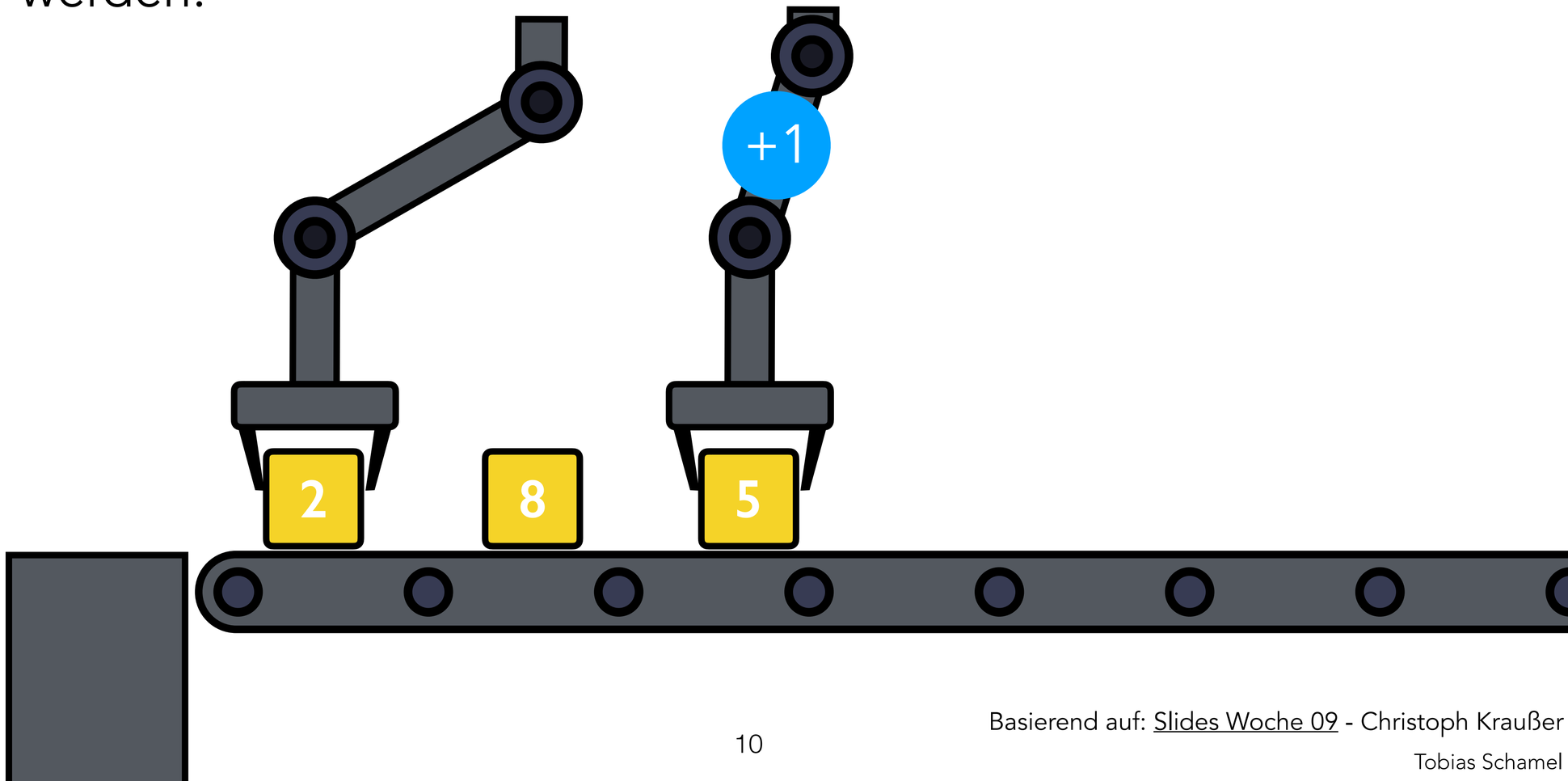
Streams

Fließband, auf dem für jedes Teil Operationen ausgeführt werden.



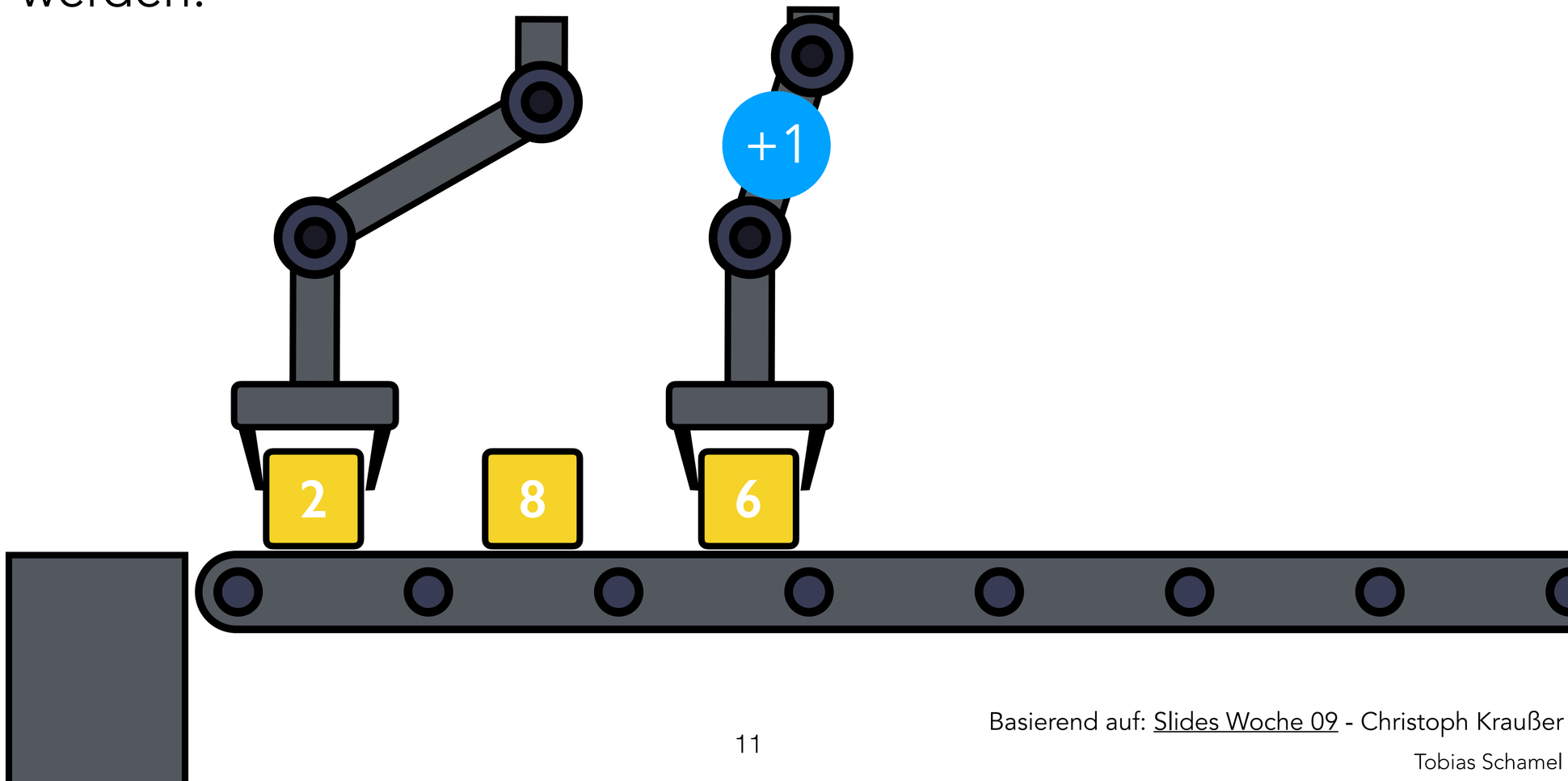
Streams

Fließband, auf dem für jedes Teil Operationen ausgeführt werden.



Streams

Fließband, auf dem für jedes Teil Operationen ausgeführt werden.

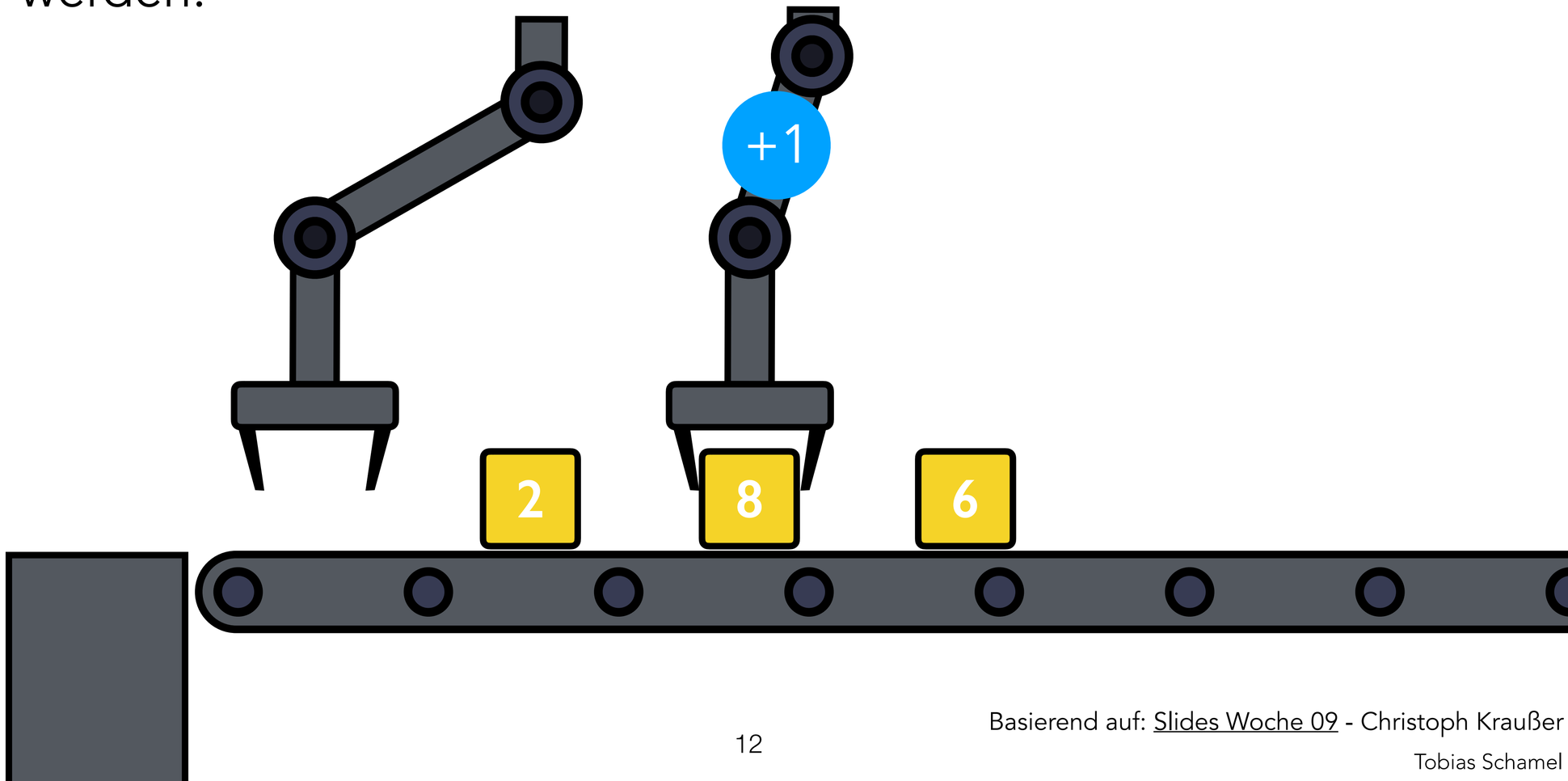


Basierend auf: Slides Woche 09 - Christoph Krauß

Tobias Schamel

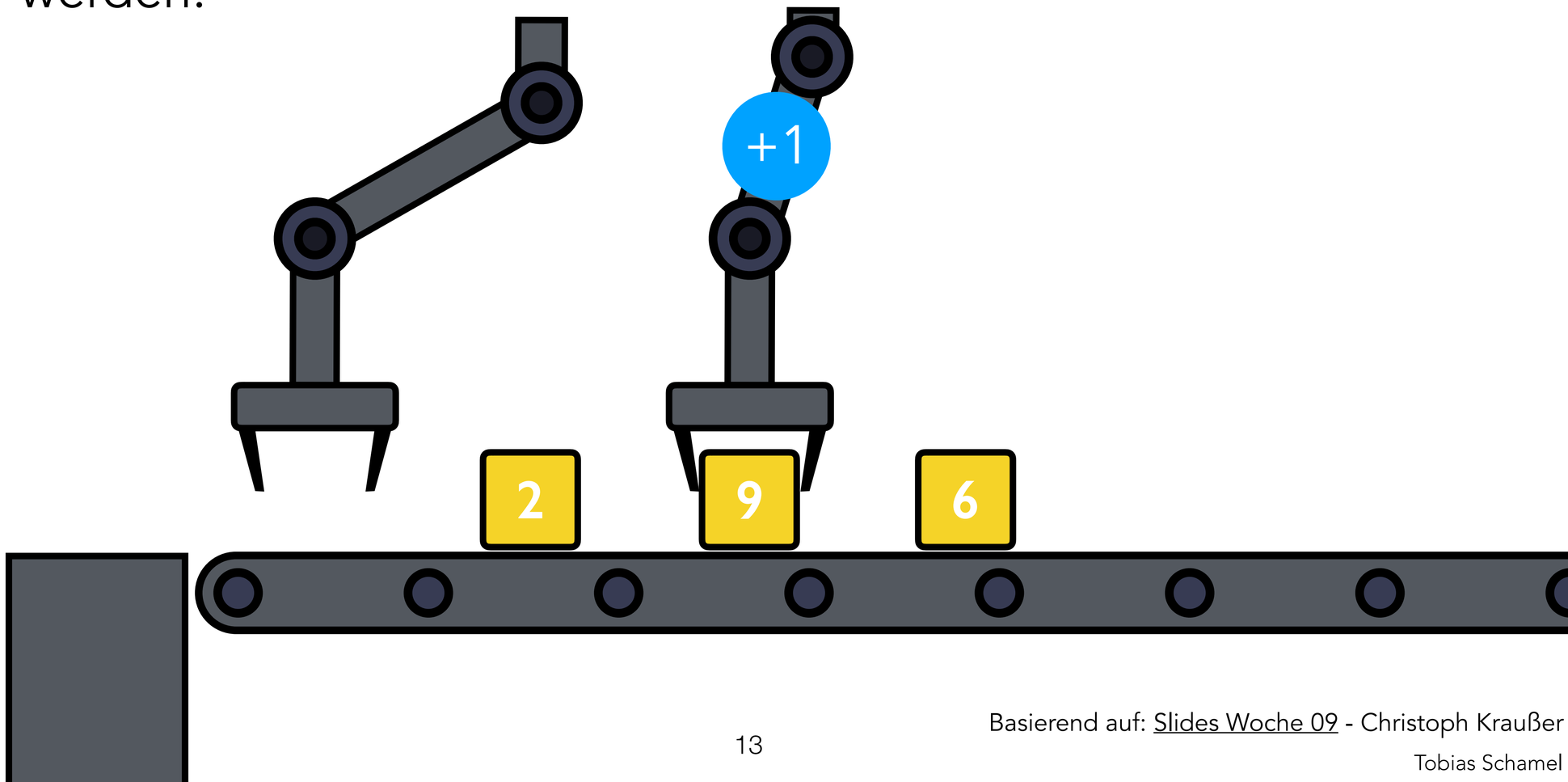
Streams

Fließband, auf dem für jedes Teil Operationen ausgeführt werden.



Streams

Fließband, auf dem für jedes Teil Operationen ausgeführt werden.

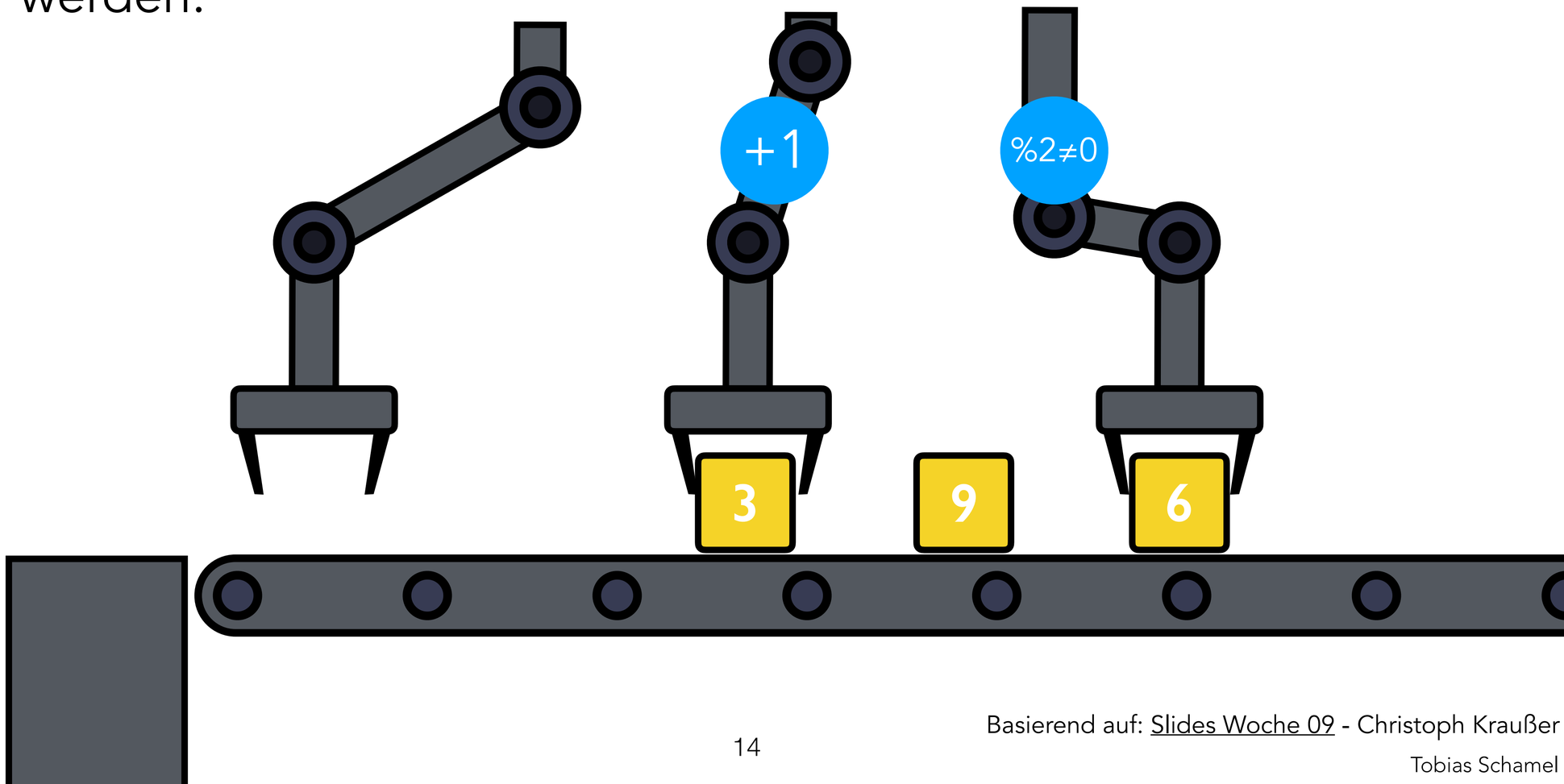


Basierend auf: Slides Woche 09 - Christoph Krauß

Tobias Schamel

Streams

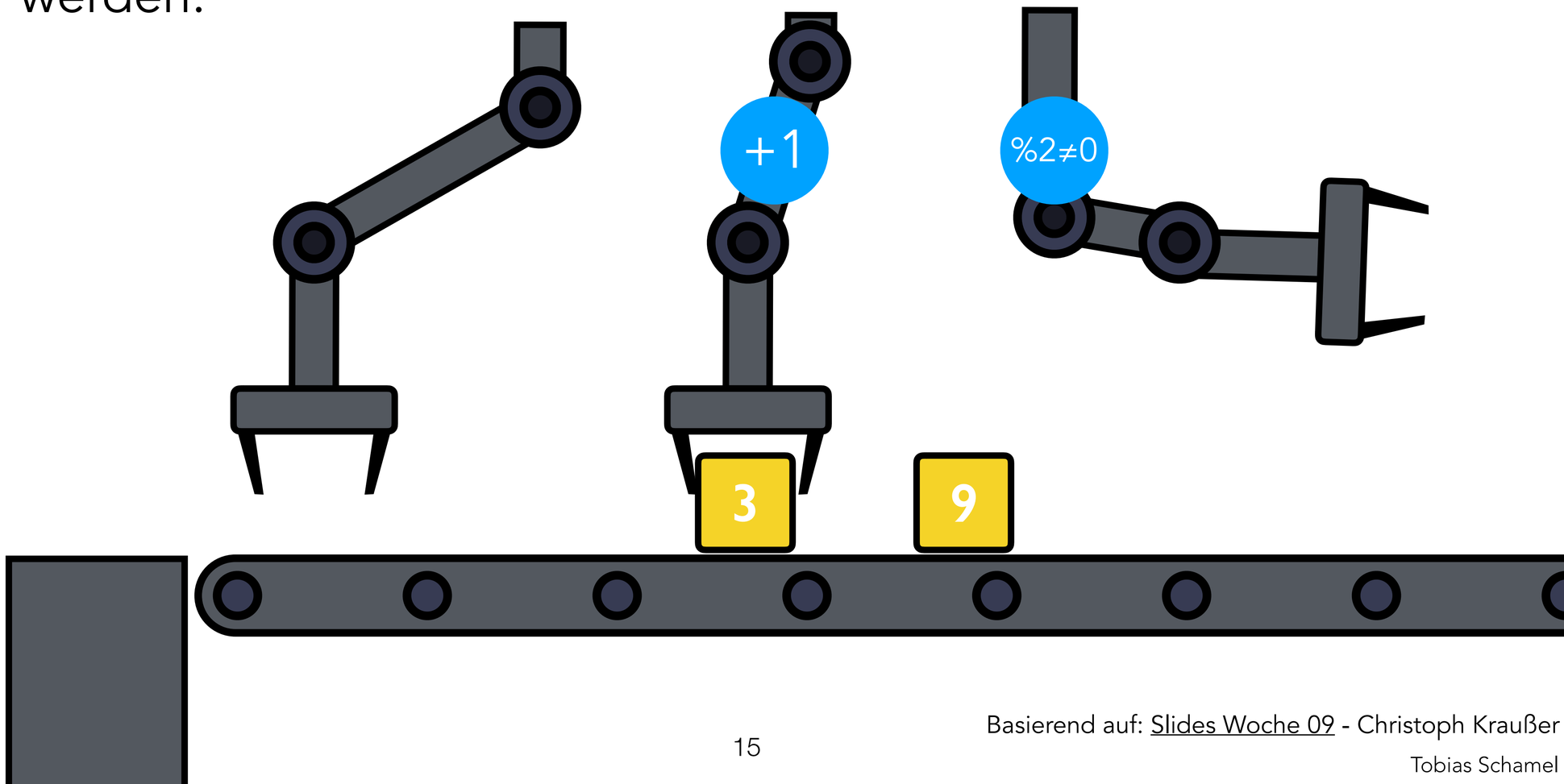
Fließband, auf dem für jedes Teil Operationen ausgeführt werden.



Basierend auf: Slides Woche 09 - Christoph Krauß
Tobias Schamel

Streams

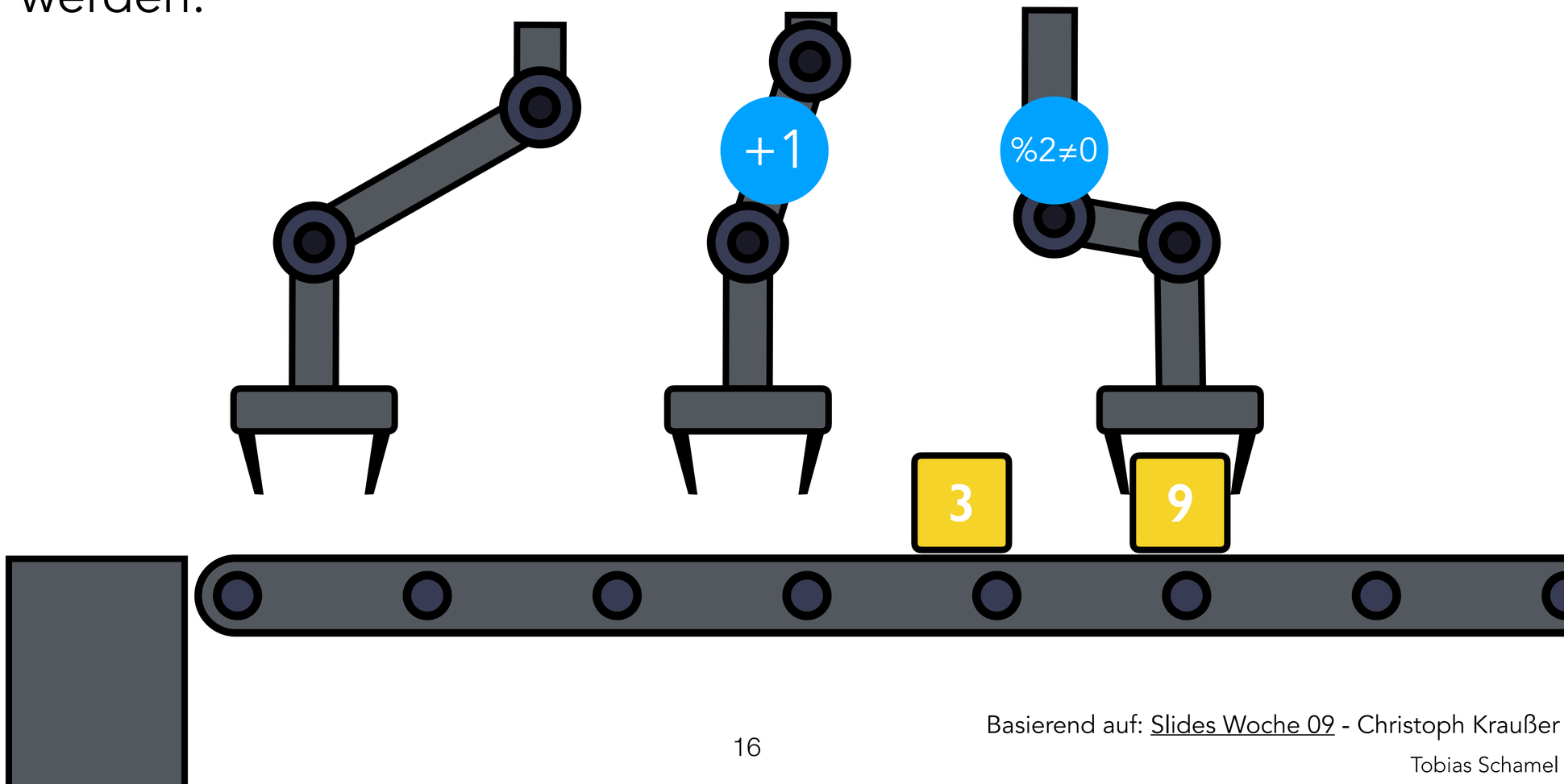
Fließband, auf dem für jedes Teil Operationen ausgeführt werden.



Basierend auf: Slides Woche 09 - Christoph Krauß
Tobias Schamel

Streams

Fließband, auf dem für jedes Teil Operationen ausgeführt werden.

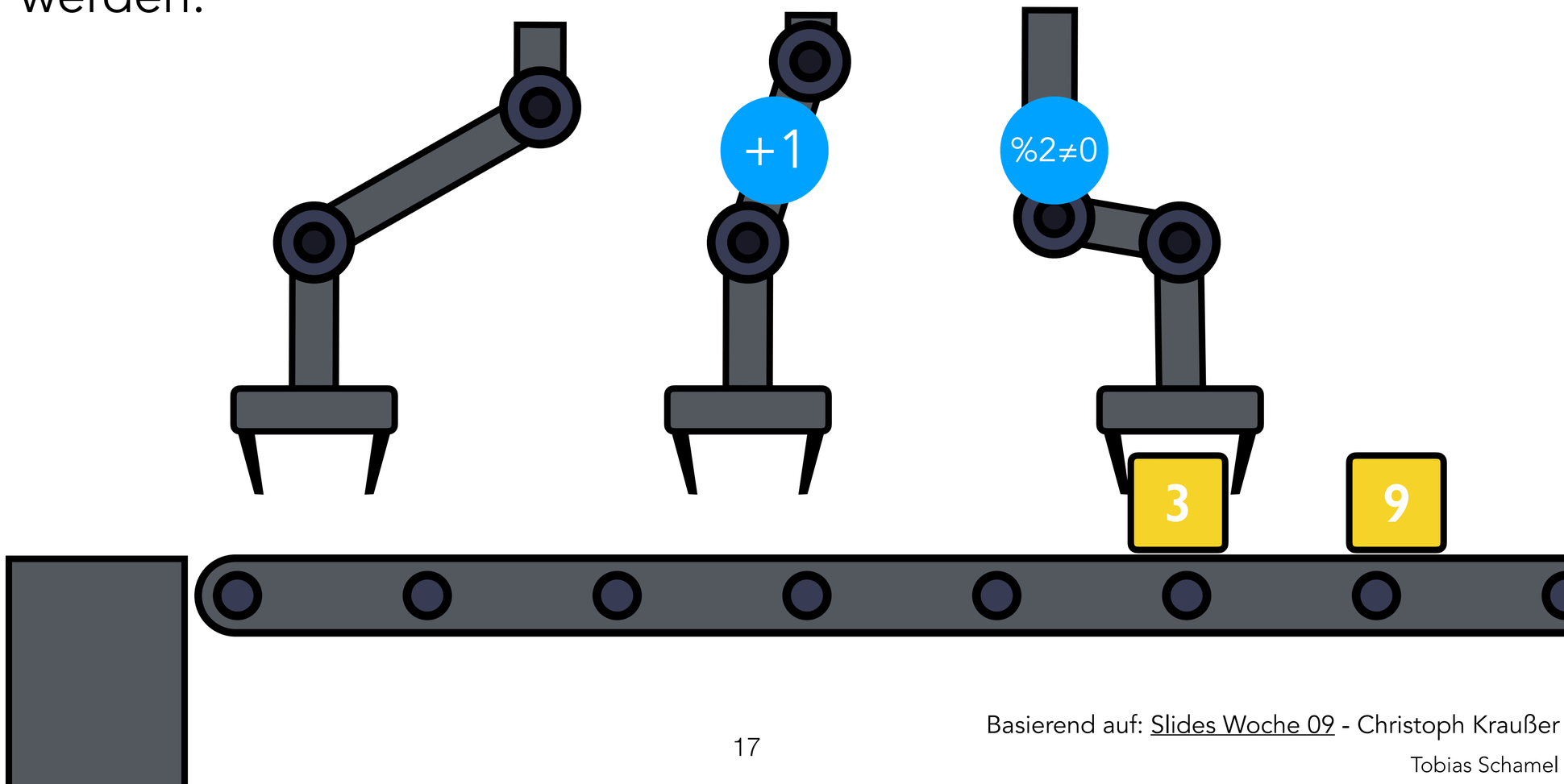


Basierend auf: Slides Woche 09 - Christoph Krauß

Tobias Schamel

Streams

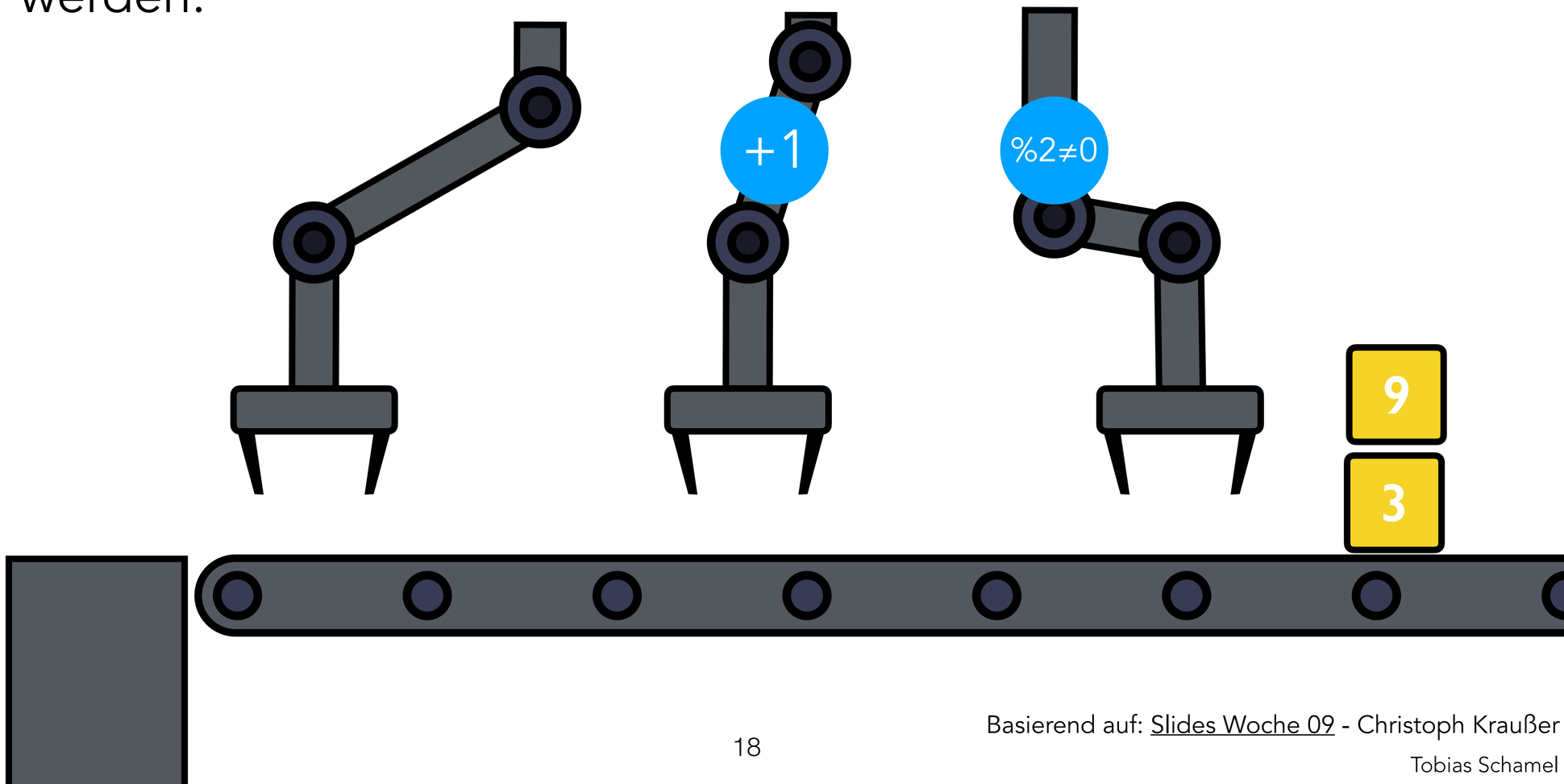
Fließband, auf dem für jedes Teil Operationen ausgeführt werden.



Basierend auf: Slides Woche 09 - Christoph Krauß
Tobias Schamel

Streams

Fließband, auf dem für jedes Teil Operationen ausgeführt werden.



Basierend auf: Slides Woche 09 - Christoph Krauß

Tobias Schamel

Streams

Die wichtigsten Streamfunktionen

Streams

Recap Lambda Ausdrücke:

Lambda Ausdrücke können wie ein Methodenargument übergeben werden.

```
Tree<Penguin> tree =
 new Tree<Penguin>((o1, o2) -> o1.age - o2.age);
```

Parameter

Methodenkörper, implizit auch return Wert

```
//für Konstruktor Tree<T>(Comparator comp)
```

Streams

Die wichtigsten Streamfunktionen: map

- Jedes Element wird mit einer Funktion verändert
  > auch Datentyp kann geändert werden
- Gibt einen neuen (veränderten) Stream zurück

```
IntStream.range(1,10).map(i -> i*i);
 → [1, 4, 9, 16, 25, 36, 49, 64, 91]


.map(Function<From, To> func);
```

Streams

Die wichtigsten Streamfunktionen: filter

- Elemente nach Prädikat (Eigenschaft) filtern
- Gibt einen neuen (veränderten) Stream zurück

```
IntStream.range(1,10).filter(i -> i > 4);
 → [5, 6, 7, 8, 9]

.filter(Predicate<T> pred);
```

Streams

Die wichtigsten Streamfunktionen: reduce

- Alle Elemente zusammenfassen
- Gibt keinen neuen Stream zurück (Terminal)

```
IntStream.range(1,10).reduce(0, (a, b) -> a+b);
 → 45
```

aktuelles Element

altes Element

Startwert

Parameter (Lamda Expression)

Rückgabewert

```
.reduce(T identity,
 BinaryOperator<T>
 accumulator);
```

Streams

Die wichtigsten Streamfunktionen: forEach

- Operation auf jedes Element ausführen
- Gibt keinen neuen Stream zurück (Terminal)

```
IntStream.range(1,10).forEach(i -> write(i));


.forEach(Consumer<T> action);
```

Streams

Die wichtigsten Streamfunktionen: collect

- Elemente in andere Datenstruktur zusammenfassen
- Gibt keinen neuen Stream zurück (Terminal)

```
Stream.of(1, 2, 3, 4, 5)
 .collect(Collectors.toList());
```

```
.collect(Collection<T> col);
```

Übersicht über weitere Collectors

Streams

Streams erstellen:

- Collections können in Streams umgewandelt werden

```
(new ArrayList<Integer>).stream();
```

- Arrays können in Streams umgewandelt werden

```
Arrays.stream(new int[4]);
```

- Streams können konkret erstellt werden

```
Stream.of(Jahreszeit.Sommer,
 Jahreszeit.Frühling, Jahreszeit.Winter);
```

- Streams für primitive Datentypen

```
IntStream.rangeClosed(1, 10);
```

Method References

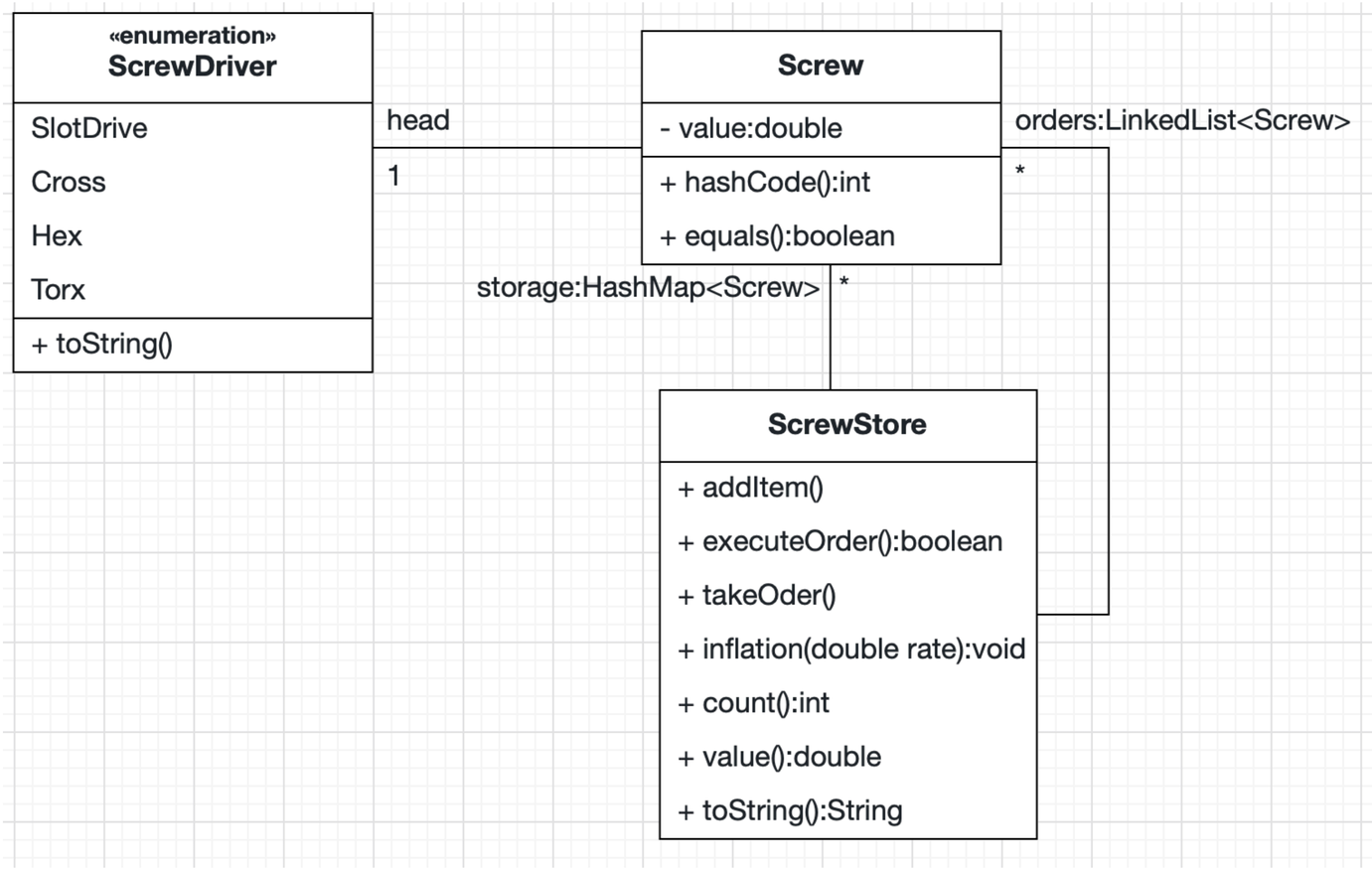
Statt Lamda Expressions können Referenzen zu Methoden übergeben werden (wenn der übergebene Parameter eindeutig ist).

```
IntStream.range(1,10)
 .forEach(System.out::println);
```

- statische Methoden: `className::methodName`
- Instanzmethoden: `objName::methodName`
  > in Lambdas auch `className::methodName`
- Konstruktoren: `className::new`

P09.01

Schraubenladen (Anwendung Enums)



P09.02

Streameinführung

```
public static double calculate(List<Double> input);
```

> negative Zahlen ignorieren, quadrieren, Summe berechnen

```
public static Set<Person>
 toSetForEach(List<Person> input);
```

> Eingabe mit forEach in ein Set umwandeln

```
public static Set<Person> toSet(List<Person> input);
```

> Eingabe mit Collector in ein Set umwandeln

P09.02

Streameinführung

```
public static double average(int[] input);
```

> Durchschnitt berechnen, siehe .average()

```
public static double averageAge(List<Person> input);
```

> Durchschnittliches Alter der Personen berechnen

P09.02

Streameinführung

```
public static Map<Integer, List<Person>>
 groupByAgeForEach(List<Person> input);
```

> Map mit nach Alter separierten Listen erstellen (mit forEach)

```
public static Map<Integer, List<Person>>
 groupByAge(List<Person> input);
```

> Map mit nach Alter separierten Listen erstellen, siehe (mit Collector) `Collectors.groupingBy()`

P09.03

Temperaturen

long size() No results
- für die Anzahl der gemessenen Temperaturen,

List<Date> dates() No results
- für die Liste aller aufsteigend sortierten Tage ohne Duplikate von allen gemessenen Temperaturen,

Set<String> cities() No results
- für die Menge aller Städte von allen gemessenen Temperaturen,

Set<String> countries() No results
- für die Menge aller Länder von allen gemessenen Temperaturen,

Map<String, List<Double>> temperaturesByCountry() No results
- für die Menge aller gemessenen Temperaturen gruppiert nach Ländernamen (ohne Test),

String coldestCountryAbs() No results
- für das Land mit der kältesten gemessenen Temperatur,

String hottestCountryAbs() No results
- für das Land mit der heißesten gemessenen Temperatur,

P09.03

Temperaturen

Map<String, Double> countriesAvgTemperature() No results

- für Ländernamen plus deren dazugehörigen Durchschnittstemperatur über alle Zeiten und beinhaltenden Städte.

Map<String, Double> avgTemperatureDeltaPerYearPerCountry() No results

- Diese Methode gibt ein Mapping von Ländernamen auf das mittlere jährliche Temperaturdelta im gesamten Zeitraum zurück. Diese Map soll zudem das globale Temperaturdelta als Durchschnitt über alle Länder im Key `Globally` enthalten. Das Temperaturdelta zwischen zwei Jahren ist nichts anderes als die Temperaturdifferenz der jährlichen Durchschnittstemperatur. Beispielsweise würde man die Durchschnittstemperatur im Jahr 1900 von der Durchschnittstemperatur im Jahr 1901 abziehen, um das Delta von 1900 auf 1901 für eine bestimmtes Land zu erhalten. Das mittlere Temperaturdelta ist dann der Durchschnitt aus allen Deltas.